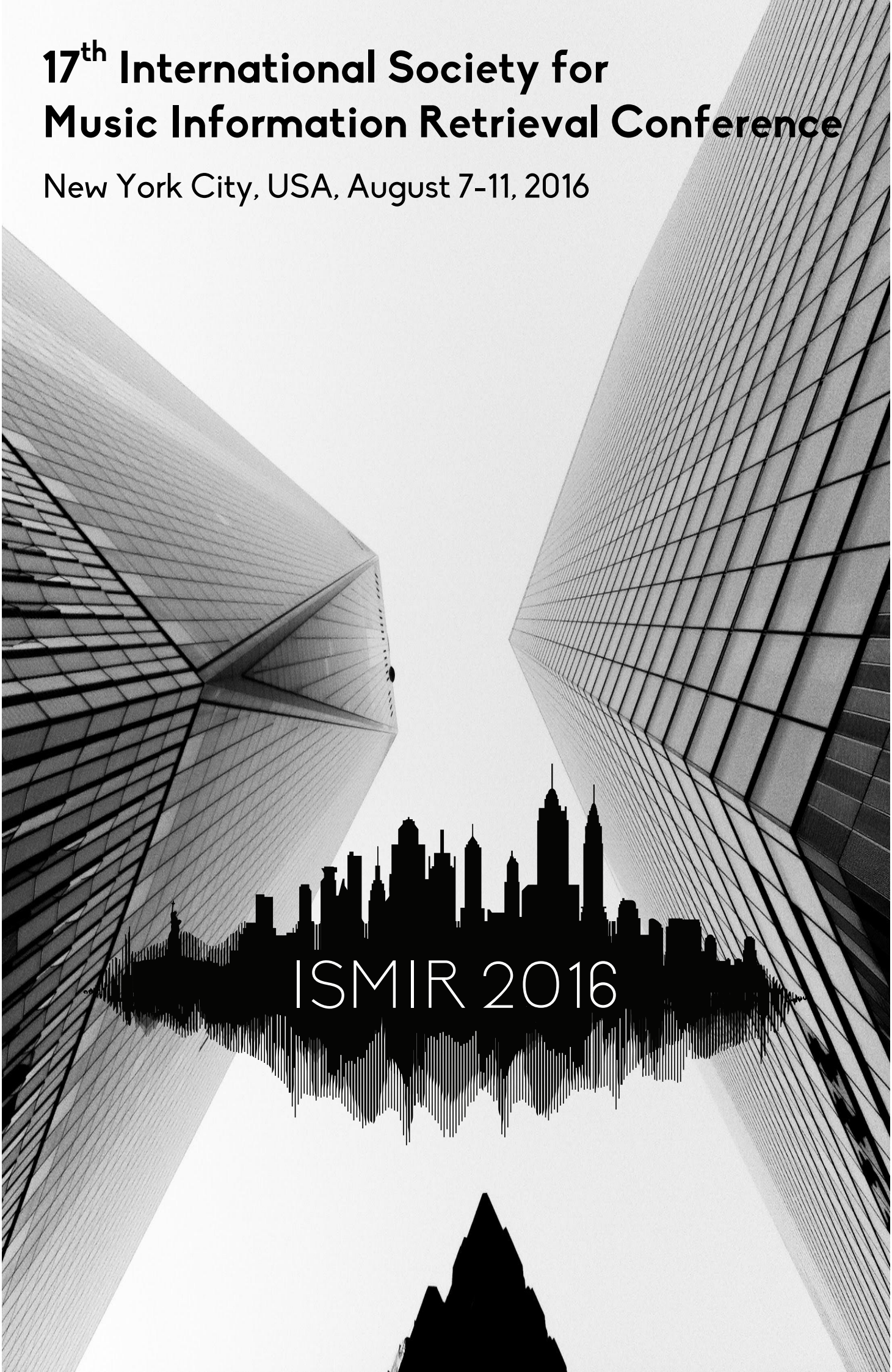
17th International Society for Music Information Retrieval Conference

New York City, USA, August 7-11, 2016



ISMIR 2016

Proceedings of the 17th International Society for Music Information Retrieval Conference

August 7 - 11, 2016
New York City, USA

Edited by
Johanna Devaney, Michael I Mandel, Douglas Turnbull, and George Tzanetakis

ISMIR 2016 is organized by the International Society for Music Information Retrieval, New York University, and Columbia University.

Website: https://wp.nyu.edu/ismir2016/

Cover design by Michael Mandel
Cover photo by Justin Salamon
ISMIR 2016 logo by Justin Salamon and Claudia Henao

Edited by:

Johanna Devaney (The Ohio State University)
Michael I Mandel (Brooklyn College, City University of New York)
Douglas Turnbull (Ithaca College)
George Tzanetakis (University of Victoria)

ISBN-13: 978-0-692-75506-8
ISBN-10: 0-692-75506-3

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2016 International Society for Music Information Retrieval

Organizers

MARL

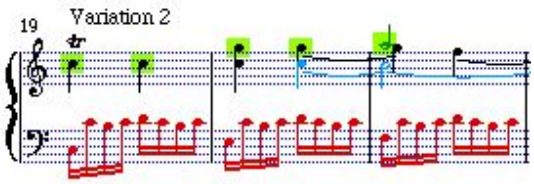
Lab ROSA

Laboratory for the Recognition and Organization of Speech and Audio

NYU | STEINHARDT

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

ISMIR



Partners

Diamond Partners

Grant # 1646882

Platinum Partners

Google

HARMONIX

Gold Partners

ACRCloud

NATIVE INSTRUMENTS

Yandex

Silver Partners

Shazam

Conference Organizing Committee

General Chairs
Juan Pablo Bello, New York University
Dan Ellis, Columbia University / Google, Inc

Program Chairs
Johanna Devaney, Ohio State University
Doug Turnbull, Ithaca College
George Tzanetakis, University of Victoria

Publication Chair
Michael Mandel, Brooklyn College, CUNY

Tutorial Chair
Fabien Gouyon, Pandora

Local Organization
Rachel Bittner, New York University
Dawen Liang, Columbia University
Brian McFee, New York University
Justin Salamon, New York University

Industry Partnerships
Eric Humphrey, Spotify

Women in MIR
Amélie Anglade, Freelance MIR & Data Science Consultant
Blair Kaneshiro, Stanford University

Education Initiatives
S. Alex Ruthmann, New York University
Andy Sarroff, Dartmouth College

Late Breaking, Demos, Unconference
Colin Raffel, Columbia University

Advisory Board
Michael Casey, Dartmouth College
Ichiro Fujinaga, McGill University
Youngmoo Kim, Drexel University
Carol Krumhansl, Cornell University
Paul Lamere, Spotify
Agnieszka Roginska, New York University

Program Committee

Jean-Julien Aucouturier, IRCAM/CNRS
Emmanouil Benetos, Queen Mary University of London
Sebastian Böck, Johannes Kepler University
John Ashley Burgoyne, University of Amsterdam
Ching-Hua Chuan, University of North Florida
Tom Collins, De Montfort University
Sally Jo Cunningham, Waikato University
Roger Dannenberg, Carnegie Mellon University
Matthew Davies, INESC TEC
W. Bas de Haas, Chordify/Utrecht University
Christian Dittmar, International Audio Laboratories Erlangen
Simon Dixon, Queen Mary University of London
J. Stephen Downie, Graduate School of Library & Information Science, University of Illinois
Zhiyao Duan, University of Rochester
Douglas Eck, Google, Inc.
Andreas Ehmann, Pandora
Sebastian Ewert, Queen Mary University of London
George Fazekas, Queen Mary University of London
Ben Fields, Goldsmiths University of London
Arthur Flexer, Austrian Research Institute for Artificial Intelligence
Jose Fornari, Universidade Estadual de Campinas
Ichiro Fujinaga, McGill University
Emilia Gómez, Universitat Pompeu Fabra
Masataka Goto, National Institute of Advanced Industrial Science and Technology
Fabien Gouyon, Pandora
Maarten Grachten, Austrian Research Institute for Artificial Intelligence
Andrew Hankinson, University of Oxford
Dorien Herremans, Queen Mary University of London
Perfecto Herrera, Universitat Pompeu Fabra
Andre Holzapfel, Austrian Research Institute for Artificial Intelligence
Xiao Hu, University of Hong Kong
Eric Humphrey, Spotify
Ozgur Izmirli, Connecticut College
Peter Knees, Johannes Kepler University Linz, Austria

Paul Lamere, Spotify
Audrey Laplante, Université de Montréal
Jin Ha Lee, University of Washington
Cynthia Liem, Delft University of Technology
Michael Mandel, Brooklyn College, CUNY
Matt McVicar, University of Bristol
Meinard Mueller, International Audio Laboratories Erlangen
Nicola Orio, University of Padua
Kevin Page, University of Oxford
Helene Papadopoulos, Centre national de la recherche scientifique
Geoffroy Peeters, Institut de Recherche et Coordination Acoustique/Musique
Matthew Prockup, Drexel University / Pandora
Marcelo Queiroz, University of São Paulo
Erik Schmidt, Pandora
Jeffrey Scott, Gracenote, Inc.
Xavier Serra, Universitat Pompeu Fabra
Mohamed Sordo, Pandora
Bob Sturm, Queen Mary University of London
Li Su, Academia Sinica
David Temperley, Eastman School of Music
Julián Urbano, Universitat Pompeu Fabra
Peter Van Kranenburg, Meertens Institute
Anja Volk, Utrecht University
Ju-Chiang Wang, Cisco Systems Inc.
Yi-Hsuan Yang, Academia Sinica
Kazuyoshi Yoshii, Kyoto University

Reviewers

Jakob Abesser
Islah Ali-MacLachlan
Joakim Anden
Tom Arjannikov
Leandro Balby
Stefan Balke
Isabel Barbancho
Ana Maria Barbancho
Alejandro Bellogín
Brian Bemman
Gilberto Bernardes
Louis Bigo
Jean-Robert Bisaillon
Laura Bishop
Dmitry Bogdanov
Ciril Bohak
Jordi Bonada
Antonio Bonafonte
Juanjo Bosch
Keki Burjorjee
Donald Byrd
Marcelo Caetano
Carlos Eduardo Cancino Chacon
Estefania Cano
Julio Carabias
Rafael Caro Repetto
Mark Cartwright
Dorian Cazau
Tak-Shing Chan
Chih-Ming Chen
Shuo Chen
Srikanth Cherla
Yu-Hao Chin
Kahyun Choi
Chia-Hao Chung
Andrea Cogliati
Debora Correa
Courtenay Cotton
Emanuele Coviello
Brecht De Man
Norberto Degara
Junqi Deng
Arnaud Dessein
Chandra Dhir
Simon Durand
Hamid Eghbal-Zadeh
Slim Essid
Steve Essinger
Zhe-Cheng Fan
Ángel Faraldo
Tiago Fernandes Tavares
Thomas Fillon
Derry Fitzgerald
Frederic Font
Peter Foster
Justin Gagen
Joachim Ganseman
Martin Gasser
Aggelos Gkiokas
Rolf Inge Godøy
Rong Gong
Yupeng Gu
Sankalp Gulati
Chun Guo
Michael Gurevich
Masatoshi Hamanaka
Pierre Hanna
Yun Hao
Steven Hargreaves
Thomas Hedges
Jason Hockman
Yu-Hui Huang
Po-Sen Huang
Katsutoshi Itoyama
Maximos Kaliakatsos-Papakostas
Blair Kaneshiro
Andreas Katsiavalos
Rainer Kelz
Minje Kim
Anssi Klapuri
Rainer Kleinertz
Alessandro Koerich
Vincent Koops
Filip Korzeniowski
Florian Krebs
Nadine Kroher
Anna Kruspe
Lun-Wei Ku
Frank Kurth
Mathieu Lagrange
Robin Laney
Thibault Langlois
Olivier Lartillot
Kyogu Lee
Sang Won Lee
Bernhard Lehner
Matthias Leimeister
Mark Levy
David Lewis
Richard Lewis
Bochen Li
Dawen Liang
Thomas Lidy
Elad Liebman
Yuan-Pin Lin
Jen-Yu Liu
Yi-Wen Liu
Marco Liuni
Antoine Liutkus
Jörn Loviscach
Lie Lu
Hanna Lukashevich
Athanasios Lykartsis
Patricio López-Serrano

Anderson Machado
Akira Maezawa
José Pedro Magalhães
Matija Marolt
Mónica Marrero
Alan Marsden
David Martins de Matos
Agustin Martorell
Panayotis Mavromatis
Brian McFee
Cory McKay
Cedric Mesnage
Andrew Milne
Stylianos Mimilakis
Marius Miron
Dirk Moelants
Josh Moore
Brandon Morton
Marcela Morvidone
Daniel Müllensiefen
Hidehisa Nagano
Eita Nakamura
Maria Navarro
Kerstin Neubarth
Eric Nichols
Oriol Nieto
Mitsunori Ogihara
Maurizio Omologo
Sergio Oramas
Vito Ostuni
Ken O'Hanlon
Francois Pachet
Rui Pedro Paiva
Gabriel Parent
Tae Hong Park
Jouni Paulus
Alfonso Perez
Lucien Perticoz
Antonio Pertusa
Pedro Pestana
Aggelos Pikrakis
Jordi Pons
Phillip Popp
Alastair Porter
Laurent Pugin
Elio Quinton
Colin Raffel
Zafar Rafii
Mathieu Ramona
Preeti Rao
Iris Ren
Gang Ren
Carolin Rindfleisch
Daniel Ringwalt
David Rizo
Matthias Robine
Francisco Rodriguez Algarra
Marcelo Rodríguez López
Jim Rogers
Gerard Roma
Robert Rowe
Alan Said
Justin Salamon
Gabriel Sargent
Alexander Schindler
Jan Schlueter
Bjoern Schuller
Prem Seetharaman
Siddharth Sanjay Sigtia
Holly Silk
Diana Siwiak
Joren Six
Olga Slizovskaia
Lloyd Smith
Reinhard Sonnleitner
Pablo Sprechmann
Ajay Srinivasamurthy
Ryan Stables
Rebecca Stewart
Fabian-Robert Stoeter
Cameron Summers
Florian Thalmann
Mi Tian
Marko Tkalcic
George Tourtellot
Kosetsu Tsukuda
Andreu Vall
Rafael Valle
Marnix van Berchum
Bastiaan van der Weij
Leigh VanHandel
Gissel Velarde
Naresh Vempala
Gabriel Vigliensoni
Richard Vogl
Xing Wang
Hsin-Min Wang
Cheng-i Wang
Xinxi Wang
Ron Weiss
Christof Weiß
Ian Whalley
Daniel Wolff
Matthew Woolhouse
Chih-Wei Wu
Fu-Hai Frank Wu
Guangyu Xia
Mu-Heng Yang
Asteris Zacharakis
Eva Zangerle
Massimiliano Zanoni
Jose Zapata
Yichi Zhang
Cárthach Ó Nuanáin
Sertan Şentürk

Preface

It is our great pleasure to welcome you to the 17th Conference of the International Society for Music Information Retrieval (ISMIR 2016). The annual ISMIR conference is the world's leading research forum on processing, analyzing, searching, organizing, and accessing music-related data. This year's conference, which takes place in New York City, USA, from August 7 to 11, 2016, is organized by New York University and Columbia University.

The present volume contains the complete manuscripts of all the peer-reviewed papers presented at ISMIR 2016. A total of 287 submissions were received before the deadline, out of which 238 complete and well-formatted papers entered the review process. Special care was taken to assemble an experienced and interdisciplinary review panel including people from many different institutions worldwide. As in previous years, the reviews were double-blinded (i.e., both the authors and the reviewers were anonymous) with a two-tier review model involving a pool of 283 reviewers, including a program committee of 59 members. Each paper was assigned to a PC member and three reviewers. Reviewer assignments were based on topic preferences and PC member assignments. After the review phase, PC members and reviewers entered a discussion phase aiming to homogenize acceptance vs. rejection decisions.

In 2015, the size of the program committee was increased substantially, and we continued using this structure, with 59 program committee members this year. Taking care of four submissions on average, the PC members were asked to adopt an active role in the review process by conducting an intensive discussion phase with the other reviewers and by providing a detailed meta-review. Final acceptance decisions were based on 955 reviews and meta-reviews. From the 238 reviewed papers, 113 papers were accepted resulting in an acceptance rate of 47.5%. The table shown on the next page summarizes the ISMIR publication statistics over its history.

The mode of presentation of the papers was determined after the accept/reject decisions and has no relation to the quality of the papers or to the number of pages allotted in the proceedings. From the 113 contributions, 25 papers were chosen for oral presentation based on the topic and broad appeal of the work, whereas the other 88 were chosen for poster presentation. Oral presentations have a 20-minute slot (including setup and questions/answers from the audience) whereas poster presentations are done in two sessions per day for a total of 3 hours, the same posters being presented in the morning and in the afternoon of a given conference day.

The ISMIR 2016 conference runs for a 5-day period. The selected submissions are presented over a period of 3.5 days, preceded by a day of tutorials and followed by half a day of late-breaking/demo & unconference sessions. Moreover, this year the conference offers four satellite events before and after the main conference: The ISMIR Think Tank, Hacking on Music and Audio Research (HAMR), as well as workshops on Cognitively Based Music Informatics Research (CogMIR), and Digital Libraries for Musicology (DLfM). We believe this is an exciting and engaging program reflecting the breadth and depth of activities across our community.

Year	Location	Oral	Poster	Total Papers	Total Pages	Total Authors	Unique Authors	Pages/ Paper	Authors/ Paper	Unique Authors/ Paper
2000	Plymouth	19	16	35	155	68	63	4.4	1.9	1.8
2001	Indiana	25	16	41	222	100	86	5.4	2.4	2.1
2002	Paris	35	22	57	300	129	117	5.3	2.3	2.1
2003	Baltimore	26	24	50	209	132	111	4.2	2.6	2.2
2004	Barcelona	61	44	105	582	252	214	5.5	2.4	2
2005	London	57	57	114	697	316	233	6.1	2.8	2
2006	Victoria	59	36	95	397	246	198	4.2	2.6	2.1
2007	Vienna	62	65	127	486	361	267	3.8	2.8	2.1
2008	Philadelphia	24	105	105	630	296	253	6	2.8	2.4
2009	Kobe	38	85	123	729	375	292	5.9	3	2.4
2010	Utrecht	24	86	110	656	314	263	6	2.	2.4
2011	Miami	36	97	133	792	395	322	6	3	2.4
2012	Porto	36	65	101	606	324	264	6	3.2	2.6
2013	Curitiba	31	67	98	587	395	236	5.9	3	2.4
2014	Taipei	33	73	106	635	343	271	6	3.2	2.6
2015	Málaga	24	90	114	792	370	296	7	3.2	2.6
2016	**New York**	**25**	**88**	**113**	**781**	**341**	**270**	**6.9**	**3.0**	**2.4**

Tutorials

Six tutorials take place on Sunday, providing a good balance between culture and technology. Three 3-hour tutorials are presented in parallel on Sunday morning, and three in parallel on Sunday afternoon.

Morning sessions:

- Tutorial 1: Jazz Solo Analysis between Music Information Retrieval, Music Psychology, and Jazz Research (Jakob Abeßer, Klaus Frieler, Wolf-Georg Zaddach)
- Tutorial 2: Music Information Retrieval: Overview, Recent Developments and Future Challenges (Emilia Gómez, Markus Schedl, Xavier Serra, Xiao Hu)
- Tutorial 3: Why is Studio Production Interesting? (Emmanuel Deruty, François Pachet)

Afternoon sessions:

- Tutorial 4: Introduction to EEG Decoding for Music Information Retrieval Research (Sebastian Stober, Blair Kaneshiro)
- Tutorial 5: Natural Language Processing for MIR (Sergio Oramas, Luis Espinosa-Anke, Shuo Zhang, Horacio Saggion)
- Tutorial 6: Why Hip-Hop is Interesting (Jan Van Balen, Ethan Hein, Dan Brown)

Keynote Speakers and Industry Panel

We are honored to have two distinguished keynote speakers:

- Barbara Haws, Archivist and Historian, New York Philharmonic, presenting "The New York Philharmonic Leon Levy Digital Archives: An Integrated View of Performance Since 1842"
- Beth Logan, PhD, VP, Optimization, DataXu, presenting "Is Machine Learning Enough?"

And a panel discussion moderated by Eric Humphrey of Spotify on the future of the music industry --- spanning creation, distribution, and consumption --- and the role that intelligent information technologies will play in that world. The panel consists of four distinguished members of industry:

- Liv Buli -- Data Journalist, Pandora / NextBigSound
- Alex Kolundzija -- Head of Web Platforms, ROLI
- Jim Lucchese -- Head of Creator Business, Spotify
- Kristin Thomson -- Co-director of Artist Revenue Streams, Future of Music Coalition

Evaluation and MIREX

Evaluation remains an important issue for the community and its discussion will be, as in previous years, an integral part of ISMIR: MIREX-16 participants will present posters of their work during the Late Breaking Demo session, and we will feature a town-hall style, plenary discussion on the topic of evaluation. For the town hall, participants are invited to submit and vote on topics for discussion through an online forum, and discussion will be open to all in attendance.

Late-Breaking/Demo & Unconference

Thursday afternoon is dedicated to late-breaking papers and MIR system demonstrations. Abstracts for these presentations are available online. Moreover, as in previous years, we have a special "unconference" session in which participants break up into smaller groups to discuss MIR issues of particular interest. This is an informal and informative opportunity to get to know peers and colleagues from all around the world.

Satellite Events

ISMIR 2016 has expanded its offering of satellite events to four, emphasizing two very important themes: the bridge between academia and industry and diversity in our community.

The **ThinkTank** is a free event organized in partnership with RealIndustry for invited graduate students in MIR and Media Technology to provide opportunities to interact with industry. It aims to expose those students who are interested in a future career outside of academia to the challenges and concepts that arise in industrial and entrepreneurial settings.

Hacking on Audio and Music Research (HAMR) is an event series which applies the hackathon model to the development of new techniques for analyzing, processing, and synthesizing audio and music signals. This is in contrast to traditional hackathons and hack days, which generally emphasize prototyping commercial applications, but have proven to be an effective way for entrepreneurs and hobbyists to spend a concentrated period of time doing preliminary work on a new project.

The sixth annual **CogMIR (Cognitively based Music Informatics Research)** seminar will be collocated with ISMIR 2016. CogMIR aims to address the need for highlighting empirically derived methods based on human cognition that inform the field of music informatics and music information retrieval and covers topics such as music similarity, music emotion, music analysis and generation, music information retrieval, and computational modeling.

The **Digital Libraries for Musicology (DLfM)** workshop presents a venue specifically for those working on, and with, Digital Library systems and content in the domain of music and musicology. This includes bibliographic and metadata for music, intersections with music Linked Data, and the challenges of working with the multiple representations of music across large-scale digital collections such as the Internet Archive and HathiTrust.

WiMIR

Women in MIR (WiMIR) is a group of people in the MIR community dedicated to promoting the role of, and increasing opportunities for, women in the field. Participants meet to network, share information, and discuss in an informal setting the goal of building a community that supports women – and more broadly, diversity – in the field of MIR.

WiMIR has held annual meetings at the ISMIR conference since 2012, garnering a high turnout of both female and male attendees. For the first time in 2016, WiMIR has organized a mentoring program connecting female students, postdocs, and early-stage researchers to more senior females and male allies in the field, and has also received substantial financial support which enables more female researchers to attend the ISMIR conference.

Social Events

In addition to the academic focus of ISMIR, we have aimed to provide a number of unique social events. The social program provides participants with an opportunity to relax after meetings, to experience New York City, and to network with other ISMIR participants. The social program includes:

- Sunday, August 7, Welcome reception at Houston Hall, a massive beer hall in NYC's West Village specializing in local craft beers.
- Tuesday, August 9: ISMIR 2016 Pandora jam session at DROM, an eclectic and vibrant music venue in the East Village, well known for introducing new talent to NYC's live music scene.
- Wednesday August 10: Banquet dinner at Brooklyn Bowl, one of the hippest venues in the city, famous for live music, bowling lanes, fried chicken and a fun atmosphere.

Host city

New York is the largest city in the US, and one of the most iconic urban areas in the world. It boasts one of the most vibrant music scenes to be found anywhere in the world, as home to landmark institutions such as Carnegie Hall, the Blue Note, The New York Philharmonic, Broadway and Juilliard. It is the birthplace of styles as diverse as hiphop, salsa, bebop, disco and punk rock, and serves as the base of an impressive array of major artists, studios and record labels.

Recently it has experienced a boom in music technology activity and entrepreneurship including presence from companies such as Google/Songza, Spotify/Echonest, Pandora, Shazam, and a growing list of music technology startups; all of which complements the already extensive network of more traditional music businesses and services. It is centrally located in the northeastern United States, which includes a critical mass of MIR laboratories and researchers, both in industry and academia.

The conference venues are centrally located in the historic Greenwich Village neighborhood of lower Manhattan, which is rich with dining and entertainment options. No less than 700 restaurants and 400 bars are listed within a 15 minute walk from the venue and other local attractions including famous Jazz clubs such as Blue Note and the Village Vanguard.

Acknowledgments

We are very proud to present to you the proceedings of ISMIR 2016. The conference program was made possible thanks to the hard work of many people including the members of the organizing committee, the many reviewers and meta-reviewers from the program committee, as well as the leadership and administrative staff of Columbia University and NYU. Special thanks go to this year's partners:

- Diamond Partners:
  - Amazon Music
  - Bose Corporation
  - Gracenote
  - National Science Foundation, via award IIS-1646882
  - Pandora
  - Smule
  - Spotify, Ltd
  - Steinberg Media Technologies GmbH

- Platinum Partners:
  - Google, Inc
  - Harmonix Music Systems

- Gold Partners:
  - ACRCloud - Automatic Content Recognition Cloud Service
  - Native Instruments GmbH
  - Yandex

- Silver Partners:
  - Shazam

Last, but not least, the ISMIR program is possible only thanks to the excellent contributions of our community in response to our call for participation. The biggest acknowledgment goes to you, the authors, researchers and participants of this conference. We wish you a productive and memorable stay in New York City.

Johanna Devaney
Douglas Turnbull
George Tzanetakis
Program Chairs, ISMIR 2016

Juan Pablo Bello
Dan Ellis
General Chairs, ISMIR 2016

Contents

Keynote Talks

Keynote Talk 1

The New York Philharmonic Leon Levy Digital Archives: An Integrated View of Performance Since 1842

Barbara Haws
Archivist and Historian, New York Philharmonic

Abstract

For nearly 175 years the New York Philharmonic has been assiduously documenting and keeping all facets of its existence in whatever formats were available; marked conducting scores, orchestra parts, printed programs, business records, contracts, letters, newspaper reviews, audio and video. With the digitization of this material, it is possible for the first time to see the relationships between seemingly disparate materials that expand our understanding of the performance experience and the music itself. As well, especially through the GitHub repository, unanticipated applications of the data have occurred. Working towards the year 2066, the challenge is to incorporate the new formats of the modern era to ensure that the single longest running dataset of a performing institution continues.

Biography

Barbara Haws has been the Archivist and Historian of the New York Philharmonic since 1984. Haws has lectured extensively about the Philharmonic's past and curated major exhibitions here and in Europe. She is a Grammy nominated producer of the Philharmonic's Special Editions historic recordings. Haws along with Burton Bernstein is the author of Leonard Bernstein: American Original published by Harper Collins, 2008 and the essay "U.C. Hill, An American Musician Abroad (1835-37)". Since 2009, Haws has led an effort funded by the Leon Levy Foundation to digitize more than three million pages of archival material since 1842, making it freely available over the internet. The Digital Archives project was recently featured on FiveThirtyEight podcast "What's The Point".

Keynote Talk 2

Is Machine Learning Enough?

Beth Logan, PhD
VP, Optimization, DataXu

Abstract

We live in a world unimaginable even 20 years ago. From our phones we can summon a car, book a vacation and of course access the word's music to find just those songs we like, even songs we didn't know we liked. Automated Machine Learning at scale enables these and thousands more applications. But in business, it's good to know when to stop. It's not always smart to automate that last, difficult 10% of performance. Indeed in music and other industries, humans often curate the results. Will humans always be in the loop or will the machines eventually take over? Will Machine Learning ever be enough?

Biography

Beth is the VP of Optimization at DataXu, a leader in programmatic marketing. She has made contributions to a wide variety of fields, including speech recognition, music indexing and in-home activity monitoring. Beth holds a PhD in speech recognition from the University of Cambridge.

Tutorials

Tutorial 1

Jazz solo analysis between music information retrieval, music psychology, and jazz research

Jakob Abeßer, Klaus Frieler, and Wolf-Georg Zaddach

Abstract

The tutorial will consist of four parts. The first part sets the scene with a short run through jazz history and its main styles and an overview of the central musical elements in jazz. Out of this introduction, the main research questions will be derived, which cover jazz research and jazz historiography, analysis of musical style, performance research, and psychology of creative processes. Particularly, research issues as addressed in the Jazzomat Research Project will be discussed and results will be presented.

The second part of the tutorial will deal with the details of the Weimar Jazz Database. Each solo encompasses pitch, onset, and offset time of the played tones as well as several additional annotations, e.g., manually tapped beat grids, chords, enumerations of sections and choruses as well as phrase segmentations. The process of creating the monophonic transcriptions will be explained (solo selection, transcription procedure, and quality management) and examples will be shown. Moreover, the underlying data-model of the Weimar Jazz Database, which includes symbolic data as well as data automatically extracted from the audio-files (e.g., intensity curves and beat-wise bass chroma) will be discussed in detail.

In the third part, we will introduce our analysis tools MeloSpySuite and MeloSpyGUI, which allow the computation of a large set of currently over 500 symbolic features from monophonic melodies. These features quantify various tonal, harmonic, rhythmic, metrical, and structural properties of the solo melodies. In addition, pattern retrieval modules, based on n-gram representations and a two-stage search option using regular expressions, play an integral part for the extraction of motivic cells and formulas from jazz solos. All tools can be readily applied to other melodic datasets besides the Weimar Jazz Database. Several use cases will be demonstrated and research results will be discussed.

The final part of the tutorial will focus on audio-based analysis of recorded jazz solo performances. We follow a score-informed analysis approach by using the solo transcriptions from the Weimar Jazz Database as prior information. This allows us to mitigate common problems in transcribing and analyzing polyphonic and multi-timbral audio such as overlapping instrument partials. A score-informed source separation algorithm is used to split the original recordings into a solo and an accompaniment track, which allows the tracking of f0-curves and intensity contours of the solo instrument. We will present the results of different analyses of the stylistic idiosyncrasies across well-known saxophone and trumpet players. Finally, we will briefly outline further potentials and challenges of score-informed MIR techniques.

Jakob Abeßer holds a degree in computer engineering (Dipl.-Ing.) from Ilmenau University of Technology. He is a postdoctoral researcher in the Semantic Music Technologies group at Fraunhofer IDMT and obtained a PhD degree (Dr.-Ing.) in Media Technology from Ilmenau University of Technology in 2014. During his PhD, he was a visiting researcher at the Finnish Centre of Excellence in Interdisciplinary Music Research, University of Jyväskylä, Finland in 2010. As a research scientist at Fraunhofer, he has experience with algorithm development in the fields of automatic music transcription, symbolic music analysis, machine learning, and music instrument recognition. Also, he works as a postdoctoral researcher at the University of Music in Weimar in the Jazzomat Research Project, focusing on analyzing jazz solo recordings using Music Information Retrieval technologies.

Klaus Frieler graduated in theoretical physics (diploma) and received a PhD in systematic musicology in 2008. In between, he worked several years as a freelance software developer before taking up a post as lecturer in systematic musicology at the University of Hamburg in 2008. In 2012, he had a short stint at the C4DM, Queen Mary University of London. Since the end of 2012, he is a post-doctoral researcher with the Jazzomat Research Project at the University of Music "Franz Liszt" Weimar. His main research interests are computational and statistical music psychology with a focus on creativity, melody perception, singing intonation, and jazz research. Since 2006, he also works as an independent music expert specializing in copyright cases.

Wolf-Georg Zaddach studied musicology, arts administration, and history in Weimar and Jena, music management and jazz guitar in Prague, Czech Republic. After finishing his Magister Artium with a thesis about jazz in Czechoslovakia in the 50s and 60s, he worked as assistant professor at the department of musicology in Weimar. Since 10/2012 he works at the jazz research project of Prof. Dr. Martin Pfleiderer. Since 02/2014, he holds a scholarship by the German National Academic Foundation (Studienstiftung des deutschen Volkes) for his Ph.D. about heavy and extreme metal in the 1980s GDR/East Germany. He frequently performs live and on records as a guitarist.

Tutorial 2

Music Information Retrieval: Overview, Recent Developments and Future Challenges

Emilia Gómez, Markus Schedl, Xavier Serra, and Xiao Hu

Abstract

This tutorial provides a survey of the field of Music Information Retrieval (MIR), that aims, among other things, at automatically extracting semantically meaningful information from various representations of music entities, such as audio, scores, lyrics, web pages or microblogs. The tutorial is designed for students, engineers, researchers, and data scientists who are new to MIR and want to get introduced to the field.

The tutorial will cover some of the main tasks in MIR, such as music identification, transcription, search by similarity, genre/mood/artist classification, query by humming, music recommendation, and playlist generation. We will review approaches based on content-based and context-based music description and show how MIR tasks are addressed from a user-centered and multicultural perspective. The tutorial will focus on latest developments and current challenges in the field.

Emilia Gómez (emiliagomez.wordpress.com) is an Associate Professor (Serra-Hunter Fellow) at the Music Technology Group, Department of Information and Communication Technologies, Universitat Pompeu Fabra in Barcelona, Spain. She graduated as a Telecommunication Engineer at Universidad de Sevilla (Spain) and she studied classical piano performance at Seville Conservatoire of Music. She then received a DEA in Acoustics, Signal Processing and Computer Science applied to Music (ATIAM) at IRCAM, Paris (France) and a Ph.D. in Computer Science and Digital Communication at the UPF (awarded by EPSON foundation). Her research is within the Music Information Retrieval (MIR) field. She tries to understand and enhance music listening experiences by automatically extracting descriptors from music signals. She has designed algorithms able to describe music signals in terms of melody, tonality, and, by incorporating machine learning techniques, she has been able to model high-level concepts such as similarity, style or emotion. Emilia Gómez has co-authored more than a 100 publications in peer-reviewed scientific journals and conferences. She has contributed to more than 15 research projects, most of them funded by the European Commission and Spanish Government. She is elected member-at-large of the International Society for Music Information Retrieval (ISMIR). At the moment, she contributes to the COFLA project on computational analysis of flamenco music and she is the principal investigator for the European research project PHENICX, trying to innovate the way we experience classical music concerts.

Markus Schedl is an Associate Professor at the Johannes Kepler University Linz / Department of Computational Perception. He graduated in Computer Science from the Vienna University of Technology and earned his Ph.D. in Technical Sciences from the Johannes Kepler University Linz. Markus further studied International Business Administration at the Vienna University of Economics and Business Administration as well as at the Handelshogskolan of the University of Gothenburg, which led to a Master's degree. Markus (co-)authored more than 120 refereed conference papers and journal articles (among others, published in ACM Multimedia, SIGIR, ECIR, IEEE Visualization; Journal of Machine Learning Research, ACM Transactions on Information Systems, Springer Information Retrieval, IEEE Multimedia). Furthermore, he is associate editor of the Springer International Journal of Multimedia Information Retrieval and serves on various program committees and reviewed submissions to several conferences and journals (among others, ACM Multimedia, ECIR, IJCAI, ICASSP, IEEE Visualization; IEEE Transactions of Multimedia, Elsevier Data & Knowledge Engineering, ACM Transactions on Intelligent Systems and Technology, Springer Multimedia Systems). His main research interests include web and social media mining, information retrieval, multimedia, and music information research. Since 2007, Markus has been giving several lectures, among others, "Music Information Retrieval", "Exploratory Data Analysis", "Multimedia Search and Retrieval", "Learning from User-generated Data", "Multimedia Data Mining", and "Intelligent Systems". He further spent several guest lecturing stays at the Universitat Pompeu Fabra, Barcelona, Spain, the Utrecht University, the Netherlands, the Queen Mary, University of London, UK, and the Kungliga Tekniska Hgskolan, Stockholm, Sweden.

Xavier Serra is Associate Professor of the Department of Information and Communication Technologies and Director of the Music Technology Group at the Universitat Pompeu Fabra in Barcelona. After a multidisciplinary academic education he obtained a PhD in Computer Music from Stanford University in 1989 with a dissertation on the spectral processing of musical sounds that is considered a key reference in the field. His research interests cover the analysis, description and synthesis of sound and music signals, with a balance between basic and applied research and approaches from both scientific/technological and humanistic/artistic disciplines. Dr. Serra is very active in promoting initiatives in the field of Sound and Music Computing at the local and international levels, being involved in the editorial board of a number of journals and conferences and giving lectures on current and future challenges of the field. He has recently been awarded an Advanced Grant of the European Research Council to carry out the project CompMusic aimed at promoting multicultural approaches in music computing research.

Xiao Hu is an Assistant Professor in the Division of Information and Technology Studies in the Faculty of Education of the University of Hong Kong. She received her Ph.D degree in Library and Information Science from the University of Illinois, with an award winning dissertation on multimodal music mood classification. Dr. Hu's research interests include music mood recognition, MIR evaluation, user-centered MIR studies and cross-cultural MIR. Dr. Hu has won the Best Student Paper award in the ACM Joint Conference on Digital Libraries (2010) and Best Student Paper award in the iConference (2010). Dr. Hu has been a visiting scholar at the National Institute of Informatics, Japan. She was a tutorial speaker on music affect recognition (2012) and a Conference Co-chair (2014) in the International Society for Music Information Retrieval Conference.

Tutorial 3

Why is studio production interesting?

Emmanuel Deruty and François Pachet

Abstract

The tutorial follows the "Why X is interesting" series that aims at bridging the gap between technology-oriented and music-related research. It will suggest a number of reasons why production is important for MIR, seen from the eyes of an expert (the first author) and a MIR researcher (the second one).

In music, the use of studio techniques has become commonplace with the advent of cheap personal computers and powerful DAWs. The MIR community has long been confronted to studio production. Since ISMIR's first installment in 2000, about 35 papers involving more than 70 authors have addressed studio production. However, more than 44% of these identify studio production as a source of problems: the so-called album or producer effect gets in the way of artist or genre identification, audio processing techniques prevent proper onset detections, or effects are responsible for false positive in singing voice detection. A few of these papers even characterize production as not being part of the music. On the other hand, another 35% of these papers either outline the knowledge of studio production as useful or indispensable to MIR tasks, or try to provide a formal characterization for this set of techniques.

A difficulty met by MIR researchers interested in studio production techniques is that production specialists are reluctant to formalize their knowledge. Like old-fashioned guild artisans, engineers and producers reputedly learn "tricks of the trade" from the "Greatest Teachers" or "mentors". As a result, knowledge of studio production techniques is not widespread in the scientific community. Even in the upcoming field of automatic mixing, a domain intrinsically related to studio production, we have found that only 15% of scientific papers take these techniques into account. A similar issue can be observed at DAFx, where papers dealing with studio production as actually performed in the music community are rare.

The tutorial aims at explaining studio production techniques to MIR researchers in a simple and practical way, in order to highlight the main production tricks and usages to a MIR audience.

We will review standard aspects of studio music production, including recording, processing, mixing, and mastering. We will then focus on the basic methods of audio processing: EQs, compression, reverbs, and such. We will illustrate how these basic techniques can be combined creatively.

Production techniques may be implemented in a variety of ways, depending on trends and available hardware. We'll go through a brief retrospective of how these techniques have been

used since the mid 60's in different ways. As different variations of the same basic processes can influence, sometimes drastically, the finished product, we believe such knowledge may be useful in relation to classification and similarity.

MIR researchers often conceptualize lead vocals as a solo line, possibly ornamented with backing vocals and audio effects. In practice, we'll show that vocal track production in mainstream pop music results in complex architectures. We will analyze the vocals tracks in some mainstream hits. It will demonstrate that studio production is an integral part of the music, not an extraneous layer of effects. Consequences are obvious for the nature of the information MIR scholars look for, e.g. in information extraction, retrieval, similarity or recommendation.

We will complete the tutorial with a short demo of automatic mixing techniques developed in our lab that use auto-adaptive audio effects. It will demonstrate that consideration of production is a definite advantage in music generation.

Emmanuel Deruty studied studio production for music at the Conservatoire de Paris (CNSMDP), where he graduated as Tonmeister in 2000. He has worked in many fields related to audio and music production in Europe and in the US: sound designer in a research context (IRCAM), sound designer in a commercial context (Soundwalk collective), film music producer and composer (Autour de Minuit & Everybody on Deck, Paris), lecturer at Alchemea College of sound engineering (London), writer for the Sound on Sound magazine (Cambridge, UK). He's worked as a M.I.R. researcher at IRCAM, INRIA and Akoustic-Arts, France. He's currently working on automatic mixing at Sony CSL, and is a specialist of the "loudness war".

François Pachet received his Ph.D. and Habilitation degrees from Paris 6 University (UPMC). He is a Civil Engineer and was Assistant Professor in Artificial Intelligence and Computer Science, at Paris 6 University, until 1997. He is now director of the Sony Computer Science Laboratory in Paris, where he conducts research in interactive music listening and performance and musical metadata and developed several innovative technologies and award winning systems. François Pachet has published intensively in artificial intelligence and computer music. He was co-chair of the IJCAI 2015 special track on Artificial Intelligence and the Arts, and has been elected ECCAI Fellow in 2014. His current goal, funded by an ERC Advanced Grant, is to build computational representations of style from text and music corpora,that can be exploited for personalized content generation. He is also an accomplished musician (guitar, composition) and has published two music albums (in jazz and pop) as composer and performer.

Tutorial 4

Introduction to EEG Decoding for Music Information Retrieval Research

Sebastian Stober and Blair Kaneshiro

Abstract

Perceptual and cognitive approaches to MIR research have very recently expanded to the realm of neuroscience, with MIR groups beginning to conduct neuroscience research and vice versa. First publications have already reached ISMIR and for the very first time, there will be a dedicated satellite event on cognitively based music informatics research (CogMIR) at this year's ISMIR conference. Within the context of this growing potential for cross-disciplinary collaborations, this tutorial will provide fundamental knowledge of neuroscientific approaches and findings with the goal of sparking the interest of MIR researchers and leading to future intersections between these two exciting fields. Specifically, our focus for this tutorial is on electroencephalography (EEG), a widely used and relatively inexpensive recording modality which offers high temporal resolution, portability, and mobility – characteristics that may prove especially attractive for applications in MIR. Attendees of this tutorial will gain a fundamental understanding of EEG responses, including how the data are recorded as well as standard procedures for preprocessing and cleaning the data. Keeping in mind the interests and objectives of the MIR community, we will highlight EEG analysis approaches, including single-trial analyses, that lend themselves to retrieval scenarios. Finally, we will review relevant open-source software, tools, and datasets for facilitating future research. The tutorial will be structured as a combination of informational slides and live-coding analysis demonstrations, with ample time for Q&A with the audience.

Sebastian Stober is head of the recently established junior research group on "Machine Learning in Cognitive Science" within the inter-disciplinary setting of the Research Focus Cognitive Science at the University of Potsdam. Before, he was a post-doctoral fellow at the Brain and Mind Institute of the University of Western Ontario where he investigated ways to identify perceived and imagined music pieces from electroencephalography (EEG) recordings. He studied computer science with focus on intelligent systems and music information retrieval at the Otto-von-Guericke University Magdeburg where he received his diploma degree in 2005 and his Ph.D. in 2011 for his thesis on adaptive methods for user-centered organization of music collections. He has also been co-organizer for the International Workshops on Learning Semantics of Audio Signals (LSAS) and Adaptive Multimedia Retrieval (AMR). With his current research on music imagery information retrieval, he combines music information retrieval with cognitive neuroscience.

Blair Kaneshiro is a Ph.D. candidate (ABD) at the Center for Computer Research in Music and Acoustics at Stanford University. She earned her B.A. in Music, M.A. in Music, Science,

and Technology, and M.S. in Electrical Engineering, all from Stanford. Her research explores musical engagement and expectation through brain responses, with an emphasis on multivariate and single-trial approaches to EEG analysis. Other research interests include the study of musical engagement using behavioral and large-scale social-media data, and promotion of reproducible and cross-disciplinary research through open-source tools and datasets. She is affiliated with CCRMA's Music Engagement Research Initiative (MERI) led by professor Jonathan Berger; music tech company Shazam; and the Brain Research group (formerly Suppes Brain Lab) at Stanford's Center for the Study of Language and Information.

Tutorial 5

Natural Language Processing for MIR

Sergio Oramas, Luis Espinosa-Anke, Shuo Zhang, and Horacio Saggion

Abstract

An increasing amount of musical information is being published daily in media like Social Networks, Digital Libraries or Web Pages. All this data has the potential to impact in musicological studies, as well as tasks within MIR such as music recommendation. Making sense of it is a very challenging task, and so this tutorial aims to provide the audience with potential applications of Natural Language Processing (NLP) to MIR and Computational Musicology.

In this tutorial, we will focus on linguistic, semantic and statistical-based approaches to extract and formalize knowledge about music from naturally occurring text. We propose to provide the audience with a preliminary introduction to NLP, covering its main tasks along with the stateoftheart and most recent developments. In addition, we will showcase the main challenges that the music domain poses to the different NLP tasks, and the already developed methodologies for leveraging them in MIR and musicological applications. We will cover the following NLP tasks:

- Basic text preprocessing and normalization
- Linguistic enrichment in the form of part-of-speech tagging, as well as shallow and dependency parsing.
- Information Extraction, with special focus on Entity Linking and Relation Extraction.
- Text Mining
- Topic Modeling
- Sentiment Analysis
- Word Vector Embeddings

We will also introduce some of the most popular python libraries for NLP (e.g. Gensim, Spacy) and useful lexical resources (e.g. WordNet, BabelNet). At the same time, the tutorial analyzes the challenges and opportunities that the application of these techniques to large amounts of texts presents to MIR researchers and musicologists, presents some research contributions and provides a forum to discuss about how address those challenges in future research. We envisage this tutorial as a highly interactive session, with a sizable amount of hands-on activities and live demos of actual systems.

Sergio Oramas received a degree in Computer Engineering by the Technical University of Madrid in 2004, and a B.A. in Musicology by the University of La Rioja in 2011. He is a PhD candidate at the Music Technology Group (Pompeu Fabra University) since 2013, holding a “La Caixa” PhD Fellowship. His research interests are focused on the extraction of

structured knowledge from text and its application in Music Information Retrieval and Computational Musicology.

Luis Espinosa-Anke is a PhD candidate at the Natural Language Processing group in at Pompeu Fabra University. His research focuses in learning knowledge representations of language, including automatic construction of glossaries; knowledge base generation, population and unification; and automatic taxonomy learning. He is Fulbright alumni, "laCaixa" scholar, and member of the Erasmus Mundus Association as well as the European Network of eLexicography.

Shuo Zhang is a PhD candidate in Computational Linguistics at Georgetown University, USA, and a collaborator/researcher at the Music Technology Group, Universitat Pompeu Fabra. He has worked in both text (NLP-information extraction) and sound (speech processing, timeseries data mining in speech prosody) aspects of computational linguistics and their applications in MIR. His past and current projects include areas such as coreference resolution, search and visualization of multilayered linguistic corpora, text mining & topic modeling in MIR, temporal semantics, timeseries mining in speech and music, etc. Shuo holds B.Sci. from the Peking University, M.A. from the Department of Music, University of Pittsburgh, and M.Sci. in Computational Linguistics from Georgetown University.

Horacio Saggion is Profesor Agregado at the Department of Technologies, Universitat Pompeu Fabra. He holds a PhD in Computer Science from Université de Montréal (Canada). He is associated to the Natural Language Processing group where he works on automatic text summarization, text simplification, information extraction, text processing in social media, sentiment analysis and related topics. His research is empirical combining symbolic, pattern-based approaches and statistical and machine learning techniques. Before joining Universitat Pompeu Fabra, he worked at the University of Sheffield for a number of UK and European research projects developing competitive human language technology. He was also an invited researcher at Johns Hopkins University in 2011. Horacio has published over 100 works in leading scientific journals, conferences, and books in the field of human language technology.

Tutorial 6

Why Hip-Hop is interesting

Jan Van Balen, Ethan Hein, and Dan Brown

Abstract

Hip-hop, as a musical culture, is extremely popular around the world. Its influence on popular music is unprecedented: the hip-hop creative process has become the dominant practice of the pop mainstream, and spawned a range of electronic styles. Strangely, Hip-hop hasn't been the subject of much research in Music Information Retrieval. Music research rooted in the European music traditions tends to look at music in terms harmony, melody and form. Even compared to other popular music, these are facets of music that don't quite work the same way in Hip-Hop as they do in the music of Beethoven and the Beatles.

In this tutorial, we will argue that a different perspective may be needed to approach the analysis of Hip-hop and popular music computationally—an analysis with a stronger focus on timbre, rhythm and lyrics and attention to groove, texture, rhyme, metadata and semiotics.

Hip-hop is often said to integrate four modes of artistic expression: Rap, turntablism, breakdance and graffiti culture. In the first part of this tutorial, we will discuss the emergence and evolution of Hip-Hop as a musical genre, and its particularities, focusing our discussion on beat-making (turntablism and sampling), and Rap. A second part of the talk will highlight the most important reasons why MIR practitioners and other music technologists should care about Hip-Hop, talking about Hip-hop's relevance today, and the role Hip-hop in popular music. We highlight its relative absence in music theory, music education, and MIR. The latter will be illustrated with examples of MIR and digital musicology studies in which Hip-hop music is explicitly ignored or found to 'break' the methodology.

Next, we review some of the work done on the intersection of Hip-hop and music technology. Because the amount of computational research on Hip-hop music is rather modest, we discuss, in depth, three clusters of research on the intersection of Hip-hop and music technology. The first two are related of MIR, focusing on 'beats' and sampling, and Rap lyrics and rhyme. For the third topic, situating Hip-hop in the broader topic of music technology, we look at how an important role for both music technology and Hip-hop is emerging in music education. Finally, the last part of the talk will give an overview of resources and research opportunities for Hip-hop research.

This tutorial is aimed at anyone interested in Music Information Retrieval and popular music, whether familiar with Hip-hop music or not. We also aim to make it relevant to a broader audience interested in music technology, touching on topics like sampling and samplers, and Hip-hop in technology and music education, and to anyone interested in text processing, with an additional focus on the analysis of lyrics.

Throughout the tutorial, we intend to include a lot of listening examples. Participants will access to an extended playlist of selected listening examples, along with a short guide to the significance of the selected recordings.

Jan Van Balen researches the use of audio MIR methods to learn new things about music and music memory. Living in London, he is finishing his PhD with Utrecht University (NL), on audio corpus analysis and popular music. As part of his PhD project and with colleagues at Utrecht University and University of Amsterdam, he worked on Hooked, a game to collect data on popular music memory and 'hooks'. Other work has focused on the analysis of the game's audio and participant data, and on content ID techniques for the analysis of samples, cover songs and folk tunes.

Ethan Hein is a doctoral student in music education at New York University. He teaches music technology, production and education at NYU and Montclair State University. As the Experience Designer In Residence with the NYU Music Experience Design Lab, Ethan has taken a leadership role in a range of technologies for learning and expression. In collaboration with Soundfly, he recently launched an online course called Theory For Producers. He maintains a widely-followed and influential blog at http://www.ethanhein.com, and has written for various publications, including Slate, Quartz, and NewMusicBox.

Dan Brown is Associate Professor of Computer Science at the University of Waterloo, where he has been a faculty member since 2000. Dan earned his Bachelor's degree in Math with Computer Science from MIT, and his PhD in Computer Science from Cornell. Before coming to Waterloo, he spent a year working on the Human and Mouse Genome Projects as a post-doc at the MIT Center for Genome Research. Dan's primary research interests are designing algorithms for understanding biological evolution, and applying bioinformatics ideas to problems in music information retrieval.

Oral Session 1

Mixed

SIMPLE: ASSESSING MUSIC SIMILARITY USING SUBSEQUENCES JOINS

Diego F. Silva[1,2], Chin-Chia M. Yeh[2],Gustavo E. A. P. A. Batista[1], Eamonn Keogh[2]
[1] Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, Brazil
[2] Department of Computer Science and Engineering, University of California, Riverside, USA
diegofsilva@icmc.usp.br,myeh003@ucr.edu,gbatista@icmc.usp.br,eamonn@ucr.edu

ABSTRACT

Most algorithms for music information retrieval are based on the analysis of the similarity between feature sets extracted from the raw audio. A common approach to assessing similarities within or between recordings is by creating similarity matrices. However, this approach requires quadratic space for each comparison and typically requires a costly post-processing of the matrix. In this work, we propose a simple and efficient representation based on a subsequence similarity join, which may be used in several music information retrieval tasks. We apply our method to the cover song recognition problem and demonstrate that it is superior to state-of-the-art algorithms. In addition, we demonstrate how the proposed representation can be exploited for multiple applications in music processing.

1. INTRODUCTION

With the growing interest in applications related to music processing, the area of music information retrieval (MIR) has attracted huge attention in both academia and industry. However, the analysis of audio recordings remains a significant challenge. Most algorithms for content-based music retrieval have at their cores some similarity or distance function. For this reason, a wide range of applications rely on some technique to assess the similarity between music objects. Such applications include segmentation [8], audio-to-score alignment [4], cover song recognition [15], and visualization [23].

A common approach to assessing similarity in music recordings is achieved by utilizing a self-similarity matrix (SSM) [5]. This representation reveals the relationship between each "snippet" of a track to all the other segments in the same recording. This idea has been generalized to measure the relationships between subsequences of *different* songs, as in the application of cross-recurrence analysis for cover song recognition [16].

The main advantage of similarity matrices is the fact that they simultaneously reveal both the *global* and the *local* structure of music recordings. However, this representation requires quadratic space in relation to the length of the feature vector used to describe the audio. For this reason, most methods to find patterns in the similarity matrix are (at least) quadratic in time complexity. In spite of this, most information contained in similarity matrices is irrelevant or has little impact in its analysis. This observation suggests the need for a more space and time efficient representation of music recordings.

In this work, we extend the *subsequences all-pairs-similarity-search*, also known as *similarity join*, in order to assess the similarity between audio recordings for MIR tasks. As with the common similarity matrices, representing the entire subsequence join requires a quadratic space, and also has a high time complexity, which is dependent on the length of the subsequences to be joined.

However, in this work we show that we can exploit a new data structure called *matrix profile* which allows a space efficient representation of the similarity join matrix between subsequences. Moreover, we can leverage recent optimizations in FFT-based all-neighbor search that allow the matrix profile to be computed efficiently [10]. For clarity, we refer to the representation presented in this paper as Similarity Matrix ProfiLE (SiMPle).

Figure 1 illustrates an example of two matrices representing the dissimilarities within and between recordings and their relative SiMPle, which correspond to the minimum value of each column of the similarity matrices.

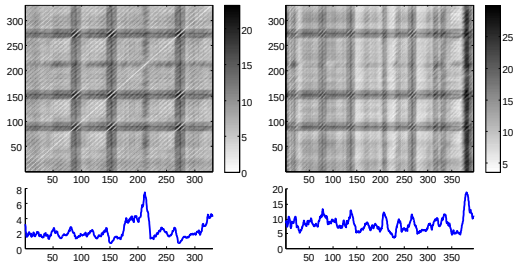


Figure 1. Similarity matrix within (*left*) and between different songs (*right*) and their respective SiMPle.

In summary, our method has the following advantages/features:

- It is a novel approach to assess the audio similarity and can be used in several MIR algorithms;
- We exploit the fastest known subsequence similarity search technique in the literature [10], which makes our method fast and exact;
- It is simple and only requires a single parameter, which is intuitive to set for MIR applications;
- It is space efficient, requiring the storage of only $O(n)$ values;
- Once we calculate the similarity profile for a dataset it can be efficiently updated, which has implications for streaming audio processing.

© Diego F. Silva, Chin-Chia M. Yeh, Gustavo E. A. P. A. Batista, Eamonn Keogh. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Diego F. Silva, Chin-Chia M. Yeh, Gustavo E. A. P. A. Batista, Eamonn Keogh. "SiMPle: Assessing Music Similarity Using Subsequences Joins", 16th International Society for Music Information Retrieval Conference, 2016.

2. SiMPle: SIMILARITY MATRIX PROFILE

We begin by describing the operation for producing the matrix profile, a *similarity join*. For clarity, we use the term *time series* to refer to the ordered set of features that describe a whole recording and *subsequence* to define any continuous subset of features from the time series.

Definition 1: *Similarity join*: given two time series *A* and *B* with the desired subsequence length *m*, the similarity join identifies the nearest neighbor of each subsequence (with length *m*) in *A* from all the possible subsequence set of *B*.

Through such a similarity join, we can gather two pieces of information about each subsequence in *A*, which are: 1) the Euclidean distance to its nearest neighbor in *B* and 2) the position of its nearest neighbor in *B*. Such information can be compactly stored in vectors, referred as *similarity matrix profile* (SiMPle) and similarity *matrix profile index* (SiMPle index) respectively.

One special case of similarity join is when both input time series refer to the same recording. We define the operation that handles this specific case *self-similarity join*.

Definition 2: *Self-similarity join*: given a time series *A* with the desired subsequence length *m*, the self-similarity join identifies the non-trivial nearest neighbor of each subsequence (with length *m*) in *A* from all the possible subsequence set of *A*.

The only major difference between self-similarity join (**Definition 2**) and similarity join (**Definition 1**) is the exclusion of trivial matched pairs when identifying the nearest neighbor. The exclusion of trivial matches is crucial as matching a subsequence with itself (or slightly shifted version of itself) produces no useful information.

We describe our method to calculate SiMPle in Algorithm 1. In line 1, we record the length of *B*. In line 2, we allocate memory and initialize SiMPle P_{AB} and SiMPle index I_{AB}. From line 3 to line 6, we calculate the *distance profile* vector *D* which contains the distances between a given subsequence in time series *B* and each subsequence in time series *A*. The particular function we used to compute *D* is *MASS* (Mueen's Algorithm for Similarity Search), which is the most efficient algorithm known for distance vector computation [10]. We then perform the pairwise minimum for each element in *D* with the paired element in P_{AB} (i.e., min($D[i]$, $P_{AB}[i]$) for $i = 0$ to length(D) - 1.) We also update $I_{AB}[i]$ with *idx* when $D[i] \leq P_{AB}[i]$ as we perform the pairwise minimum operation. Finally, we return the result P_{AB} and I_{AB} in line 7.

Algorithm 1. Procedure to calculate SiMPle and SiMPle index

Input: Two user provided time series, *A* and *B*, and the desired subsequence length *m*
Output: The SiMPle P_{AB} and the associated SiMPle index I_{AB}

```
1  n_B ← Length(B)
2  P_AB ← infs, I_AB ← zeros, idxes ← 1:n_B-m+1
3  for each idx in idxes
4        D ← MASS(B[idx:idx+m-1], T_A) // c.f. [10]
5        P_AB, I_AB ← ElementWiseMin(P_AB, I_AB, D, idx)
6  end for
7  return P_AB, I_AB
```

Note that the Algorithm 1 computes SiMPle for the general similarity join. To modify it to compute the self-similarity join SiMPle of a time series *A*, we simply replace *B* by *A* in lines 1 and 4 and ignore trivial matches in *D* when performing *ElementWiseMin* in line 5.

The method *MASS* (used in line 4) is important to speed-up the similarity calculations. This algorithm has a time complexity of $O(n \log n)$. For brevity, we refer the reader interested in details of this method to [10].

In this work, we focus on demonstrating the utility of SiMPle on the cover song recognition task. Given that the cover song recognition is a specialization of the "query-by-similarity" task, we believe that it is the best scenario to evaluate a similarity method. Specifically, we propose a SiMPle-based distance measure between a query and its potential original version.

3. COVER SONG RECOGNITION

"Cover song" is the generic term used to denote a new performance of a previously recorded track. For example, a cover song may refer to a live performance, a remix or an interpretation in a different music style. The automatic identification of covers has several applications, such as copyright management, collection organization, and search by content.

In order to identify different versions of the same song, most algorithms search for globally [20] or locally [15][18] conserved structure(s). A well-known and widely applied algorithm for measuring the global similarity between tracks is Dynamic Time Warping (DTW) [11]. In spite of its utility in other domains, DTW is not generally robust to differences in structure between the recordings. A potential solution would be segmenting the song before applying the DTW similarity estimation. However, audio segmentation itself is also an open problem, and the error on boundaries detection can cause a domino effect (compounded errors) in the whole identification process.

In addition, the complexity of the algorithm to calculate DTW is $O(n^2)$. Although methods to fast approximate the DTW have been proposed [13], there is no error bound for such approximations. In other words, it is not possible to set a maximum error in the value obtained by it in relation to the actual DTW.

Algorithms that search for local similarities have been successfully used to provide structural invariance to the cover song identification task. A widely used method for music similarity proposes the use of a binary distance function to compare chroma-based features followed by a dynamic programming local alignment [15]. Despite its demonstrated utility to recognize cover recordings, this method has several parameters, that are unintuitive to tune, and is slow. Specifically, the local alignment is estimated by an algorithm with similar complexity to DTW. Plus, the binary distance between chroma features used in each step of the algorithm relies on multiple shifts of the chroma vectors under comparison.

3.1 SiMPle-Based Cover Song Recognition

In this work, we propose to use SiMPle to measure the distance between recordings in order to identify cover songs. In essence we exploit the fact that the *global* relation between the tracks is composed of many *local* simi-

larities. In this way, we are able to simultaneously take advantage of both local and global pattern matching.

Intuitively, we should expect that the SiMPle obtained by comparing a cover song to its original version is composed mostly of low values. In contrast, two completely different songs will result in a SiMPle constituted mainly by high values. For this reason, we adopted the median value of the SiMPle as a global distance estimation. Formally, the distance between a query *B* and a candidate original recording *A* is defined in Equation 1.

$$dist(A,B)=median(SiMPle(B,A)) \quad (1)$$

Note that several other measures of statistics could be used instead of the median. However, the median is robust to outliers in the matrix profile. Such distortions may appear when a performer decides, for instance, to add a new segment (e.g., an *improvisation* or *drum solo*) to the song. The robustness of our method to this situation, as well as other changes in structure, is discussed in the next section.

3.2 On the Structural Invariance

The structural variance is a critical concern when comparing different songs. Changes in structure may occur by insertion or deletion of segments, as well as changes in the order that different excerpts are played. From a high-level point of view, SiMPle describes a global similarity outline between songs by providing information of local comparisons. This fact has several implications in our distance estimation, which makes it largely invariant to structural variations:

- If two performances are virtually identical, except for the order and the number of repetitions of each representative excerpt (i.e., chorus, verse, bridge, etc.), all the values that compose SiMPle are close to zero;
- If a segment of the original version is *deleted* in the cover song, this will cause virtually no changes in the SiMPle;
- If a new feature is *inserted* into a cover, this will have as consequence a peak in the SiMPle that will cause only a slight increase in its median value.

4. EXPERIMENTAL EVALUATION

The evaluation of different choices of features sets is not the main focus of this paper. For this reason, we fix the use of chroma-based features in our experiments, as it is the most popular feature set to analyze music data. In order to provide local tempo invariance, we used the chroma energy normalized statistics (CENS) [12]. Specifically, for the cover song recognition task, we adopted the rate of two CENS per second of audio.

In addition, we preprocessed the feature sets in each comparison to provide key invariance. Before calculating the similarity between songs, we transpose one of them in order to have the same key using the optimal transposition index (OTI) [14].

We notice that we are committed to the reproducibility of our results, and we encourage researchers and practitioners to extend our ideas and evaluate the use of the SiMPle in different MIR tasks. To this end, we created a website [19] with the complete source code used in our experiments and videos highlighting some of the results presented in this work.

4.1 Datasets

We evaluate our method in different scenarios regarding music styles and size of the databases. Specifically, we tested the proposed distance measure's utility for assessing both popular and classical recordings.

The first database considered is the YouTube Covers [18], composed of 50 different compositions, each one containing 7 different recordings obtained from YouTube videos. The data was originally split into training and testing partitions, in which the training set is composed of the original recording in studio and a live version performed by the same artist. To allow comparisons to the literature, we follow the same configuration.

The second dataset we consider is the widely used collection of Chopin's Mazurkas. The set of Mazurkas used in this work contains 2,919 recordings of 49 pieces for piano. The number of recordings of each song varies from 41 to 95.

4.2 Results and Discussion

In order to assess the performance of our method, we used three commonly applied evaluation measures: mean average precision (MAP), precision at 10 (P@10), and mean rank of first correctly identified cover (MR1). Note that for MR1, smaller values are better.

For both the YouTube Covers and Mazurkas datasets, we compared our algorithm using results previously presented in the literature. For the former case, in addition to comparing to the results presented in the paper for which the dataset was created [18], we carefully implemented the algorithm for local alignments based on the chroma binary distance [15]. Table 1 shows the results.

Algorithm	MAP	P@10	MR1
DTW	0.425	0.114	11.69
Silva et al. [18]	0.478	0.126	8.49
Serrà et al. [15]	0.525	0.132	9.43
SiMPle	**0.591**	**0.140**	**7.91**

Table 1. Mean average precision (MAP), precision at 10 (P@10), and mean rank of first correctly identified cover (MR1) on the YouTube Covers dataset. Given that this dataset has only two recordings per song in the training set, the maximum value to P@10 is 0.2.

Our method achieved the best results in this experiment. In addition, we note that our method is notably faster than the second best (Serrà et al.). Specifically, while our method took 1.3 hours, the other method took approximately one week to run on the same computer[1].

[1] In our experiments, we used an 8-core Intel® Core ™ i7-6700K CPU @ 4.00GHz with 32GB of RAM memory running Windows 10®. All our codes were implemented and executed using Matlab R2014a®.

We acknowledge that we did not invest a lot of effort optimizing the competing method. However, we do not believe that any code optimization is capable of significantly reducing the performance gap.

We also consider the Mazurkas dataset. In addition to the results achieved by DTW, we report MAP results documented in the literature, which were achieved by retrieving the recordings by structural similarity strategies using this data. Specifically, the subset of mazurkas used in this work is exactly the same as the used in [2] and [17] and has only minor differences to the dataset used in [6]. Although [15] is considered the state-of-the-art for cover song recognition, we do not include its results due to the high time complexity. Table 2 shows the results.

Algorithm	MAP	P@10	MR1
DTW	**0.882**	0.949	4.05
Bello [2]	0.767	-	-
Silva et al. [17]	0.795	-	-
Grosche et al. [6]	0.819	-	-
SiMPle	0.880	**0.952**	**2.33**

Table 2. Mean average precision (MAP), precision at 10 (P@10), and mean rank of first correctly identified cover (MR1) on the Mazurkas dataset.

The structures of the pieces on this dataset are respected in most of the recordings. In this case, DTW performs similar than our algorithm. However, our method is faster (approximately two times in our experiments) and has several advantages over DTW, such as its incremental property, discussed in the next section.

4.3 Streaming Cover Song Recognition

Real-time audio matching has attracted the attention of the community in the last years. In this scenario, the input is a stream of audio and the output is a sorted list of similar objects in a database.

In this section, we evaluate our algorithm in an online cover song recognition scenario. For concreteness, consider that a TV station is broadcasting a live concert. In order to automatically present the name of the song to the viewers or to synchronize the concert with a second screen app, we would like to take the streaming audio as input to our algorithm and be able to recognize what song the band is playing as soon as possible. To accomplish this task, we need to match the input to a set of (previously processed) recordings.

In addition to allowing the fast calculation of all the distances of a subsequence to a whole song, the proposed algorithm has an incremental property that can be exploited to estimate cover song similarity in a streaming fashion. If we have a previously calculated SiMPle, then, when we extract a new vector of (chroma) features, we do not need to recalculate the whole SiMPle from the beginning. Instead, just two quick steps are required:

- Calculation of the distance profile to the new subsequence, i.e., the distance of the last observed subsequence (including the new feature vector) to all the subsequences of the original song;
- Update of SiMPle by selecting the minimum value between the new distance profile and the previous SiMPle for each subsequence.

These steps are done by the Algorithm 2.

Algorithm 2. Procedure to incrementally update SiMPle and SiMPle index

Input: The current time series A and B, the new chroma vector c, the desired subsequence length m, and the current SiMPle P_{AB} and SiMPle index I_{AB}

Output: The updated SiMPle $P_{AB,new}$ and the associated SiMPle index $I_{AB,new}$

1 $newB \leftarrow$ Concatenate(B,c), $n_B \leftarrow$ Length($newB$)
2 $D \leftarrow$ MASS($newB[n_B\text{-}m\text{+}1: n_B]$, A) // c.f. [10]
3 P_{AB}, $I_{AB} \leftarrow$ ElementWiseMin(P_{AB}, I_{AB}, D, $n_B\text{-}m\text{+}1$)
4 $P_{AB,last}$, $I_{AB,last} \leftarrow$ FindMin(D)
5 $P_{AB,new} \leftarrow [P_{AB}, P_{AB,last}]$, $I_{AB,new} \leftarrow [I_{AB}, I_{AB,last}]$
6 **return** $P_{AB,new}$, $I_{AB,new}$

To evaluate the ability of our method for streaming recognition, we performed a simple experiment simulating the previously described scenario. First, we extracted features from each track in the dataset of original recordings. For clarity, we will refer to this database as the *training set*. Then, we randomly chose another recording as our query and processed it according to the following steps. We begin extracting features from the first three seconds of the query in order to calculate the first distance estimation to each training object. After this initial step, for each second of the query, we repeat the process of extracting features and re-estimating the distance measure to the training set.

In this experiment, we used the Mazurkas dataset with two CENS per second. The training set is composed of the first recording (in alphabetical order) of each piece. We used a performance with approximately 275 seconds as a query, and we were able to maintain the process faster than real-time. Specifically, the updates took approximately 0.7 seconds to extract the features, update SiMPle, and recalculate the distance for all the training objects.

Figure 2 visualizes the changes in distance estimation in an audio streaming query session. In this case, we used a recording of the "*Mazurka in F major, Op. 68, No. 3*" as query. In the first estimation, its training version appears as the sixth nearest neighbor. However as we see more evidence, it quickly becomes the best match.

Figure 2. Changes in the distance when querying a recording of the "*Mazurka in F major, Op. 68, No. 3*" in a streaming fashion. The graphs represent the top 6 matches after processing 3 (*left*), 5 (*middle*), and 10 (*right*) seconds of the audio.

Another strategy that can be used in this scenario is an amnesic sliding window, in order to forget old values and further speedup the matching of new subsequences. For a given window length *w*, we can maintain just the last *w* values in the SiMPle. In this way, a change in the distribution of the distance estimates may assist in the identification of the ending and beginning of a song. At the same time, the positions of the most recently matched sections can be used as estimation of the moment in the current song. These ideas may help to identify songs being sequentially played in a random or unknown order.

5. EXPANDING THE RANGE OF APPLICATIONS OF SiMPle

In this work, we focus on assessing music similarity by joining subsequences. While we evaluate our method on the cover song recognition task, we claim that the SiMPle is a powerful tool for other music processing tasks. To reinforce this argument, we present insights on how to use SiMPle in different application domains, as well some initial results. The methods presented in this section have room for improvements, but they are simple yet effective. We intend to further explore and evaluate SiMPle in (at least) the tasks listed below.

In contrast to the previous experiments, when we use the self-similarity join to highlight points of interest in a recording, we apply ten CENS per second.

5.1 Motifs and Discords

The SiMPle from a self-similarity join has several exploitable properties. For example, the lowest points correspond to the locations of the most faithfully repeated section (i.e., the *chorus* or *refrain*). Between several definitions of *motifs* in the literature, such as *harmonic* or *melodic motifs*, its simplest definition is the *closest pair of subsequences*. As noted in the time series literature, given the best motif pair, other definitions of motifs can be solved by minor additional calculations [9].

On the other hand, the highest point on the SiMPle corresponds to the "most unique" snippet from the recording. The procedure to search for such a subsequence that is the furthest from any other, known as *discord discovery*, can be used in music processing to find interesting segments in recordings. For example, it can be used to identify a solo, improvisation segments or the bridge.

For example, consider the song "*Let It Be*" by The Beatles. Figure 3. shows the SiMPle obtained for this track and points to their discord and pair of best motifs.

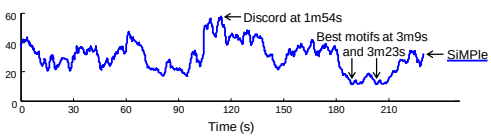


Figure 3. The pair of best motifs in a recording is determined by the subsequences starting at the positions of the minimum values of its SiMPle. At the same time, the position of the highest value points to the beginning of the discord excerpt.

While the motifs point to refrains, the discord includes bridge and the beginning of the guitar solo.

5.2 Audio Thumbnailing

Audio thumbnails are short representative excerpts of audio recordings. Thumbnails have several applications in music information retrieval. For example, they can be used as the snippet shown the result of a search to the user. In a commercial application, they can be used as the preview to a potential costumer in an online music store.

There is a consensus in the MIR community that the "ideal" music thumbnail is the most repeated excerpt, such as the chorus [1]. Using this assumption, the application of SiMPle to identify a thumbnail is direct. Consider the SiMPle index obtained by the self-join procedure. The thumbnail is given by the subsequence starting in the position that is most used as a nearest neighbor. In other words, the beginning of the thumbnail is given by the position related to the mode of SiMPle index.

To illustrate this idea, we considered the song "*New York, New York"* by Frank Sinatra. Looking for a 30 seconds thumbnail, we found an excerpt that is comprised of the last refrain, as well as the famous (brass) instrumental basis of the song. Figure 4 shows the histogram of the SiMPle index found in this experiment.

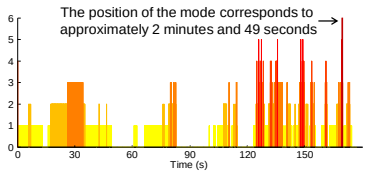


Figure 4. Histogram of SiMPle index for the song "*New York, New York*". Each bar counts how many times the subsequence starting at that point was considered the nearest neighbor of any other. We consider the subsequence represented by the most prominent peak as the thumbnail for this recording.

5.3 Visualization

The visualization of music structure aids the understanding of the music content. Introduced in [21], the arc diagram is a powerful tool to visualize repetitive segments in MIDI files [22] and audio recordings [23]. This approach represents a song by plotting arcs linking *repeated* segments.

All the information required to create such arcs are completely comprised on the SiMPle and the SiMPle index obtained by a self-join. Specifically, SiMPle provides the distances between subsequences, which can be used to determine if they are similar enough to exist a link between them and to define the color or transparency of each arc. The SiMPle index can be used to define both the positions and width of the arcs.

Figure 5 shows the scatter plot of the SiMPle index for "*Hotel California*" by Eagles. In this figure, there is a point (x,y) only if y is the nearest neighbor of x. The clear diagonals on this plot represent regions of n points such that the nearest neighbors of $[x,x+1,\ldots,x+n-1]$ are approximately $[y,y+1,\ldots,y+n-1]$. If the distance between such excerpts is low, then these regions may have a link between them. For this example, we defined the mean value of the SiMPle in that region as the distance threshold between the segments, in order to resolve if they should

stablish a link. Such threshold has direct impact on the number of arcs plotted.

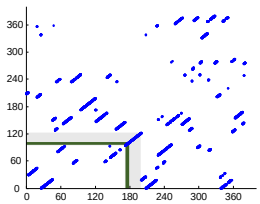


Figure 5. Scatter plot of the SiMPle index for the song "*Hotel California*". The (light) gray area indicates a possible link, but only the values in the (dark) green area represent subsequences with distance lower than the threshold.

By using a simple algorithm to spot such diagonals, we only need to define a threshold of distance and minimum length of the linkages. We set the width of the links in our experiment to be greater than or equal to 5 seconds. Figure 6 shows the resulting arc plot for the example shown in Figure 5.

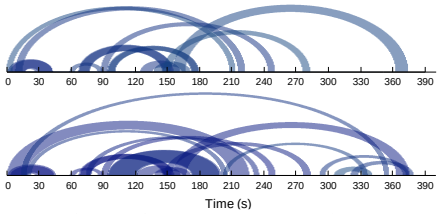


Figure 6. Arc plot for the song "*Hotel California*". These plots show the difference between using the mean value of SiMPle as distance threshold (*above*) and no distance threshold at all (*below*). The color of the arcs are related to their relevance, i.e., as darker the arc, closer the subsequences linked by it.

5.4 Endless Reproduction

Consider a music excerpt s_1, which starts at the time t_1 of a specific song, has a small distance to its nearest neighbor s_2, which starts at time t_2. When the reproduction of this song arrives t_1, we can make a random decision to "skip" the reproduction to t_2. Given that s_1 and s_2 are similar, this jump may be imperceptible to the listener. By creating several points of skip, we are able to define a sequence of jumps that creates an endless reproduction of the song. A well-known deployed example of this kind of player is the Infinite Jukebox [7].

The distance values obtained by the self-join represent how similar each subsequence is to its nearest neighbor in another region of the song. Adopting a small threshold to the distance between subsequences, we can use SiMPle to define the jumps. These characteristics may be explored in order to create a player for endless reproduction. We refer the interested reader to the supporting website [19] for examples of this functionality.

5.5 Sampling Identification

In addition to providing a *global* distance estimation between different songs, SiMPle is also powerful to examine *local* similarities. An interesting application that may exploit this ability is the automatic identification of samples. Sampling is the act of "borrowing" the instrumental basis or main melody from another song. This is a common approach in electronic and hip-hop music.

In contrast to cover versions, sampling is used as a virtual "instrument" to compose new songs. However, algorithms that look only for local patterns to identify versions of the same track may classify a recording using samples as a cover song. Using SiMPle, we can discover that the sampling excerpts have small distance values. In contrast, the segments related to the new song have significantly higher values.

Figure 7 shows an example of the usage of SiMPle to spot sampling. In this case, we compare the song "*Under Pressure*" by Queen and David Bowie with "*Ice Ice Baby*" by Vanilla Ice. Most of the continuous regions with values lower than the mean refer to the sampling of the famous bass line of the former song.

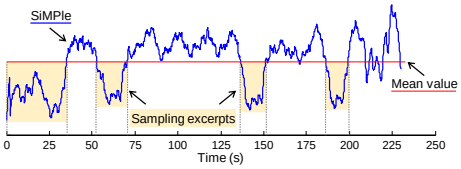


Figure 7. SiMPle (in blue) obtained between the songs "*Ice Ice Baby*" and "*Under Pressure*". The continuous regions below the mean value (in red) represent the excerpts sampled by Vanilla Ice from Queen's song.

6. CONCLUSIONS

In this paper, we introduced a technique to exploit subsequences joins to assess similarity in music. The presented method is very fast and requires only one parameter that is intuitively set in music applications.

While we focused our evaluation on the cover song recognition, we have shown that our approach has the potential for applications in different MIR tasks. We intend to further investigate the use of matrix profiles in the tasks discussed in Section 5 and the effects of different features in the process.

The main limitation of the proposed method is that the use of only one nearest neighbor may be sensitive to hubs, i.e., subsequences that are considered the nearest neighbor of many other snippets. In addition, SiMPle cannot be directly used to identify regions where several subsequences are next to each other, composing a dense region. For this reason, we intend to measure the impact of the reduction in the amount of information in different tasks. Given that, we plan to explore how to incorporate additional information to SiMPle with no loss of time and space efficiency.

We have encouraged the community to confirm our results and explore or extend our ideas by making the code freely available [19].

Acknowledgements: The authors would like to thank FAPESP by the grants #2013/26151-5 and #2015/07628-0 and CNPq by the grants #303083/2013-1 and #446330/2014-0, and NSF by the grant IIS 1510741.

7. REFERENCES

[1] M. A. Bartsch and G. H. Wakefield. "Audio thumbnailing of popular music using chroma-based representations". *IEEE Transactions on Multimedia*, Vol. 7, No. 1, pp 96–104, 2005.

[2] J. P. Bello. "Measuring Structural Similarity in Music". *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 9, No. 7, pp. 2013–2025, 2011.

[3] AHRC Research - Centre for the History and Analysis of Recorded Music. "Mazurka project". url: www.mazurka.org.uk/ (accessed 24 May, 2016)

[4] J. J. Carabias-Orti, F. J. Rodriguez-Serrano, P. Vera-Candeas, N. Ruiz-Reyes, and F. J. Canadas-Quesada. "An audio to score alignment framework using spectral factorization and dynamic time warping". *International Society for Music Information Retrieval Conference*, pp. 742–748, 2015.

[5] J. Foote. "Visualizing music and audio using self-similarity". *ACM International Conference on Multimedia*, pp. 77–80, 1999.

[6] P. Grosche, J. Serrà, M. Müller, and J. L. Arcos. "Structure-based audio fingerprinting for music retrieval". *International Society for Music Information Retrieval Conference*, pp. 55–60, 2012.

[7] P. Lamere. "The Infinite Jukebox", url: www.infinitejuke.com/ (accessed 24 May, 2016).

[8] B. McFee and D. P. W. Ellis. "Analyzing song structure with spectral clustering", *International Society for Music Information Retrieval Conference*, pp. 405–410, 2014.

[9] A. Mueen, E. Keogh, Q. Zhu, S. Cash, and M. B. Westover. "Exact discovery of time series motifs", *SIAM International Conference on Data Mining*, pp. 473–484, 2009.

[10] A. Mueen, K. Viswanathan, C. K. Gupta and E. Keogh. "The fastest similarity search algorithm for time series subsequences under Euclidean distance", url: www.cs.unm.edu/~mueen/FastestSimilaritySearch.html (accessed 24 May, 2016).

[11] M. Müller. "Dynamic time warping". *Information retrieval for music and motion*, pp. 69-84, Springer, 2007.

[12] M. Müller, F. Kurth, and M. Clausen. "Audio matching via chroma-based statistical features". *International Society for Music Information Retrieval Conference*, pp. 288–295, 2005.

[13] S. Salvador and P. Chan. "Toward accurate dynamic time warping in linear time and space". *Intelligent Data Analysis*, Vol. 11, No. 5, pp 561–580, 2007.

[14] J. Serrà, E. Gómez, and P. Herrera. "Transposing chroma representations to a common key". *CS Conference on The Use of Symbols to Represent Music and Multimedia Objects*, pp. 45–48, 2008.

[15] J. Serrà, E. Gómez, P. Herrera, and X. Serra. "Chroma binary similarity and local alignment applied to cover song identification". *IEEE Transactions on Audio, Speech, and Language Processing*. Vol. 16, No. 6, pp. 1138–1151, 2008.

[16] J. Serrà, X. Serra, and R. G. Andrzejak. "Cross recurrence quantification for cover song identification". *New Journal of Physics*, Vol. 11, No. 9, pp. 093017, 2009.

[17] D. F. Silva, H. Papadopoulos, G. E. A. P. A. Batista, and D. P. W. Ellis. "A video compression-based approach to measure music structural similarity". *International Society for Music Information Retrieval Conference*, pp. 95–10, 2014.

[18] D. F. Silva, V. M. A. Souza, and G. E. A. P. A. Batista. "Music shapelets for fast cover song recognition". *International Society for Music Information Retrieval Conference*, pp. 441–447, 2015.

[19] D. F. Silva, C.-C. M. Yeh, G. E. A. P. A. Batista, E. Keogh. "Supporting website for this work", url: http://sites.google.com/site/ismir2016simple/ (accessed 24 May, 2016).

[20] W.-H. Tsai, H.-M. Yu, and H.-M. Wang. "Using the similarity of main melodies to identify cover versions of popular songs for music document retrieval". *Journal of Information Science and Engineering*, vol. 24, No. 6, pp. 1669–1687, 2008.

[21] M. Wattenberg. "Arc diagrams: Visualizing structure in strings". *IEEE Symposium on Information Visualization*, pp 110–116, 2002.

[22] M. Wattenberg. "The Shape of Song", url: www.turbulence.org/Works/song/ (accessed 24 May, 2016).

[23] H. H. Wu, J. P. Bello. "Audio-based music visualization for music structure analysis". *Sound and Music Computing Conference*, pp. 1–6, 2010.

SCORE-INFORMED IDENTIFICATION OF MISSING AND EXTRA NOTES IN PIANO RECORDINGS

Sebastian Ewert[1] **Siying Wang**[1] **Meinard Müller**[2] **Mark Sandler**[1]

[1] Centre for Digital Music (C4DM), Queen Mary University of London, UK
[2] International Audio Laboratories Erlangen, Germany

ABSTRACT

A main goal in music tuition is to enable a student to play a score without mistakes, where common mistakes include missing notes or playing additional extra ones. To automatically detect these mistakes, a first idea is to use a music transcription method to detect notes played in an audio recording and to compare the results with a corresponding score. However, as the number of transcription errors produced by standard methods is often considerably higher than the number of actual mistakes, the results are often of limited use. In contrast, our method exploits that the score already provides rough information about what we seek to detect in the audio, which allows us to construct a tailored transcription method. In particular, we employ score-informed source separation techniques to learn for each score pitch a set of templates capturing the spectral properties of that pitch. After extrapolating the resulting template dictionary to pitches not in the score, we estimate the activity of each MIDI pitch over time. Finally, making again use of the score, we choose for each pitch an individualized threshold to differentiate note onsets from spurious activity in an optimized way. We indicate the accuracy of our approach on a dataset of piano pieces commonly used in education.

1. INTRODUCTION

Automatic music transcription (AMT) has a long history in music signal processing, with early approaches dating back to the 1970s [1]. Despite the considerable interest in the topic, the challenges inherent to the task are still to overcome by state-of-the-art methods, with error rates for note detection typically between 20 and 40 percent, or even above, for polyphonic music [2–8]. While these error rates can drop considerably if rich prior knowledge can be provided [9, 10], the accuracy achievable in the more general case still prevents the use of AMT technologies in many useful applications.

This paper is motivated by a music tuition application, where a central learning outcome is to enable the student to read and reproduce (simple) musical scores using an instrument. In this scenario, a natural use of AMT technologies could be to detect which notes have been played by the student and to compare the results against a reference score – this way one could give feedback, highlighting where notes in the score have not been played (*missed notes*) and where notes have been played that cannot be found in the score

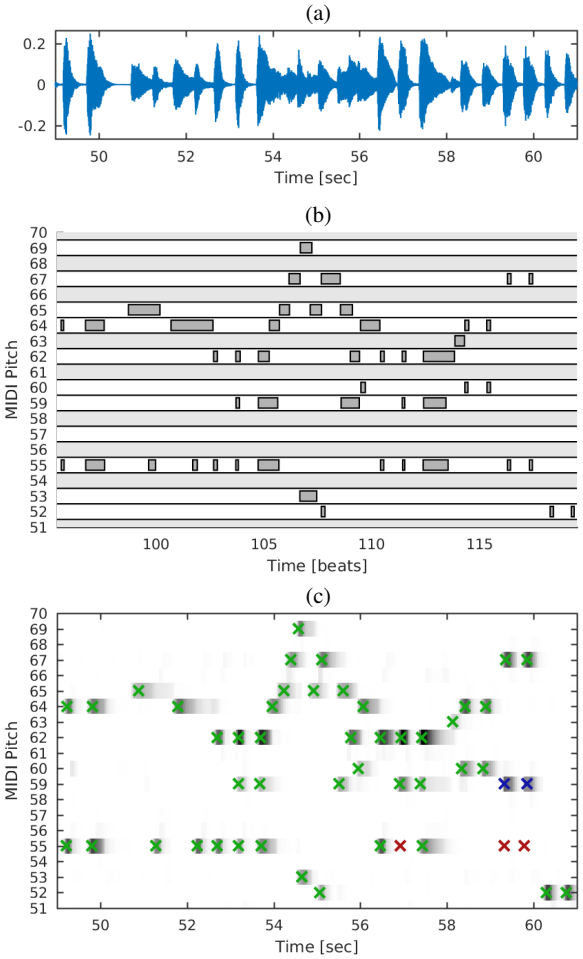


Figure 1. Given (a) an audio recording and (b) a score (e.g. as a MIDI file) for a piece of music, our method (c) estimates which notes have been played correctly (green/light crosses), have been missed (red/dark crosses for pitch 55) or have been added (blue/dark crosses for pitch 59) in the recording compared to the score.

© Sebastian Ewert, Siying Wang, Meinard Müller and Mark Sandler. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Sebastian Ewert, Siying Wang, Meinard Müller and Mark Sandler. "Score-Informed Identification of Missing and Extra Notes in Piano Recordings", 17th International Society for Music Information Retrieval Conference, 2016.

(*extra notes*). Unfortunately, the relatively low accuracy of standard AMT methods prevents such applications: the number of mistakes a student makes is typically several times lower than the errors produced by AMT methods.

Using a standard AMT method in a music tuition scenario as described above, however, would ignore a highly valuable source of prior knowledge: the score. Therefore, the authors in [11] make use of the score by first aligning the score to the audio, synthesizing the score using a wavetable method, and then transcribing both the real and the synthesized audio using an AMT method. To lower the number of falsely detected notes for the real recording, the method discards any detected note if the same note is also detected in the synthesized recording while no corresponding note can be found in the score. Here, the underlying assumption is that in such a situation, the local note constellation might lead to uncertainty in the spectrum, which could cause an error in their proposed method. To improve the results further, the method requires the availability of single note recordings for the instrument to be transcribed (under the same recording conditions) – a requirement not unrealistic to fulfil in this application scenario but leading to additional demands for the user. Under these additional constraints, the method lowered the number of transcription errors considerably compared to standard AMT methods. To the best of the authors' knowledge, the method presented in [11] is the only score-informed transcription method in existence.

Overall, the core concept in [11] is to use the score information to post-process the transcription results from a standard AMT method. In contrast, the main idea in this paper is to exploit the available score information to adapt the transcription method itself to a given recording. To this end, we use the score to modify two central components of an AMT system: the set of spectral patterns used to identify note objects in a time-frequency representation, and the decision process responsible for differentiating actual note events from spurious note activities. In particular, after aligning the score to the audio recording, we employ the score information to constrain the learning process in non-negative matrix factorization similar to strategies used in score-informed source separation [12]. As a result, we obtain for each pitch in the score a set of template vectors that capture the spectro-temporal behaviour of that pitch – adapted to the given recording. Next, we extrapolate the template vectors to cover the entire MIDI range (including pitches not used in the score), and compute an activity for each pitch over time. After that we again make use of the score to analyze the resulting activities: we set, for each pitch, a threshold used to differentiate between noise and real notes such that the resulting note onsets correspond to the given score as closely as possible. Finally, the resulting transcription is compared to the given score, which enables the classification of note events as correct, missing or extra. This way, our method can use highly adapted spectral patterns in the acoustic model eliminating the need for additional single note recordings, and remove many spurious errors in the detection stage. An example output of our method is shown in Fig. 1, where correctly played notes are marked in green, missing notes in red and extra notes in blue.

The remainder of this paper is organized as follows. In Section 2, we describe the details of our proposed method. In Section 3, we report on experimental results using a dataset comprising recordings of pieces used in piano education. We conclude in Section 4 with a prospect on future work.

2. PROPOSED METHOD

2.1 Step 1: Score-Audio Alignment

As a first step in our proposed method, we align a score (given as a MIDI file) to an audio recording of a student playing the corresponding piece. For this purpose, we employ the method proposed in [13], which combines chroma with onset indicator features to increase the temporal accuracy of the resulting alignments. Since we expect differences on the note level between the score and the audio recording related to the playing mistakes, we manually checked the temporal accuracy of the method but found the alignments to be robust in this scenario. It should be noted, however, that the method is not designed to cope with structural differences (e.g. the student adding repetitions of some segments in the score, or leaving out certain parts) – if such differences are to be expected, partial alignment techniques should be used instead [14, 15].

2.2 Step 2: Score-Informed Adaptive Dictionary Learning

As a result of the alignment, we now roughly know for each note in the score, the corresponding or expected position in the audio. Next, we use this information to learn how each pitch manifests in a time-frequency representation of the audio recording, employing techniques similarly used in score-informed source separation (SISS). There are various SISS approaches to choose from: Early methods essentially integrated the score information into existing signal models, which already drastically boosted the stability of the methods. These signal models, however, were designed for blind source separation and thus have the trade-off between the capacity to model details (*variance*) and the robustness in the parameter estimation (*bias*) heavily leaned towards the bias. For example, various approaches make specific assumptions to keep the parameter space small, such as that partials of a harmonic sound behave like a Gaussian in frequency direction [16], are highly stationary in a single frame [17] or occur as part of predefined clusters of harmonics [6]. However, with score information providing extremely rich prior knowledge, later approaches found that the variance-bias trade-off can be shifted considerably towards variance.

For our method, we adapt an approach that makes fewer assumptions about how partials manifests and rather learns these properties from data. The basic idea is to constrain a (shift-invariant) non-negative matrix factorization (NMF) based model using the score, making only use of rough information and allowing the learning process to identify

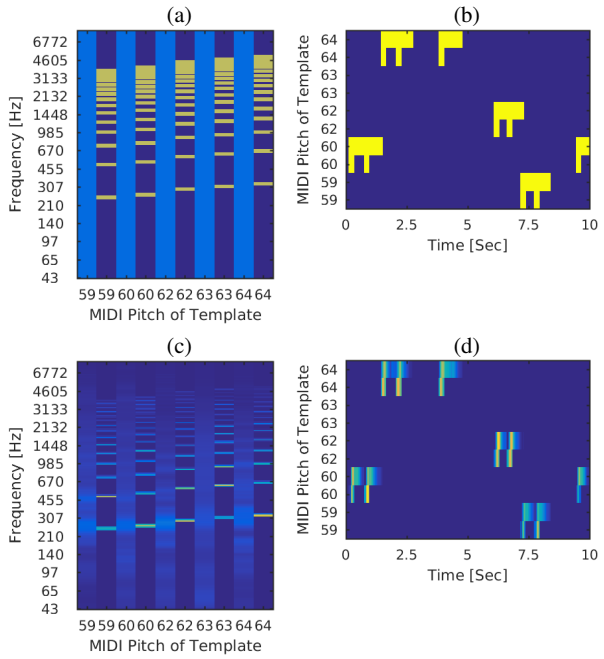


Figure 2. Score-Informed Dictionary Learning: Using multiplicative updates in non-negative matrix factorization, semantically meaningful constraints can easily be enforced by setting individual entries to zero (dark blue): Templates and activations after the initialization (a)/(b) and after the optimization process (c)/(d).

the details, see also [12]. Since we focus on piano recordings where tuning shifts in a single recording or vibrato do not occur, we do not make use of shift invariance. In the following, we assume general familiarity with NMF and refer to [18] for further details. Let $V \in \mathbb{R}^{M \times N}$ be a magnitude spectrogram of our audio recording, with logarithmic spacing for the frequency axis. We approximate V as a product of two non-negative matrices $W \in \mathbb{R}^{M \times K}$ and $H \in \mathbb{R}^{K \times N}$, where the columns of W are called (spectral) templates and the rows in H the corresponding activities. We start by allocating two NMF templates to each pitch in the score – one for the attack and one for the sustain part. The sustain part of a piano is harmonic in nature and thus we do not expect significant energy in frequencies that lie between its partials. We implement this constraint as in [12] by initializing for each sustain template only those entries with positive values that are close to a harmonic of the pitch associated with the template, i.e. entries between partials are set to zero, compare Fig. 2a. This constraint will remain intact throughout the NMF learning process as we will use multiplicative update rules and thus setting entries to zero is a straightforward way to efficiently implement certain constraints in NMF, while letting some room for the NMF process to learn where exactly each partial is and how it spectrally manifests. The attack templates are initialized with a uniform energy distribution to account for their broadband properties.

Constraints on the activations are implemented in a similar way: activations are set to zero if a pitch is known to be inactive in a time segment, with a tolerance used to account for alignment inaccuracies, compare Fig. 2b. To counter the lack of constraints for attack templates, the corresponding activations are subject to stricter rules: attack templates are only allowed to be used in a close vicinity around expected onset positions. After these initializations, the method presented in [12] employs the commonly used Lee-Seung NMF update rules [18] to minimize a generalized Kullback-Leibler divergence between V and WH. This way, the NMF learning process refines the information within the unconstrained areas on W and H.

However, we propose a modified learning process that enhances the broadband properties for the attack templates. More precisely, we include attack templates to bind the broadband energy related to onsets and thus reduce the number of spurious note detections. We observed, however, that depending on the piece, the attack templates would capture too much of the harmonic energy, which interfered with the note detection later on. Since harmonic energy manifest as peaks along the frequency axis, we discourage such peaks for attack templates and favour smoothness using an additional spectral continuity constraint in the objective function:

$$f(W,H) := \sum_{m,n} V_{m,n} \log\left(\frac{V_{m,n}}{(WH)_{m,n}}\right) - V_{m,n} + (WH)_{m,n} + \sigma \sum_{m} \sum_{k \in \mathcal{A}} (W_{m,k} - W_{m-1,k})^2$$

where the first sum is the generalized Kullback-Leibler divergence and the second sum is a total variation term in frequency direction, with $\mathcal{A} \subset \{1, \ldots, K\}$ denoting the index set of attack templates and σ controlling the relative importance of the two terms. Note that $W_{m,k} - W_{m-1,k} = (F \star W_{:,k})(m)$, where $W_{:,k}$ denotes the k-th column of W and $F = (-1, 1)$ is a high-pass filter. To find a local minimum for this bi-convex problem, we propose the following iterative update rules alternating between W and H (we omit the derivation for a lack of space but followed similar strategies as used for example in [19]):

$$W_{m,k} \leftarrow W_{m,k} \cdot \frac{\sum_n H_{k,n} \frac{V_{m,n}}{(WH)_{m,n}} + \mathcal{I}_{\mathcal{A}}(k)\, 2\sigma (W_{m+1,k} + W_{m-1,k})}{\sum_n H_{k,n} + \mathcal{I}_{\mathcal{A}}(k)\, 4\sigma W_{m,k}}$$

$$W_{m,k} \leftarrow \frac{W_{m,k}}{\sum_{\tilde{m}} W_{\tilde{m},k}}$$

$$H_{k,n} \leftarrow H_{k,n} \cdot \frac{\sum_m W_{m,k} \frac{V_{m,n}}{(WH)_{m,n}}}{\sum_m W_{m,k}}$$

where $\mathcal{I}_{\mathcal{A}}$ is the indicator function for $\mathcal{A}$. The result of this update process is shown in Fig. 2c and d. It is clearly visible how the learning process refined the unconstrained areas in W and H, closely reflecting the acoustical properties in the recording. Further, the total variation term led to attack templates with broadband characteristics for all pitches, while still capturing the non-uniform, pitch dependent energy distribution typical for piano attacks.

2.3 Step 3: Dictionary Extrapolation and Residual Modelling

All notes not reflected by the score naturally lead to a difference or residual between V and WH as observed also in [20]. To model this residual, the next step in our proposed method is to extrapolate our learnt dictionary of spectral templates to the complete MIDI range, which enables us to transcribe pitches not used in the score. Since we use a time-frequency representation with a logarithmic frequency scale, we can implement this step by a simple shift operation: for each MIDI pitch not in the score, we find the closest pitch in the score and shift the two associated templates by the number of frequency bins corresponding to the difference between the two pitches. After this operation we can use our recording-specific full-range dictionary to compute activities for all MIDI pitches. To this end, we add an activity row to H for each extrapolated template and reset any zero constraints in H by adding a small value to all entries. Then, without updating W, we re-estimate this full-range H using the same update rules as given above.

2.4 Step 4: Onset Detection Using Score-Informed Adaptive Thresholding

After convergence, we next analyze H to detect note onsets. A straightforward solution would be to add, for each pitch and in each time frame, the activity for the two templates associated with that pitch and detecting peaks afterwards in time direction. This approach, however, leads to several problems. To illustrate these, we look again at Fig. 2c, and compare the different attack templates learnt by our procedure. As we can see, the individual attack templates do differ for different pitches, yet their energy distribution is quite broadband leading to considerable overlap or similarity between some attack templates. Therefore, when we compute H there is often very little difference with respect to the objective function if we activate the attack template associated with the correct pitch, or an attack template for a neighboring pitch (from an optimization point of view, these similarities lead to relatively wide plateaus in the objective function, where all solutions are almost equally good). The activity in these neighboring pitches led to wrong note detections.

As one solution, inspired by the methods presented in [21, 22], we initially incorporated a Markov process into the learning process described above. Such a process can be employed to model that if a certain template (e.g. for the attack part) is being using in one frame, another template (e.g. for the sustain part) has to be used in the next frame. This extension often solved the problem described above as attack templates cannot be used without their sustain parts anymore. Unfortunately, the dictionary learning process with this extension is not (bi-)convex anymore and in practice we found the learning process to regularly get stuck in poor local minima leading to less accurate transcription results.

A much simpler solution, however, solved the above problems in our experiments similar to the Markov process, without the numerical issues associated with it: we simply ignore activities for attack templates. Here, the idea is that as long as the broadband onset energy is meaningfully captured by some templates, we do not need to care about spurious note detections caused by this energy and can focus entirely on detecting peaks in the cleaner, more discriminative sustain part to detect the notes (compare also Fig. 2d). Since this simpler solution turned out to be more robust, efficient and accurate overall, we use this approach in the following. The result of using only the sustain activities is shown in the background of Fig. 1. Comparing these results to standard NMF-based transcription methods, these activities are much cleaner and easier to interpret – a result of using learnt, recording-specific templates.

As a next step, we need to differentiate real onsets from spurious activity. A common technique in the AMT literature is to simply use a global threshold to identify peaks in the activity. As another approach often used for sustained instruments like the violin or the flute, hidden Markov models (HMMs) implement a similar idea but add capabilities to smooth over local activity fluctuations, which might otherwise be detected as onsets [2]. We tried both approaches for our method but given the distinctive, fast energy decay for piano notes, we could not identify significant benefits for the somewhat more complex HMM solution and thus only report on our thresholding based results. A main difference in our approach to standard AMT methods, however, is the use of *pitch-dependent* thresholds, which we optimize again using the score information. The main reason why this pitch dependency is useful is that loudness perception in the human auditory system non-linearly depends on the frequency and is highly complex for non-sinusoidal sounds. Therefore, to reach a specific loudness for a given pitch, a pianist might strike the corresponding key with different intensity compared to another pitch, which can lead to considerable differences in measured energy.

To find pitch-wise thresholds, our method first generates $C \in \mathbb{N}$ threshold candidates, which are uniformly distributed between 0 and $\max_{k,n} H_{k,n}$. Next, we use each candidate to find note onsets in each activity row in H that is associated with a pitch in the score. Then, we evaluate how many of the detected onsets correspond to notes in the aligned score, how many are extra and how many are missing – expressed as a precision, recall and F-measure value for each candidate and pitch. To increase the robustness of this step, in particular for pitches with only few notes, we compute these candidate ratings not only using the notes for a single pitch but include the notes and onsets for the N closest neighbouring pitches. For example, to rate threshold candidates for MIDI pitch P, we compute the F-measure using all onsets and notes corresponding to, for example, MIDI pitch $P-1$ to $P+1$. The result of this step is a curve for each pitch showing the F-measure for each candidate, from which we choose the lowest threshold maximizing the F-measure, compare Fig. 3. This way, we can choose a threshold that generates the least amount of extra and missing notes, or alternatively, a threshold that maximizes the match between the detected onsets and the given score. Thresholds for pitches not used in the score are

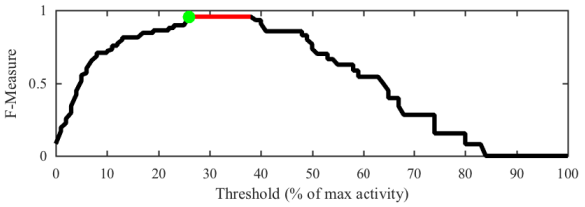


Figure 3. Adaptive and pitch-dependent thresholding: For each pitch we choose the smallest threshold maximizing the F-measure we obtain by comparing the detected offsets against the aligned nominal score. The red entries show threshold candidates having maximal F-measure.

interpolated from the thresholds for neighbouring pitches that are in the score.

2.5 Step 5: Score-Informed Onset Classification

Using these thresholds, we create a final transcription result for each pitch. As our last step, we try to identify for each detected onset a corresponding note in the aligned score, which allows us to classify each onset as either *correct* (i.e. note is played and is in the score) or *extra* (i.e. played but not in the score). All score notes without a corresponding onset are classified as *missing*. To identify these correspondences we use a temporal tolerance T of ± 250 ms, where T is a parameter that can be increased to account for local alignment problems or if the student cannot yet follow the rhythm faithfully (e.g. we observed concurrent notes being pulled apart by students for non-musical reasons). This classification is indicated in Fig. 1 using crosses having different colours for each class.

3. EXPERIMENTS

3.1 Dataset

We indicate the performance of our proposed method using a dataset [1] originally compiled in [11]. The dataset comprises seven pieces shown in Table 1 that were taken from the syllabus used by the Associated Board of the Royal Schools of Music for grades 1 and 2 in the 2011/2012 period. Making various intentional mistakes, a pianist played these pieces on a Yamaha U3 Disklavier, an acoustic upright piano capable of returning MIDI events encoding the keys being pressed. The dataset includes for each piece an audio recording, a MIDI file encoding the reference score, as well as three annotation MIDI files encoding the extra, missing and correctly played notes, respectively.

In initial tests using this dataset, we observed that the annotations were created in a quite rigid way. In particular, several note events in the score were associated with one missing and one extra note, which were in close vicinity of each other. Listening to the corresponding audio recording, we found that these events were seemingly played correctly. This could indicate that the annotation process was potentially a bit too strict in terms of temporal tolerance. Therefore, we modified the three annotation files in some cases. Other corrections included the case that a single score note was played more than once and we re-assigned in some cases which of the repeated notes should be considered as extra notes and which as the correctly played note, taking the timing of other notes into account. Further, some notes in the score were not played but were not found in the corresponding annotation of missing notes. We make these slightly modified annotation files available online [2] . It should be noted that these modifications were made before we started evaluating our proposed method.

ID	Composer	Title
1	Josef Haydn	Symp. No. 94: Andante (Hob I:94-02)
2	James Hook	Gavotta (Op. 81 No. 3)
3	Pauline Hall	Tarantella
4	Felix Swinstead	A Tender Flower
5	Johann Krieger	Sechs musicalische Partien: Bourrée
6	Johannes Brahms	The Sandman (WoO 31 No. 4)
7	Tim Richards (arr.)	Down by the Riverside

Table 1. Pieces of music used in the evaluation, see also [11].

3.2 Metrics

Our method yields a transcription along with a classification into correct, extra and missing notes. Using the available ground truth annotations, we can evaluate each class individually. In each class, we can identify up to a small temporal tolerance the number of true positives (TP), false positives (FP) and false negatives (FN). From these, we can derive the *Precision* $P = \frac{TP}{TP+FP}$, the *Recall* $R = \frac{TP}{TP+FN}$, the *F-measure* $2PR/(P+R)$ and the *Accuracy* $A = \frac{TP}{TP+FP+FN}$. We use a temporal tolerance of ± 250ms to account for the inherent difficulties aligning different versions of a piece with local differences, i.e. playing errors can lead to local uncertainties which position in the one version corresponds to which position in the other.

3.3 Results

The results for our method are shown in Table 2 for each class and piece separately. As we can see for the 'correct' class, with an F-measure of more than 99% the results are beyond the limits of standard transcription methods. However, this is expected as we can use prior knowledge provided by the score to tune our method to detect exactly these events. More interestingly are the results for the events we do not expect. With an F-measure of 94.5%, the results for the 'missing' class are almost on the same level as for the 'correct' class. The F-measure for the 'extra' class is 77.2%, which would be a good result for a standard AMT method but it is well below the results for the other two classes.

Let us investigate the reasons. A good starting point is piece number 6 where the results for the 'extra' class are well below average. In this recording, MIDI notes in the score with a pitch of 54 and 66 are consistently replaced in the recording with notes of MIDI pitch 53 and 65. In particular, pitches 54 and 66 are never actually played in the recording. Therefore, the dictionary learning process

[1] available online: `http://c4dm.eecs.qmul.ac.uk/rdr/`

[2] `http://www.eecs.qmul.ac.uk/~ewerts/`

ID	Class	Prec.	Recall	F-Meas.	Accur.
	C	100.0	100.0	100.0	100.0
1	E	100.0	71.4	83.3	71.4
	M	100.0	100.0	100.0	100.0
	C	100.0	99.7	99.8	99.7
2	E	90.0	81.8	85.7	75.0
	M	92.3	100.0	96.0	92.3
	C	99.2	99.2	99.2	98.4
3	E	100.0	66.7	80.0	66.7
	M	100.0	100.0	100.0	100.0
	C	98.7	100.0	99.3	98.7
4	E	80.0	80.0	80.0	66.7
	M	100.0	85.7	92.3	85.7
	C	99.5	98.6	99.1	98.1
5	E	75.0	92.3	82.8	70.6
	M	87.5	100.0	93.3	87.5
	C	99.2	99.2	99.2	98.4
6	E	50.0	52.9	51.4	34.6
	M	93.3	93.3	93.3	87.5
	C	99.5	97.1	98.3	96.7
7	E	75.0	80.0	77.4	63.2
	M	76.2	100.0	86.5	76.2
	C	**99.4**	**99.1**	**99.3**	**98.6**
Avg.	**E**	**81.4**	**75.0**	**77.2**	**64.0**
	M	**92.8**	**97.0**	**94.5**	**89.9**

Table 2. Evaluation results for our proposed method in percent.

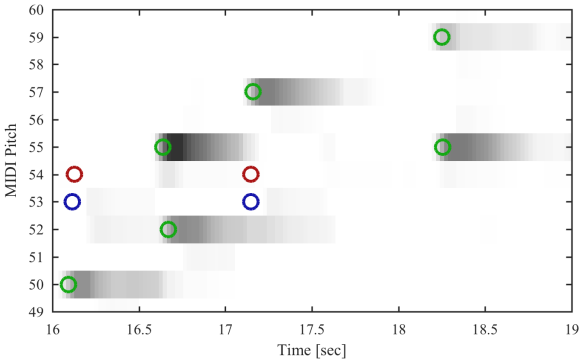


Figure 4. Cause of errors in piece 6: Activation matrix with ground truth annotations showing the position of notes in the 'correct', 'extra' and 'missing' classes.

in step 2 cannot observe how these two pitches manifest in the recording and thus cannot learn a meaningful template. Yet, being in a direct neighbourhood, the dictionary extrapolation in step 3 will use the learnt templates for pitch 54 and 66 to derive templates for pitches 53 and 65. Thus, these templates, despite the harmonicity constraints which still lead to some enforced structure in the templates, do not well represent how pitches 53 and 65 actually manifest in the recording and thus the corresponding activations will typically be low. As a result the extra notes were not detected as such by our method. We illustrate these effects in Fig. 4, where a part of the final full-range activation matrix is shown in the background and the corresponding ground-truth annotations are plotted on top as coloured circles. It is clearly visible, that the activations for pitch 53 are well below the level for the other notes. Excluding piece 6 from the evaluation, we obtain an average F-measure of 82% for 'extra' notes.

Finally, we reproduce the evaluation results reported for the method proposed in [11] in Table 3. It should be noted, however, that the results are not directly comparable with the results in Table 2 as we modified the underlying ground truth annotations. However, some general observations might be possible. In particular, since the class of 'correct' notes is the biggest in numbers, the results for this class are roughly comparable. In terms of accuracy, the number of errors in this class is five times higher in [11] (6.8 errors vs 1.4 errors per 100 notes). In this context, we want to remark that the method presented in [11] relied on the availability of recordings of single notes for the instrument in use, in contrast to ours. The underlying reason for the difference in accuracy between the two methods could be that instead of post-processing a standard AMT method, our approach yields a transcription method optimized in each step using score information. This involves a different signal model using several templates with dedicated meaning per pitch, the use of score information to optimize the onset detection and the use of pitch-dependent detection thresholds. Since the number of notes in the 'extra' and 'missing' classes are lower, it might not be valid to draw conclusions here.

Class	C	E	M
Accuracy	93.2	60.5	49.2

Table 3. Results reported for the method proposed in [11]. Remark: Values are not directly comparable with the results shown in Table 2 due to using different ground truth annotations in the evaluation.

4. CONCLUSIONS

We presented a novel method for detecting deviations from a given score in the form of missing and extra notes in corresponding audio recordings. In contrast to previous methods, our approach employs the information provided by the score to adapt the transcription process from the start, yielding a method specialized in transcribing a specific recording and corresponding piece. Our method is inspired by techniques commonly used in score-informed source separation that learn a highly optimized dictionary of spectral templates to model the given recording. Our evaluation results showed a high F-measure for notes in the classes 'correct' and 'missing', and a good F-measure for the 'extra' class. Our error analysis for the latter indicated possible directions for improvements, in particular for the dictionary extrapolation step. Further it would be highly valuable to create new datasets to better understand the behaviour of score-informed transcription methods under more varying recording conditions and numbers of mistakes made.

Acknowledgements: This work was partly funded by EPSRC grant EP/L019981/1. The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institut für Integrierte Schaltungen IIS. Sandler acknowledges the support of the Royal Society as a recipient of a Wolfson Research Merit Award.

5. REFERENCES

[1] James A Moorer. On the transcription of musical sound by computer. *Computer Music Journal*, pages 32–38, 1977.

[2] Anssi P. Klapuri and Manuel Davy, editors. *Signal Processing Methods for Music Transcription*. Springer, New York, 2006.

[3] Masataka Goto. A real-time music-scene-description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication (ISCA Journal)*, 43(4):311–329, 2004.

[4] Graham E. Poliner and Daniel P.W. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007.

[5] Zhiyao Duan, Bryan Pardo, and Changshui Zhang. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):2121–2133, 2010.

[6] Emmanuel Vincent, Nancy Bertin, and Roland Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):528–537, 2010.

[7] Sebastian Böck and Markus Schedl. Polyphonic piano note transcription with recurrent neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 121–124, Kyoto, Japan, 2012.

[8] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, PP(99):1–1, 2016.

[9] Holger Kirchhoff, Simon Dixon, and Anssi Klapuri. Multi-template shift-variant non-negative matrix deconvolution for semi-automatic music transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 415–420, 2012.

[10] Sebastian Ewert and Mark Sandler. Piano transcription in the studio using an extensible alternating directions framework. *(to appear)*, 2016.

[11] Emmanouil Benetos, Anssi Klapuri, and Simon Dixon. Score-informed transcription for automatic piano tutoring. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, pages 2153–2157, 2012.

[12] Sebastian Ewert, Bryan Pardo, Meinard Müller, and Mark D. Plumbley. Score-informed source separation for musical audio recordings: An overview. *IEEE Signal Processing Magazine*, 31(3):116–124, May 2014.

[13] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[14] Andreas Arzt, Sebastian Böck, Sebastian Flossmann, Harald Frostel, Martin Gasser, and Gerhard Widmer. The complete classical music companion v0.9. In *Proceedings of the AES International Conference on Semantic Audio*, pages 133–137, London, UK, 18–20 2014.

[15] Meinard Müller and Daniel Appelt. Path-constrained partial music synchronization. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 65–68, Las Vegas, Nevada, USA, 2008.

[16] Katsutoshi Itoyama, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Instrument equalizer for query-by-example retrieval: Improving sound source separation based on integrated harmonic and inharmonic models. In *Proceedings of the International Conference for Music Information Retrieval (ISMIR)*, pages 133–138, Philadelphia, USA, 2008.

[17] Romain Hennequin, Bertrand David, and Roland Badeau. Score informed audio source separation using a parametric model of non-negative spectrogram. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 45–48, Prague, Czech Republic, 2011.

[18] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *Proceedings of the Neural Information Processing Systems (NIPS)*, pages 556–562, Denver, CO, USA, 2000.

[19] Andrzej Cichocki, Rafal Zdunek, and Anh Huy Phan. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*. John Wiley and Sons, 2009.

[20] Jonathan Driedger, Harald Grohganz, Thomas Prätzlich, Sebastian Ewert, and Meinard Müller. Score-informed audio decomposition and applications. In *Proceedings of the ACM International Conference on Multimedia (ACM-MM)*, pages 541–544, Barcelona, Spain, 2013.

[21] Emmanouil Benetos and Simon Dixon. Multiple-instrument polyphonic music transcription using a temporally constrained shift-invariant model. *Journal of the Acoustical Society of America*, 133(3):1727–1741, 2013.

[22] Sebastian Ewert, Mark D. Plumbley, and Mark Sandler. A dynamic programming variant of non-negative matrix deconvolution for the transcription of struck string instruments. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 569–573, Brisbane, Australia, 2015.

FEATURE LEARNING FOR CHORD RECOGNITION: THE DEEP CHROMA EXTRACTOR

Filip Korzeniowski and Gerhard Widmer
Department of Computational Perception, Johannes Kepler University Linz, Austria
filip.korzeniowski@jku.at

ABSTRACT

We explore frame-level audio feature learning for chord recognition using artificial neural networks. We present the argument that chroma vectors potentially hold enough information to model harmonic content of audio for chord recognition, but that standard chroma extractors compute too noisy features. This leads us to propose a learned chroma feature extractor based on artificial neural networks. It is trained to compute chroma features that encode harmonic information important for chord recognition, while being robust to irrelevant interferences. We achieve this by feeding the network an audio spectrum with context instead of a single frame as input. This way, the network can learn to selectively compensate noise and resolve harmonic ambiguities.

We compare the resulting features to hand-crafted ones by using a simple linear frame-wise classifier for chord recognition on various data sets. The results show that the learned feature extractor produces superior chroma vectors for chord recognition.

1. INTRODUCTION

Chord Recognition (CR) has been an active research field since its inception by Fujishima in 1999 [10]. Since then, researchers have explored many aspects of this field, and developed various systems to automatically extract chords from audio recordings of music (see [20] for a recent review). Chord recognition meets this great interest in the MIR (music information research) community because harmonic content is a descriptive mid-level feature of (Western) music that can be used directly (e.g. for creating lead sheets for musicians) and as basis for higher-level tasks such as cover song identification, key detection or harmonic analysis.

Most chord recognition systems follow a common pipeline of feature extraction, pattern matching, and chord sequence decoding (also called *post-filtering*) [7]. In this paper, we focus on the first step in this pipeline: feature extraction.

© Filip Korzeniowski and Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Filip Korzeniowski and Gerhard Widmer. "Feature Learning for Chord Recognition:
The Deep Chroma Extractor", 17th International Society for Music Information Retrieval Conference, 2016.

Two observations lead us to explore better features for chord recognition: (1) *The capabilities of chord models for pattern matching are limited.* In [7], Cho and Bello conclude that appropriate features largely redeem the benefits of complex chord models. (2) *The capabilities of post-filtering are limited.* As shown in [6, 7], post-filtering methods are useful because they enforce continuity of individual chords rather than providing information about chord transitions. Incorporating such information did not considerably improve recognition results in both studies. Chen et al. [6] also observed quantitatively that in popular music "chord progressions are less predictable than it seems", and thus knowing chord history does not greatly narrow the possibilities for the next chord. Given these apparent limitations of the pattern matching and post-filtering stages, it is not surprising that they only partly compensate the performance gap between features [7]. We therefore have to compute better features if we want to improve chord recognition.

In this paper, we take a step towards better features for chord recognition by introducing a data-driven approach to extract chromagrams that specifically encode content relevant to harmony. Our method learns to discard irrelevant information like percussive noise, overtones or timbral variations automatically from data. We argue that it is thus able to compensate a broader range of interferences than hand-crafted approaches.

2. CHROMAGRAMS

The most popular feature used for chord recognition is the *Chromagram*. A chromagram comprises a time-series of chroma vectors, which represent harmonic content at a specific time in the audio as $c \in \mathbb{R}^{12}$. Each c_i stands for a pitch class, and its value indicates the current saliency of the corresponding pitch class. Chroma vectors are computed by applying a filter bank to a time-frequency representation of the audio. This representation results from either a short-time Fourier transform (STFT) or a constant-q transform (CQT), the latter being more popular due to a finer frequency resolution in the lower frequency area.

Chromagrams are concise descriptors of harmony because they encode tone quality and neglect tone height. In theory, this limits their representational power: without octave information, one cannot distinguish e.g. chords that comprise the same pitch classes, but have a different bass note (like G vs. G/5, or A:sus2 vs. E:sus4). In prac-

tice, we can show that given chromagrams derived from ground truth annotations, using logistic regression we can recognise 97% of chords (reduced to major/minor) in the Beatles dataset. This result encourages us to create chroma features that contain harmony information, but are robust to spectral content that is harmonically irrelevant.

Chroma features are noisy in their basic formulation because they are affected by various interferences: musical instruments produce overtones in addition to the fundamental frequency; percussive instruments pollute the spectrogram with broadband frequency activations (e.g. snare drums) and/or pitch-like sounds (tom-toms, bass drums); different combinations of instruments (and different, possibly genre-dependent mixing techniques) create different timbres and thus increase variance [7, 20].

Researchers have developed and used an array of methods that mitigate these problems and extract cleaner chromagrams: Harmonic-percussive source separation can filter out broadband frequency responses of percussive instruments [22, 27], various methods tackle interferences caused by overtones [7, 19], while [21, 27] attempt to extract chromas robust to timbre. See [7] for a recent overview and evaluation of different methods for chroma extraction. Although these approaches improve the quality of extracted chromas, it is very difficult to hand-craft methods for all conceivable disturbances, even if we could name and quantify them.

The approaches mentioned above share a common limitation: they mostly operate on single feature frames. Single frames are often not enough to decide which frequencies salient in the spectrum are relevant to harmony and which are noise. This is usually countered by contextual aggregation such as moving mean/median filters or beat synchronisation, which are supposed to smooth out noisy frames. Since they operate only *after* computing the chromas, they address the symptoms (noisy frames) but do not tackle the cause (spectral content irrelevant to harmony). Also, [7] found that they blur chord boundaries and details in a signal and can impair results when combined with more complex chord models and post-filtering methods.

It is close to impossible to find the rules or formulas that define harmonic relevance of spectral content manually. We thus resort to the data-driven approach of deep learning. Deep learning was found to extract strong, hierarchical, discriminative features [1] in many domains. Deep learning based systems established new state-of-the-art results in computer vision [1], speech recognition, and MIR tasks such as beat detection [3], tempo estimation [4] or structural segmentation [28].

In this paper, we want to exploit the power of deep neural networks to compute harmonically relevant chroma features. The proposed chroma extractor learns to filter harmonically irrelevant spectral content from a *context of audio frames*. This way, we circumvent the necessity to temporally smooth the features by allowing the chroma extractor to use context information directly. Our method computes cleaner chromagrams while retaining their advantages of low dimensionality and intuitive interpretation.

[1] See https://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html for results on computer vision.

3. RELATED WORK

A number of works used neural networks in the context of chord recognition. Humphrey and Bello [14] applied Convolutional Neural Networks to classify major and minor chords end-to-end. Boulanger-Lewandowski et al. [5], and Sigtia et al. [24] explored Recurrent Neural Networks as a post-filtering method, where the former used a deep belief net, the latter a deep neural network as underlying feature extractor. All these approaches train their models to directly predict major and minor chords, and following [1], the hidden layers of these models learn a hierarchical, discriminative feature representation. However, since the models are trained to distinguish major/minor chords only, they consider other chord types (such as seventh, augmented, or suspended) mapped to major/minor as intra-class variation to be robust against, which will be reflected by the extracted internal features. These features might thus not be useful to recognise other chords.

We circumvent this by using chroma templates derived from chords as distributed (albeit incomplete) representation of chords. Instead of directly classifying a chord label, the network is required to compute the *chroma representation* of a chord given the corresponding spectrogram. We expect the network to learn which saliency in the spectrogram is responsible for a certain pitch class to be harmonically important, and compute higher values for the corresponding elements of the output chroma vector.

Approaches to directly learn a mapping from spectrogram to chroma include those by İzmirli and Dannenberg [29] and Chen et al. [6]. However, both learn only a linear transformation of the time-frequency representation, which limits the mapping's expressivity. Additionally, both base their mapping on a single frame, which comes with the disadvantages we outlined in the previous section.

In an alternative approach, Humphrey et al. apply deep learning methods to produce Tonnetz features from a spectrogram [15]. Using other features than the chromagram is a promising direction, and was also explored in [6] for bass notes. Most chord recognition systems however still use chromas, and more research is necessary to explore to which degree and under which circumstances Tonnetz features are favourable.

4. METHOD

To construct a robust chroma feature extractor, we use a deep neural network (DNN). DNNs consist of L hidden layers h_l of U_l computing units. These units compute values based on the results of the previous layer, such that

$$h_l(x) = \sigma_l \left(W_l \cdot h_{l-1}(x) + b_l \right), \qquad (1)$$

where x is the input to the net, $W_l \in \mathbb{R}^{U_l \times U_{l-1}}$ and $b_l \in \mathbb{R}^{U_l}$ are the weights and the bias of the l^{th} layer re-

spectively, and σ_l is a (usually non-linear) *activation function* applied point-wise.

We define two additional special layers: an *input layer* that is feeding values to h_1 as $h_0(x) = x$, with U_0 being the input's dimensionality; and an *output layer* h_{L+1} that takes the same form as shown in Eq. 1, but has a specific semantic purpose: it represents the output of the network, and thus its dimensionality U_{L+1} and activation function σ_{L+1} have to be set accordingly. [2]

The weights and biases constitute the model's parameters. They are trained in a supervised manner by gradient methods and error back-propagation in order to minimise the loss of the network's output. The loss function depends on the domain, but is generally some measure of difference between the current output and the desired output (e.g. mean squared error, categorical cross-entropy, etc.)

In the following, we describe how we compute the input to the DNN, the concrete DNN architecture and how it was trained.

4.1 Input Processing

We compute the time-frequency representation of the signal based on the magnitude of its STFT X. The STFT gives significantly worse results than the constant-q transform if used as basis for traditional chroma extractors, but we found in preliminary experiments that our model is not sensitive to this phenomenon. We use a frame size of 8192 with a hop size of 4410 at a sample rate of 44100 Hz. Then, we apply triangular filters to convert the linear frequency scale of the magnitude spectrogram to a logarithmic one in what we call the *quarter-tone spectrogram* $S = F^{\triangle}_{Log} \cdot |X|$, where $F^{\triangle}_{Log}$ is the filter bank. The quarter-tone spectrogram contains only bins corresponding to frequencies between 30 Hz and 5500 Hz and has 24 bins per octave. This results in a dimensionality of 178 bins. Finally, we apply a logarithmic compression such that $S_{log} = \log(1 + S)$, which we will call the *logarithmic quarter-tone spectrogram*. To be concise, we will refer to S_{Log} as "spectrogram" in rest of this paper.

Our model uses a context window around a target frame as input. Through systematic experiments on the validation folds (see Sec.5.1) we found a context window of ± 0.7 s to work best. Since we operate at 10 fps, we feed our network at each time 15 consecutive frames, which we will denote as *super-frame*.

4.2 Model

We define the model architecture and set the model's hyper-parameters based on validation performance in several preliminary experiments. Although a more systematic approach might reveal better configurations, we found that results do not vary by much once we reach a certain model complexity.

[2] For example, for a 3-class classification problem one would use 3 units in the output layer and a softmax activation function such that the network's output can be interpreted as probability distribution of classes given the data.

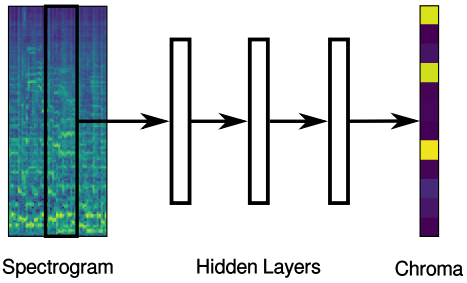


Figure 1. Model overview. At each time 15 consecutive frames of the input quarter-tone spectrogram S_{Log} are fed to a series of 3 dense layers of 512 rectifier units, and finally to a sigmoid output layer of 12 units (one per pitch class), which represents the chroma vector for the centre input frame.

Our model is a deep neural network with 3 hidden layers of 512 rectifier units [11] each. Thus, $\sigma_l(x) = \max(0, x)$ for $1 \leq l \leq L$. The output layer, representing the chroma vector, consists of 12 units (one unit per pitch class) with a sigmoid activation function $\sigma_{L+1}(x) = 1/1+\exp(-x)$. The input layer represents the input super-frame and thus has a dimensionality of 2670. Fig. 1 shows an overview of our model.

4.3 Training

To train the network, we propagate back through the network the gradient of the loss $\mathcal{L}$ with relation to the network parameters. Our loss is the binary cross-entropy between each pitch class in the predicted chroma vector $\mathbf{p} = h_{L+1}(S_{log})$ and the target chroma vector $\mathbf{t}$, which is derived from the ground truth chord label. For a single data instance,

$$\mathcal{L} = \frac{1}{12} \sum_{i=1}^{12} -t_i \log(p_i) - (1 - t_i) \log(1 - p_i). \quad (2)$$

We learn the parameters with mini-batch training (batch size 512) using the ADAM update rule [16]. We also tried simple stochastic gradient descent with Nesterov momentum and a number of manual learn rate schedules, but could not achieve better results (to the contrary, using ADAM training usually converged earlier). To prevent over-fitting, we apply dropout [26] with probability 0.5 after each hidden layer and early stopping if validation accuracy does not increase after 20 epochs.

5. EXPERIMENTS

To evaluate the chroma features our method produces, we set up a simple chord recognition task. We ignore any post-filtering methods and use a simple, linear classifier (logistic regression) to match features to chords. This way we want to isolate the effect of the feature on recognition accuracy. As it is common, we restrict ourselves to distinct only major/minor chords, resulting in 24 chord classes and a 'no chord' class.

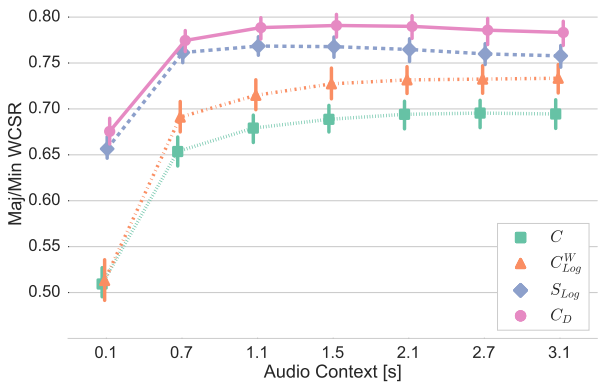


Figure 2. Validation WCSR for Major/minor chord recognition of different methods given different audio context sizes. Whiskers represent 0.95 confidence intervals.

Our compound evaluation dataset comprises the Beatles [13], Queen and Zweieck [18] datasets (which form the "Isophonics" dataset used in the MIREX [3] competition), the RWC pop dataset [4] [12], and the Robbie Williams dataset [8]. The datasets total 383 songs or approx. 21 hours and 39 minutes of music.

We perform 8-fold cross validation with random splits. For the Beatles dataset, we ensure that each fold has the same album distribution. For each test fold, we use six of the remaining folds for training and one for validation.

As evaluation measure, we compute the Weighted Chord Symbol Recall (WCSR), often called Weighted Average Overlap Ratio (WAOR) of major and minor chords using the mir_eval library [23].

5.1 Compared Features

We evaluate our extracted features C_D against three baselines: a standard chromagram C computed from a constant-q transform, a chromagram with frequency weighting and logarithmic compression of the underlying constant-q transform C^W_{Log}, and the quarter-tone spectrogram S_{Log}. The chromagrams are computed using the librosa library [5]. Their parametrisation follows closely the suggestions in [7], where C^W_{Log} was found to be the best chroma feature for chord recognition.

Each baseline can take advantage of context information. Instead of computing a running mean or median, we allow logistic regression to consider multiple frames of each feature [6]. This is a more general way to incorporate context, because running mean is a subset of the context aggregation functions possible in our setup. Since training logistic regression is a convex problem, the result is at least as good as if we used a running mean.

[3] http://www.music-ir.org/mirex

[4] Chord annotations available at https://github.com/tmc323/Chord-Annotations

[5] https://github.com/bmcfee/librosa

[6] Note that this description applies only to the baseline methods. For our DNN feature extractor, "context" means the amount of context the DNN sees. The logistic regression always sees only one frame of the feature the DNN computed.

	Btls	Iso	RWC	RW	Total
C	71.0±0.1	69.5 ±0.1	67.4±0.2	71.1±0.1	69.2±0.1
C^W_{Log}	76.0±0.1	74.2 ±0.1	70.3±0.3	74.4±0.2	73.0±0.1
S_{Log}	78.0±0.2	76.5 ±0.2	74.4±0.4	77.8±0.4	76.1±0.2
C_D	**80.2**±0.1	**79.3**±0.1	**77.3**±0.1	**80.1**±0.1	**78.8**±0.1

Table 1. Cross-validated WCSR on the Maj/min task of compared methods on various datasets. Best results are bold-faced ($p < 10^{-9}$). Small numbers indicate standard deviation over 10 experiments. "Btls" stands for the Beatles, "Iso" for Isophonics, and "RW" for the Robbie Williams datasets. Note that the Isophonics dataset comprises the Beatles, Queen and Zweieck datasets.

We determined the optimal amount of context for each baseline experimentally using the validation folds, as shown in Fig. 2. The best results achieved were 79.0% with 1.5 s context for C_D, 76.8% with 1.1 s context for S_{Log}, 73.3% with 3.1 s context for C^W_{Log}, and 69.5% with 2.7 s context for C. We fix these context lengths for testing.

6. RESULTS

Table 1 presents the results of our method compared to the baselines on several datasets. The chroma features C and C^W_{Log} achieve results comparable to those [7] reported on a slightly different compound dataset. Our proposed feature extractor C_D clearly performs best, with $p < 10^{-9}$ according to a paired t-test. The results indicate that the chroma vectors extracted by the proposed method are better suited for chord recognition than those computed by the baselines.

To our surprise, the raw quarter-tone spectrogram S_{Log} performed better than the chroma features. This indicates that computing chroma vectors in the traditional way mixes harmonically relevant features found in the time-frequency representation with irrelevant ones, and the final classifier cannot disentangle them. This raises the question of why chroma features are preferred to spectrograms in the first place. We speculate that the main reason is their much lower dimensionality and thus ease of modelling (e.g. using Gaussian mixtures).

Artificial neural networks often give good results, but it is difficult to understand what they learned, or on which basis they generate their output. In the following, we will try to dissect the proposed model, understand its workings, and see what it pays attention to. To this end, we compute saliency maps using guided back-propagation [25], adapting code freely available [7] for the Lasagne library [9]. Leaving out the technical details, a saliency map can be interpreted as an attention map of the same size as the input. The higher the absolute saliency at a specific input dimension, the stronger its influence on the output, where positive values indicate a direct relationship, negative values an indirect one.

Fig. 3 shows a saliency map and its corresponding super-frame, representing a C major chord. As expected,

[7] https://github.com/Lasagne/Recipes/

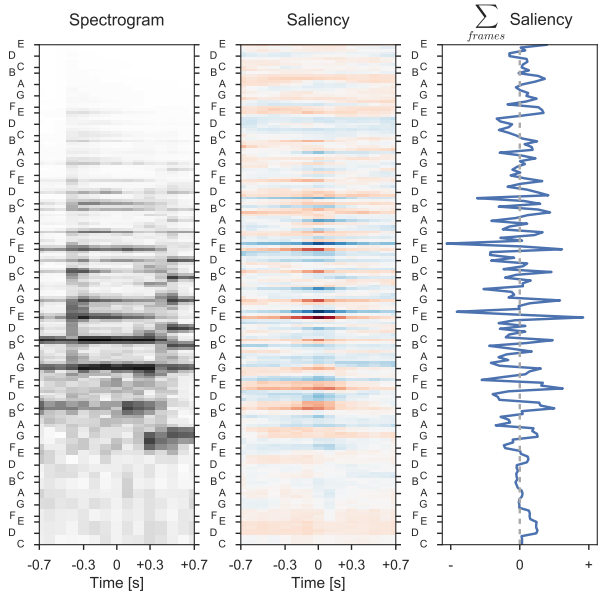


Figure 3. Input example (C major chord) with corresponding saliency map. The *left image* shows the spectrogram frames fed into the network. The *centre image* shows the corresponding saliency map, where red pixels represent positive, blue pixels negative values. The stronger the saturation, the higher the absolute value. The *right plot* shows the saliency summed over the time axis, and thus how each frequency bin influences the output. Note the strong positive influences of frequency bins corresponding to *c*, *e*, and *g* notes that form a C major chord.

the saliency map shows that the most relevant parts of the input are close to the target frame and in the mid frequencies. Here, frequency bins corresponding to notes contained in a C major chord (c, e, and g) showing positive saliency peeks, with the third, e, standing out as the strongest. Conversely, its neighbouring semitone, f, exhibits strong negative saliency values. Fig. 4 depicts average saliencies for two chords computed over the whole Beatles corpus.

Fig. 5 shows the average saliency map over all super-frames of the Beatles dataset summed over the frequency axis. It thus shows the magnitude with which individual frames in the super-frame contribute to the output of the neural network. We observe that most information is drawn from a ± 0.3 s window around the centre frame. This is in line with the results shown in Fig. 2, where the proposed method already performed well with 0.7 s of audio context.

Fig. 6 shows the average saliency map over all super-frames of the Beatles dataset, and its sum over the time axis. We observe that frequency bins below 110 Hz and above 3136 Hz (wide limits) are almost irrelevant, and that the net focuses mostly on the frequency range between 196 Hz and 1319 Hz (narrow limits). In informal experiments, we could confirm that recognition accuracy drops only marginally if we restrict the frequency range to the wide limits, but significantly if we restrict it to the narrow

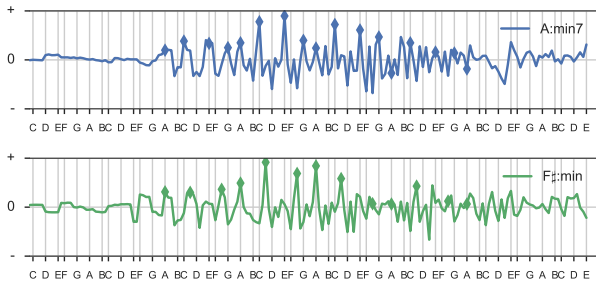


Figure 4. Average saliency map summed over the time axis for A:min7 and F♯:min chords computed on the Beatles dataset. As expected, we observe mostly positive peaks for frequency bins corresponding to notes present in the chords (a, c, e, g for A:min7; f♯, a, c♯ for F♯:min).

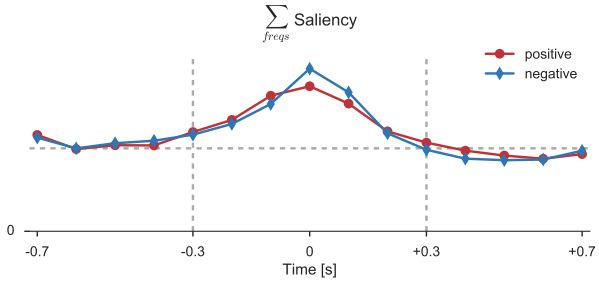


Figure 5. Average positive and negative saliencies of all input frames of the Beatles dataset, summed over the frequency axis. Most of the important information is within ± 0.3 s around the centre frame, and past data seems to be more important than future data. Around the centre frame, the network pays relatively more attention to what should be *missing* than present in a given chroma vector, and vice versa in areas further away from the centre. The differences are statistically significant due to the large number of samples.

limits. This means that the secondary information captured by the additional frequency bins of the wide limits is also crucial.

To allow for a visual comparison of the computed features, we depict different chromagrams for the song "Yesterday" by the Beatles in Fig. 7. The images show that the chroma vectors extracted by the proposed method are less noisy and chord transitions are crisper compared to the baseline methods.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a data-driven approach to learning a neural-network-based chroma extractor for chord recognition. The proposed extractor computes cleaner chromagrams than state-of-the-art baseline methods, which we showed quantitatively in a simple chord recognition experiment and examined qualitatively by visually comparing extracted chromagrams.

We inspected the learned model using saliency maps and found that a frequency range of 110 Hz to 3136 Hz

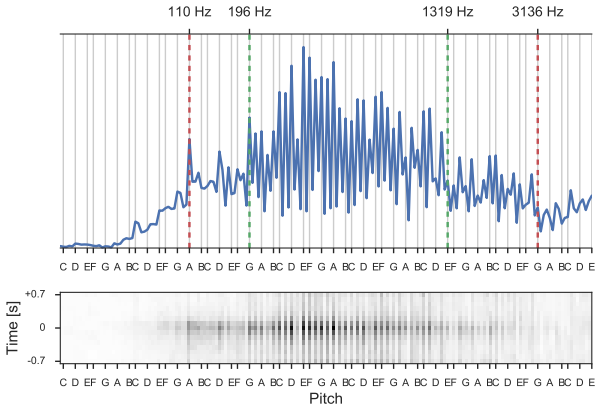


Figure 6. Average saliency of all input frames of the Beatles dataset (bottom image), summed over the time axis (top plot). We see that most relevant information can be collected in barely 3 octaves between G3 at 196 Hz and E6 at 1319 Hz. Hardly any harmonic information resides below 110 Hz and above 3136 Hz. The plot is spiky at frequency bins that correspond to clean semitones because most of the songs in the dataset seem to be tuned to a reference frequency of 440 Hz. The network thus usually pays little attention to the frequency bins between semitones.

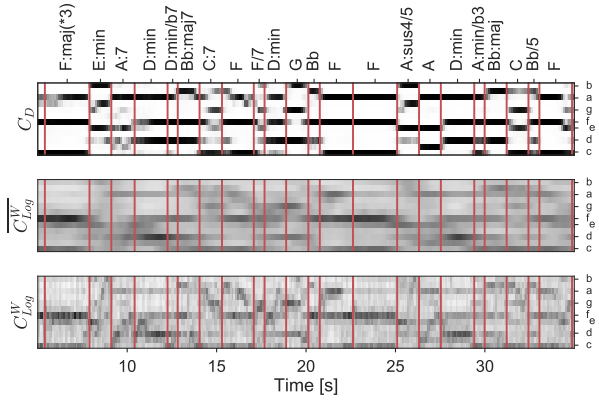


Figure 7. Excerpts of chromagrams extracted from the song "Yesterday" by the Beatles. The *lower image* shows chroma computed by the C^{W}_{Log} without smoothing. We see a good temporal resolution, but also noise. The centre image shows the same chromas after a moving average filter of 1.5 seconds. The filter reduced noise considerably, at the cost blurring chord transitions. The upper plot shows the chromagram extracted by our proposed method. It displays precise pitch activations and low noise, while keeping chord boundaries crisp. Pixel values are scaled such that for each image, the lowest value in the respective chromagram is mapped to white, the highest to black.

seems to suffice as input to chord recognition methods. Using saliency maps and preliminary experiments on validation folds we also found that a context of 1.5 seconds is adequate for local harmony estimation.

There are plenty possibilities for future work to extend and/or improve our method. To achieve better results, we could use DNN ensembles instead of a single DNN. We could ensure that the network sees data for which its predictions are wrong more often during training, or similarly, we could simulate a more balanced dataset by showing the net super-frames of rare chords more often. To further assess how useful the extracted features are for chord recognition, we shall investigate how well they interact with post-filtering methods; since the feature extractor is trained discriminatively, Conditional Random Fields [17] would be a natural choice.

Finally, we believe that the proposed method extracts features that are useful in any other MIR applications that use chroma features (e.g. structural segmentation, key estimation, cover song detection). To facilitate respective experiments, we provide source code for our method as part of the *madmom* audio processing framework [2]. Information and source code to reproduce our experiments can be found at http://www.cp.jku.at/people/korzeniowski/dc.

8. ACKNOWLEDGEMENTS

This work is supported by the European Research Council (ERC) under the EU's Horizon 2020 Framework Programme (ERC Grant Agreement number 670035, project "Con Espressione"). The Tesla K40 used for this research was donated by the NVIDIA Corporation.

9. REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, Aug. 2013.

[2] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer. madmom: a new Python Audio and Music Signal Processing Library. *arXiv preprint arXiv:1605.07008*, 2016.

[3] S. Böck, F. Krebs, and G. Widmer. A multi-model approach to beat tracking considering heterogeneous music styles. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, Taiwan, 2014.

[4] S. Böck, F. Krebs, and G. Widmer. Accurate tempo estimation based on recurrent neural networks and resonating comb filters. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, 2015.

[5] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Audio chord recognition with recurrent neural networks. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, 2013.

[6] R. Chen, W. Shen, A. Srinivasamurthy, and P. Chordia. Chord recognition using duration-explicit hidden Markov models. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, Porto, Portugal, 2012.

[7] T. Cho and J. P. Bello. On the Relative Importance of Individual Components of Chord Recognition Systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(2):477–492, Feb. 2014.

[8] B. Di Giorgi, M. Zanoni, A. Sarti, and S. Tubaro. Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony. In *Proceedings of the 8th International Workshop on Multidimensional Systems*, Erlangen, Germany, 2013.

[9] S. Dieleman, J. Schlüter, C. Raffel, E. Olson, S. K. Sønderby, D. Nouri, E. Battenberg, A. van den Oord, et al. Lasagne: First release, 2015.

[10] T. Fujishima. Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music. In *Proceedings of the International Computer Music Conference (ICMC)*, Beijing, China, 1999.

[11] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, USA, 2011.

[12] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC Music Database: Popular, Classical and Jazz Music Databases. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.

[13] C. Harte. *Towards Automatic Extraction of Harmony Information from Music Signals*. Dissertation, Department of Electronic Engineering, Queen Mary, University of London, London, United Kingdom, 2010.

[14] E. J. Humphrey and J. P. Bello. Rethinking Automatic Chord Recognition with Convolutional Neural Networks. In *11th International Conference on Machine Learning and Applications (ICMLA)*, Boca Raton, USA, 2012.

[15] E. J. Humphrey, T. Cho, and J. P. Bello. Learning a robust tonnetz-space transform for automatic chord recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Kyoto, Japan, 2012.

[16] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, Williamstown, USA, 2001.

[18] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. OMRAS2 metadata project 2009. In *Late Breaking Demo of the 10th International Conference on Music Information Retrieval (ISMIR)*, Kobe, Japan, 2009.

[19] M. Mauch and S. Dixon. Approximate note transcription for the improved identification of difficult chords. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, Utrecht, Netherlands, 2010.

[20] M. McVicar, R. Santos-Rodriguez, Y. Ni, and T. D. Bie. Automatic Chord Estimation from Audio: A Review of the State of the Art. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(2):556–575, Feb. 2014.

[21] M. Müller, S. Ewert, and S. Kreuzer. Making chroma features more robust to timbre changes. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Taipei, Taiwan, 2009.

[22] N. Ono, K. Miyamoto, J. Le Roux, H. Kameoka, and S. Sagayama. Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram. In *16th European Signal Processing Conference (EUSIPCO)*, Lausanne, France, 2008.

[23] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis. mir_eval: a transparent implementation of common MIR metrics. In *Proceedings of the 15th International Conference on Music Information Retrieval (ISMIR)*, Taipei, Taiwan, 2014.

[24] S. Sigtia, N. Boulanger-Lewandowski, and S. Dixon. Audio chord recognition with a hybrid recurrent neural network. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, 2015.

[25] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for Simplicity: The All Convolutional Net. *arXiv preprint arXiv:1412.6806*, 2014.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[27] Y. Ueda, Y. Uchiyama, T. Nishimoto, N. Ono, and S. Sagayama. HMM-based approach for automatic chord detection using refined acoustic features. In *International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Dallas, USA, Mar. 2010.

[28] K. Ullrich, J. Schlüter, and T. Grill. Boundary detection in music structure analysis using convolutional neural networks. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, Taiwan, 2014.

[29] Ö. İzmirli and R. B. Dannenberg. Understanding Features and Distance Functions for Music Sequence Alignment. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, Utrecht, Netherlands, 2010.

LEARNING TO PINPOINT SINGING VOICE FROM WEAKLY LABELED EXAMPLES

Jan Schlüter
Austrian Research Institute for Artificial Intelligence, Vienna
jan.schlueter@ofai.at

ABSTRACT

Building an instrument detector usually requires temporally accurate ground truth that is expensive to create. However, song-wise information on the presence of instruments is often easily available. In this work, we investigate how well we can train a singing voice detection system merely from song-wise annotations of vocal presence. Using convolutional neural networks, multiple-instance learning and saliency maps, we can not only detect singing voice in a test signal with a temporal accuracy close to the state-of-the-art, but also localize the spectral bins with precision and recall close to a recent source separation method. Our recipe may provide a basis for other sequence labeling tasks, for improving source separation or for inspecting neural networks trained on auditory spectrograms.

1. INTRODUCTION

A fundamental step in automated music understanding is to detect which instruments are present in a music audio recording, and at what time they are active. Traditionally, developing a system detecting and localizing a particular instrument requires a set of music pieces annotated at the same granularity as expected to be output by the system – no matter if the system is constructed by hand or by machine learning algorithms.

Annotating music pieces at high temporal accuracy requires skilled annotators and a lot of time. On the other hand, instrument annotations at a song level are often easily available online, as part of the tags given by users of streaming services, or descriptions or credits by the publisher. Even if not, collecting or cleaning song-wise annotations requires very little effort and low skill compared to curating annotations with sub-second granularity.

As a step towards tapping into such resources, in this work, we explore how to obtain high-granularity vocal detection results from low-granularity annotations. Specifically, we train a Convolutional Neural Network (CNN) on 10,000 30-second song snippets annotated as to whether they contain singing voice anywhere within, and subsequently use it to detect the presence of singing voice with sub-second granularity. As the main contribution of our work, we develop a recipe to improve initial results using multiple-instance learning and saliency maps. Finally, we investigate how well the system can even pinpoint the spectral bins containing singing voice, instead of the time frames only. While we constrain our experiments to singing voice detection as a special case of instrument detection (and possibly the easiest), we do not assume any prior knowledge about the content to be detected, and thus expect the recipe to carry over to other instruments.

The next section will provide a review of related work on learning from weakly-annotated data both outside and within of the music domain, and on singing voice detection. Section 3 explains the methods we combined in this paper, Section 4 describes how we combined them, and Section 5 evaluates the resulting system on four datasets. Finally, Section 6 discusses what we achieved and what is still open, highlights avenues for future research and points out alternative uses for some of our findings.

2. RELATED WORK

The idea of training on weakly-labeled data is far from new, since coarse labels are almost always easier to obtain than fine ones. The general framework for this setting is Multiple-Instance Learning, which we will return to in Section 3.2. As one of the first instances, Keeler et al. [8] train a CNN to recognize and localize two hand-written digits in an input image of about 36×36 pixels, giving only the identities of the two digits as training targets. As a recent work closer to our setting, Hou et al. [6] train a CNN to detect and classify brain tumors in gigapixel resolution tissue images. As such images are too large to be processed as a whole, they propose to train on patches, still using image-level labels only. To account for the fact that not all patches in a tumor image show tumorous tissue, Hou et al. employ an expectation maximization algorithm that iteratively prunes non-discriminative patches from the training set based on the CNN's predictions. As far as we are aware, the only work in music information retrieval aiming to produce fine-grained predictions from coarse training data is that of Mandel and Ellis [14]: They train special variants of SVMs on song, album or artist labels to predict tags on a granularity of 10-second clips. In contrast, we aim for sub-second granularity, and for identifying spectral bins.

© Jan Schlüter. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jan Schlüter. "Learning to Pinpoint Singing Voice From Weakly Labeled Examples", 17th International Society for Music Information Retrieval Conference, 2016.

Recent approaches for singing voice detection [9,10,18] are all based on machine learning from temporally accurate labels. The current state of the art is our previous work [18], a CNN trained on mel spectrogram excerpts. We will use it both as a starting point and for comparing our results against, and describe it in more detail in Section 3.1.

Singing voice detection does not entail identifying the spectral bins. The closest task to this is singing voice extraction, which aims to extract a purely vocal signal from a single-channel audio recording and thus has to estimate its spectral extends. It differs from general blind source separation in that it can leverage prior knowledge about the two signals to be separated – vocals and background music. As an improvement over Nonnegative Matrix Factorization (NMF) [20], which can only encode such prior knowledge in the form of spectral templates, REPET [16] uses the fact that background music is repetitive while vocals are not. Kernel-Additive Modeling [11] generalizes this method and uses a set of assumptions on local regularities of vocals and background music to perform singing voice extraction. We will use it as a mark to compare our results to.

3. INGREDIENTS

Our recipe combines a few methods that we shall introduce up front: a singing voice detector based on CNNs, multiple-instance learning, and saliency maps.

3.1 CNN-based Singing Voice Detection

The base of our system is a CNN trained to predict whether a short spectrogram excerpt contains singing voice at its center. We mostly follow Schlüter et al. [18], and will limit the description here to what is needed for understanding the paper, as well as how we deviated from that approach.

Input signals are converted to 22.05 kHz mono and processed by a Short-Time Fourier Transform with 70 1024-sample frames per second. Phases are discarded, magnitudes are scaled by $\log(1+x)$, the spectrogram is cropped above 8 kHz (keeping 372 bins) and each frequency band is normalized to zero mean and unit variance over the training set. We skip the mel-scaling step of Schlüter et al. to enable more accurate spectral localization of singing voice.

As in [18], the network architecture starts with 64 and 32 3×3 convolutions, 3×3 max-pooling, 128 and 64 3×3 convolutions. At this point, the 372 frequency bands have been reduced to 118. We add 128 3×115 convolutions and 1×4 max-pooling. This way, the network learns spectrotemporal patterns spanning almost the full frequency range, applies them at four different frequency offsets and keeps the maximum activation of those four, effectively introducing some pitch invariance. We finish with three dense layers of 256, 64 and 1 unit, respectively. Except for the final layer, each convolution and dense layer is followed by batch normalization [7] and the leaky rectifier $\max(x/100, x)$ [13]. The final layer uses the sigmoid activation function.

Training is done on excerpts of 115 frames (about 1.6 sec) paired with a binary label for the excerpt's central frame. We follow the training protocol described in [18, Sec. 3.3], augmenting inputs with random pitch-shifting and time-stretching of $\pm30\%$ and frequency filtering of ±10 dB using the code accompanying [18].

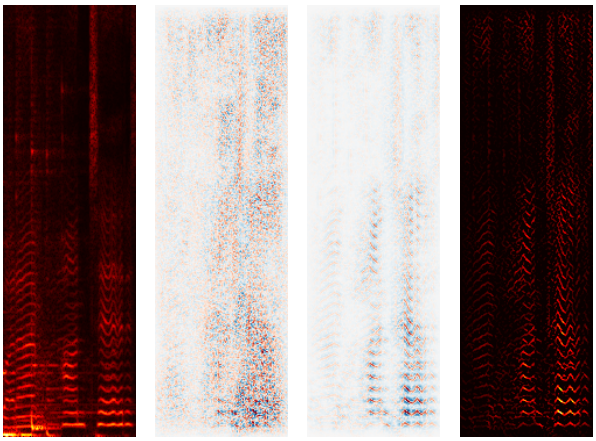
We arrived at this system by selectively modifying our previous work [18] to work well with linear-frequency spectrograms, on an internal dataset with fine-grained annotations. It slightly outperforms [18] on this dataset.

3.2 Multiple-Instance Learning

While we train our network on short spectrogram excerpts, we actually only have a single label per 30-second clip. In the Multiple-Instance Learning (MIL) framework, each explicitly labeled 30-second clip is called a *bag*, and the excerpts we train on are called *instances*. In our setting, a bag is labeled positively if and only if at least one of the instances contained within are positive (referred to as the *standard MI assumption* [4]). This gives an interesting asymmetry: If a 30-second clip is labeled as "no vocals", we can infer that neither of its excerpts contains vocals. If a clip contains vocals, we only know that *some* excerpts will contain vocals, but neither which ones nor how many.

One approach for training a neural network in this setting is based on the observation that the label of a bag is simply the maximum over its instance labels. If we define the network's prediction for a bag to be the maximum over the predictions for its instances, and the objective function to measure the discrepancy between this bag-wise prediction and the true label, minimizing it by gradient descent directly results in the following algorithm (BP-MIP, [24]): Propagate all instances of a bag through the network, pick the instance that gives the highest output, and update the network weights to minimize its discrepancy with the bag label. Unfortunately, this scheme is very costly: It computes predictions for all instances of a bag, then performs an update for a single instance only. Furthermore, it is easy to overfit: It is enough for the network to produce a strongly positive output for a single chosen instance per bag and negative outputs for all others. For 10,000 30-second clips, it would require merely learning 10,000 short excerpts by heart.

A different approach is to present all instances from negative bags as negative examples (we know they are negative) and all instances from positive bags as positive examples (they *could* be positive), and use a classifier that can underfit the training data, i.e., that may deviate from the training labels for some examples. This naive idea alone can produce good results, but it is also the basis for algorithms iteratively refining this starting point: The mi-SVM algorithm [1] uses the predictions of the initial classifier to re-label some instances from positive bags to become negative, and alternates between re-training and re-labeling until convergence. A variant proposed by Hou et al. [6] is to prune instances from positive bags that are not clearly positive, and also iterate until convergence. We will find that for our task, the idea of improving initial results by relabeling instances is important as well.



(a) (b) (c) (d)

Figure 1: Demonstration of saliency mapping: Network input (a), gradient (b), guided backpropagation (c) and its positive values (d). Best viewed on screen.

3.3 Saliency Mapping

Saliency maps for neural networks have been popularized by Zeiler et al. [23] as a means of inspecting how a trained neural network forms its decisions. One of the most elegant forms computes the saliency map as the gradient of the network's output [1] with respect to its input [19]. For a single data point, this tells how and in which direction each input feature influences the prediction for that data point. In our case, the input is a spectrogram excerpt and the gradient shows for each spectrogram bin how an infinitesimal increase would affect the probability of predicting singing voice (demonstrated in Figure 1a and 1b, respectively). Unfortunately, for a deep neural network, an input feature can influence the output in convoluted ways: Some input may increase the output by decreasing activities in hidden layers that are negatively connected to the output unit.

To get a clearer picture, Springenberg et al. [21] propose *guided backpropagation*: At each layer, only propagate the positive gradient values to the previous layer. This limits the saliency map to showing how input features affect the output by a chain of changes in the same direction. Figure 1c demonstrates this for our example. A positive value (displayed in red) for a bin means increasing this bin will increase the output *by increasing activities in all layers in between*. Likewise, a negative value (displayed in blue) for a bin means that increasing it will decrease the output.

Note that the negative saliencies are not very useful: They form "halos" around the positive saliencies, indicating that the network hinges on the local contrast, and they are much less sharply localized. Assuming the hidden layers show a similar picture, this explains why ignoring negative gradients in guided backpropagation gives a sharper saliency map. To obtain a map of spectrogram bins corresponding to what the network used to detect singing voice, we keep the positive saliencies only (Figure 1d).

[1] Precisely, the pre-activation of the output unit, before applying the nonlinearity, as the sigmoid would dampen gradients at high activations.

4. RECIPE

Having the basic ingredients in place, we will now describe our recipe. We begin by showing how to use a naively trained network for temporal detection and spectral localization, and then highlight an observation about saliency maps that enables higher precision. Finally, we give a three-step training procedure that further improves results.

4.1 Naive Training

The easiest solution to dealing with the problem of incomplete labels – and the starting point of our recipe – is to pretend the labels were complete. We train an initial network by presenting all excerpts from instrumental songs as negative, and all excerpts from vocal songs as positive. This already works quite well: even on the training data, it produces lower output for excerpts of vocal songs that do not contain voice than for those that do.

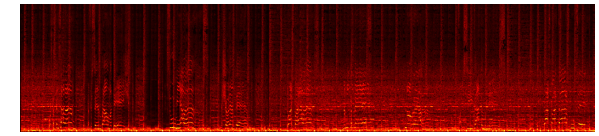
To obtain a temporal detection curve for a test song, we pass overlapping spectrogram excerpts of 115 frames through the network (with a hop size of 1 frame), recording each prediction. This way, for each spectrogram frame, we obtain a probability of singing voice being present in the surrounding ± 57 frames. As in [18], we post-process this curve by a sliding median filter of 56 frames (800 ms).

For spectral localization, we also pass overlapping spectrogram excerpts through the network, each time computing the 115×372-pixels saliency map for the excerpt. To combine these into a single map for a test song, we concatenate the central frames of the saliency maps. This gives a much sharper picture than an overlap-add of the excerpt maps. When used for vocal extraction, we apply two post-processing steps: As the saliency maps are very sparse, we apply Gaussian blurring with a standard dev. of 1 bin. And as the saliency values are very low, we scale them to a range comparable to the spectrogram and take the element-wise minimum of the scaled map and spectrogram.

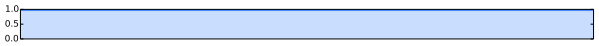
4.2 Overshoot Correction

When examining the predictions of the initial, naively trained network, we observe that it regularly overshoots at the boundaries of segments of singing voice. Figure 2 illustrates the problem for a training example: Predictions only decline far away from vocal parts, and short pauses are glossed over. This is an artifact from training on weak labels: The network predicts singing voice whenever its 1.6-second input excerpt contains vocals, even if only at the edge, because such excerpts have never been presented as negative examples during training.

The saliency map provides additional information, though. By computing the saliency of the central frame of each excerpt (Figure 2e), we can check whether it was important for that excerpt's prediction. Summing up the saliency map over frequencies (Figure 2f) gives an alternative prediction curve that can be used to improve the precision of temporal detection. In the next section, we will use this observation to improve the network.



(a) spectrogram of a 30-second training clip containing vocals

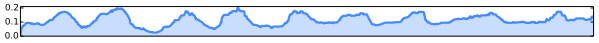


(b) weak labels for training (all positive)

(c) actual ground truth

(d) network predictions

(e) network saliency map



(f) network saliency map summarized over frequencies

Figure 2: Network predictions (d) overshoot vocal segments (c) because input windows only partially containing vocals were always presented as positive examples (b). Summarizing the saliency map (e) over frequencies (f) allows to correct such overshoots.

4.3 Self-Improvement

A basic idea we described in Section 3.2 is that the naively trained network could be used to relabel the positive training instances, which necessarily contain false positives (compare Figure 2b to 2c). The intuition is that correcting even just a few of those and re-training should improve results.

We tried several variants of this idea: Relabeling by thresholding the initial network's predictions (using a low threshold to only relabel very certainly false positives), weighting positive examples by the initial network's predictions (so confidently positive examples would affect training more than others), or removing positive examples the initial network is not confident about. However, the only effect was that the bias of the re-trained network was lower, and iterating such a scheme lead to a network predicting "no" all the time. In hindsight, this is not surprising: The only positive instances we relabel this way are those the network got correct with naive training already, so it will not learn anything new when re-training.

We found a single scheme that does not deteriorate results: Training a second network to output the temporally smoothed predictions of the initial network for positive instances, and the actual labels for negative instances. It does not result in better predictions either, but in much clearer saliency maps with less noise in non-vocal sections. Using the technique of Section 4.2, we find that for such a second network, the summarized saliency maps alone actually provide a better temporal detection curve than the network output, as it does not suffer from overshoot.

Iterating the latter scheme by re-training on the second network's predictions does not help. However, we can train a third network to output the temporally smoothed summarized saliency maps of the second network for positive instances, again keeping negative instances at their true labels. To bring them to a suitable range for training (i.e., between 0 and 1), we scale them by 10 and apply the $\tanh$ function. Finally, this third network gives predictions that do not need any overshoot correction. It can be seen as finding a more efficient way of computing the summarized saliency map of the second network.

Put together, our recipe consists of: (1) Training a first network (subsequently called CNN-α) on the weak instance labels, (2) training a second network (CNN-β) on the predictions of CNN-α, (3) training a third network (CNN-γ) on the summarized saliency maps of CNN-β.

5. EXPERIMENTS

To test our approach, we train networks on a dataset of 10,000 weakly-annotated 30-second snippets, then evaluate them on two public datasets for temporal detection and two public datasets for spectral localization.

5.1 Datasets

5.1.1 Training and Development

We collected 10,000 30-second song snippets of 10,000 artists. Using a custom web interface, 5 annotators sorted 2,000 snippets each into vocal and non-vocal ones, where "vocal" was defined to be "any use of human vocal chords". Annotators were allowed to skip examples they were very unsure about or that only contained voice-like sound effects, leaving 9751 annotated clips, 6772 of which contain vocals. To check inter-annotator agreement, we had 100 clips be labeled by 6 annotators. Of those, annotators agreed on 97 clips, the remaining 3 were skipped by at least one annotator and disagreed on by others.

We annotate the multiply-annotated clips with sub-second granularity to be used for testing, and keep the remaining clips for training. We further annotate 100 of the training clips finely to train a network equivalent to the one of Schlüter et al. [18] (dataset *In-House A* in that work).

The finely annotated positive clips feature vocals for 70% of the running time (between 13% and 99% per clip). Extrapolating, we thus expect CNN-α to be presented with 30% false positives among its positive training instances.

5.1.2 Testing

For testing temporal detection, we use two public datasets finely annotated with singing voice presence:

– *RWC*, collected by Goto et al. [5] and annotated by Mauch et al. [15], contains 100 pop songs.

– *Jamendo*, curated by Ramona et al. [17], contains 93 songs. It comes with a predefined train/test split, but we use all songs for testing.

	internal		RWC		Jamendo	
	AU-ROC	max acc.	AU-ROC	max acc.	AU-ROC	max acc.
CNN-α pred.	.911	.888	.879	.856	.913	.865
sal.	.955	.896	.912	.843	.930	.849
CNN-β pred.	.922	.888	.890	.861	.923	.875
sal.	.970	.916	.936	.883	.955	.894
CNN-γ pred.	.970	.915	.939	.887	.960	.901
sal.	.965	.914	.931	.884	.950	.898
CNN-fine pred.	.979	.930	.947	.882	.951	.880
sal.	.969	.909	.937	.883	.948	.885

Table 1: Temporal detection results for the three steps of self-improvement on weak labels (Section 4.3) as well as a network trained on fine labels, for three datasets.[2]

For spectral localization, we use two public datasets that come with separated vocal tracks:

– *ccMixter*, collected by Liutkus et al. [11], consists of 50 recordings from `ccmixter.org`.

– *MedleyDB*, compiled by Bittner et al. [2], contains 174 songs. We only use the 52 songs that feature vocals (singer or rapper) and do not have bleed between tracks. Downmixes are provided, we obtain the corresponding vocal tracks by mixing all vocal stems per song.

5.2 Temporal Detection Results

Singing voice detection is usually evaluated via the classification error at a particular threshold [9, 10, 15, 18]. However, different tasks may require different thresholds tuned towards higher precision or recall. Furthermore, tuning the threshold towards some goal requires a finely-annotated validation set, which we assume is not available in our setting. We therefore opt to assess the quality of the detection curves, rather than hard classifications. Specifically, we compute two measures: The Area Under the Receiver Operating Characteristic Curve (AUROC), and the classification accuracy at the optimal threshold. The former gives the probability that a randomly drawn positive example gets a higher prediction than a randomly drawn negative example, and the latter gives a lower bound on the error.

Table 1 shows the results. We compare four CNNs: CNN-α to CNN-γ from the three-step self-improvement scheme of Section 4.3, and CNN-fine, a network of the same architecture trained on 100 finely-annotated clips, as a reimplementation of the state-of-the-art by Schlüter et al. [18]. For each network, we assess the quality of its predictions and of its summarized saliency map, on the held-out part of our internal dataset as well as the two public datasets. [2]

We can see that for CNN-α, summarized saliencies are better than its direct predictions in terms of AUROC, but worse in terms of optimal classification error. CNN-β, which is trained on the predictions of CNN-α, performs strictly better than CNN-α and gains a lot when using summarized saliencies instead of predictions. CNN-γ is trained to predict the saliencies of CNN-β and matches or even outperforms those. Its saliency maps do not provide any benefit. Figure 3 provides a qualitative comparison of the predictions for the three networks.

Comparing results to CNN-fine, we see that it performs comparable to CNN-γ: It is strictly better on the internal test set (which is taken from the same source as the training data), but not on the public test sets, indicating that CNN-γ profits from its larger training set despite the weak labels. The summarized saliencies of CNN-fine do not provide any improvement, which was to be expected: There is no overshoot they could correct as in Section 4.2.

[2] Note that we did not train on the public datasets and used the full datasets for evaluation, so results are not directly comparable to literature.

	ccMixter			MedleyDB		
	prec.	rec.	F_1	prec.	rec.	F_1
baseline	.324	.947	.473	.247	.955	.361
KAML [12]	.597	.681	.627	.416	.739	.484
CNN-α	.575	.651	.603	.467	.637	.497
CNN-β	.565	.763	.643	.522	.618	.529
CNN- fine	.552	.795	.646	.494	.669	.528

Table 2: Spectral localization results for the baseline of just predicting the spectrogram of the mix, a voice/music separation method, and saliency maps of three networks.

5.3 Spectral Localization Results

Spectral localization of singing voice, i.e., identifying the spectrogram bins containing vocals, is not an established task. The closest to this is singing voice extraction, which is evaluated with standard source separation measures (Source to Distortion Ration, Source to Interference Ratio). However, developing our method into a full-fledged source separator is out of scope for this work. We therefore resort to comparing the saliency map produced by the network for a mixed signal to the spectrogram of the known pure-vocal signal. Specifically, we compute a generalization of precision, recall and F_1-measure towards real-valued (instead of binary) targets: For a predicted saliency map P_{ij} and pure-vocal spectrogram T_{ij}, we define the total amount of true positives as $t = \sum_{i,j} \min(P_{ij}, T_{ij})$. We then obtain precision as $p = t/\sum_{i,j} P_{ij}$, recall as $r = t/\sum_{i,j} T_{ij}$, and F_1-measure as $f = 2pr/(p+r)$.

Results are shown in Table 2. We first compute a simple baseline: Using the spectrogram of the mix as our vocal predictions. In theory, this should give 100% recall, but since the songs are mastered, the mix spectrogram is sometimes lower than the spectrogram of the vocal track, leaving a gap. We then obtain results for KAML [12], a refined implementation of KAM [11] described in Section 2. As the vocal prediction P_{ij}, we compute the spectrogram of the vocal signal it extracts, clipped to 8 kHz to be comparable to our network's saliency maps. Finally, we evaluate the saliency maps of CNN-α, CNN-β and CNN-fine, post-processing them as described in Section 4.1. We omit CNN-γ as it is tailored for temporal detection and will not produce better saliencies than CNN-β.

We find that all methods are comparably far from the baseline, with the CNNs obtaining higher F_1-measure than KAML. As in temporal detection, CNN-β has an edge over CNN-α, and is close to CNN-fine. While these results look promising, it should be noted that the saliency maps will not necessarily be better for source separation: A lot of the improvement in F_1-score hinges on the fact that the saliency maps are often perfectly silent in passages that do not contain vocals, while KAML still extracts parts of background instruments. To obtain high recall, for passages that do contain vocals, the post-processed saliency maps include more instrumental interference than KAML.

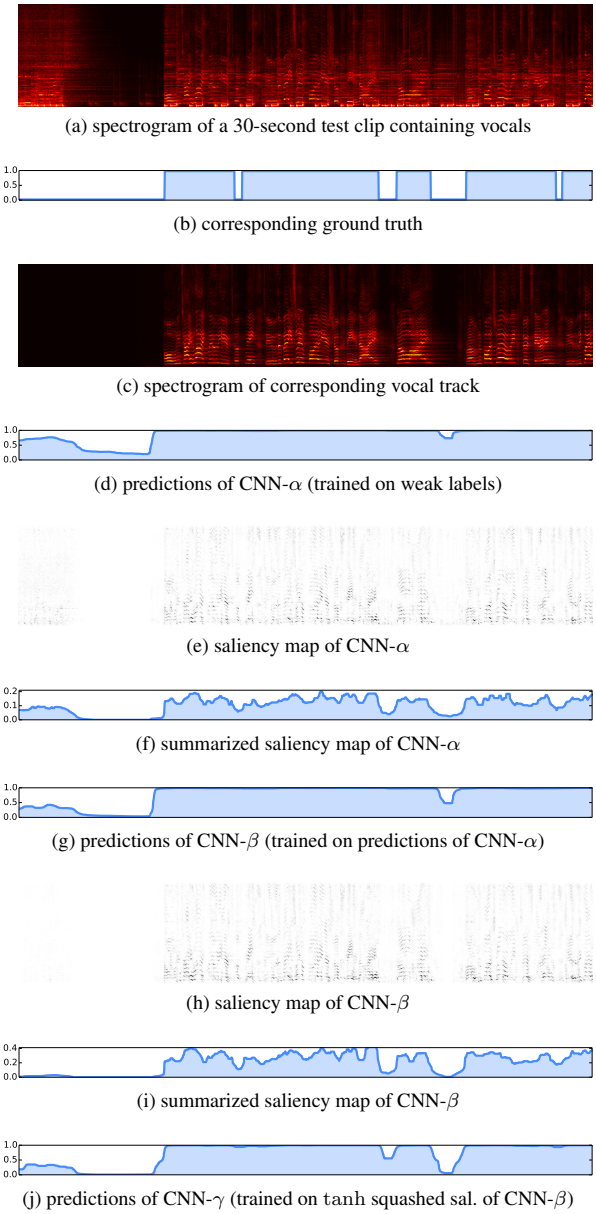
6. DISCUSSION

We have explored how to train CNNs for singing voice detection on coarsely annotated training data and still obtain temporally accurate predictions, closely matching performance of a network trained on finely annotated data. Furthermore, we have investigated a method for localizing the spectral bins that contain singing voice, without requiring according ground truth for training. We expect the recipe to carry over from human voice to musical instruments, if good contrasting examples are available – training on weakly-annotated data can only learn to distinguish instruments that occur independently from one another in different music pieces.

While our results are promising, there are a few shortcomings that provide opportunities for further research. For one, our networks produce good prediction curves, but when used for binary classification, we still need to choose a suitable threshold (e.g., optimizing accuracy, or a precision/recall tradeoff). We did not find good heuristics for selecting such a threshold solely based on weakly labeled examples. Secondly, comparing training on 10,000 weakly-labeled clips against 100 finely-labeled clips is clearly arbitrary. The main purpose in this work was to show that training from weakly-labeled clips can give high temporal accuracy *at all*, which is useful if such weak labels are easy to obtain. For future work, it would be interesting to compare the two methods on more even grounds. Specifically, we could investigate if weak labeling, fine labeling or a combination of both provides the best value for a given budget of annotator time.

The spectral localization results could be a starting point for instrument-specific source separation. But as for temporal detection, this route would first have to be compared on even grounds to learning from finely-annotated or source-separated training data, and its viability depends on how easily these types of data are obtainable. And in contrast to temporal detection, it requires further work to be turned into a source separation method.

Finally, the kind of saliency maps explored in this work could be used for other purposes: For example, it can be used to visualize and auralize precisely which content in a spectrogram was responsible for a particular false positive given by a network, and thus give a hint on how to enrich the training data to improve results.



(a) spectrogram of a 30-second test clip containing vocals

(b) corresponding ground truth

(c) spectrogram of corresponding vocal track

(d) predictions of CNN-α (trained on weak labels)

(e) saliency map of CNN-α

(f) summarized saliency map of CNN-α

(g) predictions of CNN-β (trained on predictions of CNN-α)

(h) saliency map of CNN-β

(i) summarized saliency map of CNN-β

(j) predictions of CNN-γ (trained on $\tanh$ squashed sal. of CNN-β)

Figure 3: Qualitative demonstration of the self-improvement recipe in Section 4.3 for a single test clip (0:24 to 0:54 of "Vermont" by "The Districts", part of the MedleyDB dataset [2]). Visit `http://ofai.at/~jan.schlueter/pubs/2016_ismir/` for an interactive version.

7. ACKNOWLEDGMENTS

The author would like to thank the anonymous reviewers for their valuable comments and suggestions. This research is funded by the Federal Ministry for Transport, Innovation & Technology (BMVIT) and the Austrian Science Fund (FWF): TRP 307-N23 and Z159. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of a Tesla K40 GPU used for this research. Finally, we thank the authors and co-developers of Theano [22] and Lasagne [3] the experiments build on.

8. REFERENCES

[1] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems 15*, pages 577–584. 2003.

[2] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello. MedleyDB: A multitrack dataset for annotation-intensive MIR research. In *Proc. of the 15th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, Taipei, Taiwan, Oct 2014.

[3] S. Dieleman, J. Schlüter, C. Raffel, E. Olson, S. K. Sønderby, D. Nouri, et al. Lasagne: First release., Aug 2015.

[4] J. Foulds and E. Frank. A review of multi-instance learning assumptions. *Knowledge Engineering Review*, 25(1):1–25, 2010.

[5] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical, and jazz music databases. In *Proc. of the 3rd Int. Conf. on Music Information Retrieval (ISMIR)*, pages 287–288, Oct 2002.

[6] L. Hou, D. Samaras, T. M. Kurç, Y. Gao, J. E. Davis, and J. H. Saltz. Efficient multiple instance convolutional neural networks for gigapixel resolution image classification. *CoRR*, abs/1504.07947v3, 2015.

[7] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. of the 32nd Int. Conf. on Machine Learning (ICML)*, 2015.

[8] J. D. Keeler, D. E. Rumelhart, and W. K. Leow. Integrated segmentation and recognition of hand-printed numerals. In *Advances in Neural Information Processing Systems 3*, pages 557–563. 1991.

[9] S. Leglaive, R. Hennequin, and R. Badeau. Singing voice detection with deep recurrent neural networks. In *Proc. of the 2015 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, Apr 2015.

[10] B. Lehner, G. Widmer, and S. Böck. A low-latency, real-time-capable singing voice detection method with LSTM recurrent neural networks. In *Proc. of the 23th European Signal Processing Conf. (EUSIPCO)*, Nice, France, 2015.

[11] A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet. Kernel additive models for source separation. *IEEE Transactions on Signal Processing*, 62(16):4298–4310, Aug 2014.

[12] A. Liutkus, D. Fitzgerald, and Z. Rafii. Scalable audio separation with light kernel additive modelling. In *Proc. of the 2015 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, Apr 2015.

[13] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. of the 30th Int. Conf. on Machine Learning (ICML)*, 2013.

[14] M. I. Mandel and D. P. W. Ellis. Multiple-instance learning for music information retrieval. In *Proc. of the 9th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, Philadelphia, USA, 2008.

[15] M. Mauch, H. Fujihara, K. Yoshii, and M. Goto. Timbre and melody features for the recognition of vocal activity and instrumental solos in polyphonic music. In *Proc. of the 12th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, 2011.

[16] Z. Rafii and B. Pardo. REpeating Pattern Extraction Technique (REPET): A simple method for music/voice separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(1):73–84, Jan 2013.

[17] M. Ramona, G. Richard, and B. David. Vocal detection in music with support vector machines. In *Proc. of the 2008 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1885–1888, 2008.

[18] J. Schlüter and T. Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *Proc of the 16th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, Malaga, Spain, 2015.

[19] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop of the 2nd Int. Conf. on Learning Representations (ICLR)*, Banff, Canada, 2014.

[20] P. Smaragdis and J. C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pages 177–180, Oct 2003.

[21] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *Workshop of the 3rd Int. Conf. on Learning Representations (ICLR)*, San Diego, USA, 2015.

[22] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.

[23] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proc. of the 13th European Conf. on Computer Vision (ECCV)*, pages 818–833, Zürich, Switzerland, 2014.

[24] Z.-H. Zhou and M.-L. Zhang. Neural networks for multi-instance learning. Technical report, Nanjing University, China, Aug 2002.

Poster Session 1

A CORPUS OF ANNOTATED IRISH TRADITIONAL DANCE MUSIC RECORDINGS: DESIGN AND BENCHMARK EVALUATIONS

Pierre Beauguitte, Bryan Duggan and John Kelleher
School of Computing, Dublin Institute of Technology
pierre.beauguitte@mydit.ie, bryan.duggan@dit.ie, john.d.kelleher@dit.ie

ABSTRACT

An emerging trend in music information retrieval (MIR) is the use of supervised machine learning to train automatic music transcription models. A prerequisite of adopting a machine learning methodology is the availability of annotated corpora. However, different genres of music have different characteristics and modelling these characteristics is an important part of creating state of the art MIR systems. Consequently, although some music corpora are available the use of these corpora is tied to the specific music genre, instrument type and recording context the corpus covers. This paper introduces the first corpus of annotations of audio recordings of Irish traditional dance music that covers multiple instrument types and both solo studio and live session recordings. We first discuss the considerations that motivated our design choices in developing the corpus. We then benchmark a number of automatic music transcription algorithms against the corpus.

1. INTRODUCTION

Automatic music transcription is one of the main challenges in MIR. The focus of most recent research is on polyphonic music, since it is sometimes claimed that the problem of transcribing a melody from a monophonic signal is solved [4]. The standard method in evaluating transcription algorithms is to take a data driven approach. This requires a corpus of data with original audio and ground truth annotations. Several such corpora exist, for example those provided for the annual MIREX evaluations. However no corpus can be universal, and the difficulty of gathering such a dataset is often discussed [20].

We are interested particularly in Irish traditional dance music and applications of automatic and accurate transcription in this genre. This musical tradition has several characteristics that are not captured by the existing datasets. Furthermore, to the best of our knowledge, there is no corpus of annotated Irish traditional dance music available. Since this is of absolute necessity in order to evaluate any new transcription method, we have created and released our own corpus. This paper presents its design and some baseline results for existing melody extraction algorithms.

The structure of the paper is as follows: Section 2 explains the rationale behind the creation of this corpus. We also detail the characteristics of the music we consider and the challenges it presents. Section 3 presents existing work related to our own. We then give the detailed design of our audio annotations dataset in Section 4. In Section 5, we evaluate four transcription algorithms on the dataset.

2. CHARACTERISTICS OF IRISH TRADITIONAL DANCE MUSIC

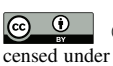
In common with similar, aurally transmitted musical traditions, Irish traditional dance music is subject to variations and ornamentation in its interpretation. The melodies, or *tunes*, are usually rather short and consist of two or sometimes more repeated parts.

The nature of the music is melodic and modal, as opposed to the more common harmonic and tonal aesthetics of classical and contemporary Western music. Most of the tunes are played on a D major or G major scale, but they can be in different keys including D ionian or mixolydian (major keys), E dorian or aeolian (minor keys) or G mixolydian (major key).

Many tunes were originally dance tunes, with a fixed rhythm, tempo and structure, though nowadays the tunes are mostly performed for listening rather than dancing. The most common rhythms are reels, with a 4/4 time signature, and jigs, with a 6/8 time signature. Other types of tunes include polkas, hornpipes, slides, barndances and airs. An in-depth presentation of Irish music can be found in [22].

Although in the past Irish music was performed mostly by solo instrumentalists, and this type of performance is still highly valued today, in contemporary practice it is commonly played by ensembles of musicians. On commercial recordings, the most common configuration involves several melodic instruments with rhythmic and harmonic accompaniment.

The environment in which this music is most commonly played is that of *sessions*: gatherings of musicians, professional or amateurs, of a usually informal nature. When played in a session, Irish music can often be adequately qualified as heterophonic [23]. All players of melodic instruments (typically greater in number than rhytmic and



© Pierre Beauguitte, Bryan Duggan and John Kelleher. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Pierre Beauguitte, Bryan Duggan and John Kelleher. "A corpus of annotated Irish traditional dance music recordings: design and benchmark evaluations", 17th International Society for Music Information Retrieval Conference, 2016.

harmonic instruments) play the same tune together, but the result is often far from unison for several reasons. First of all, different instruments can play the same melody in different octaves (*e.g.* flute and tin whistle). Additionally, due to the acoustic limitations of certain instruments, or as an intended variation, some notes of a tune can be played in a different octave. The low B (B3) for example cannot be played on most traditional flutes. Consequently flute players often play a B4 instead, while a banjo player would play the "correct" note B3 and a whistle player would play an octave higher than the flute, B5. Yet, all would be considered as playing the same tune.

Another important aspect is the amount of variations present in Irish music. Because of personal or regional stylistic differences, the abundance of different sources (notations, archive or commercial recordings), and of the music being transmitted aurally (thus relying on the memory of the musician and therefore subject to interpretation), many different versions of a same tune may exist. Although musicians will often try to adapt to each other in order to play a common version during a session, it is not uncommon to hear some differences in the melodies. Finally, tunes are almost always ornamented differently by each individual musician depending on their style and personal preferences.

Our goal with this project is to create a representative corpus of annotated audio data. We will then be able to establish a baseline on the performance of existing transcription algorithms for this particular music style. This will facilitate the development and evaluation of new melody extraction algorithms for Irish traditional dance music. We believe this is of great interest for such applications as accurate transcription of archive recordings, improvement of popular digital tools for musicians [9], and monitoring of contemporary music practice [21].

3. RELATED WORK

Automatic melody transcription algorithms have been designed specifically for *a cappella* flamenco singing in [10], and evaluated against manual annotations made by experts musicians. Although the article cites previous work related to the creation of a representative audio dataset, the manual ground truths annotations were created specifically for that project.

Turkish makam music has also been the subject of research, in particular with the SymbTr project [12], whose goal was to offer a comprehensive database of symbolic music. Applications of automatic transcription algorithms have been studied in [5]. Ground truth annotations were obtained by manually aligning the symbolic transcription from the SymbTr database to existing audio recordings. The paper points out the predominance of monophonic and polyphonic Eurogenetic music in Music Information Retrieval evaluations, and the challenges presented by music falling out of this range, such as heterophonic music.

Applications of automatic transcription algorithms to large scale World & Traditional music archives are proposed in [1]. More than twenty-nine thousand pieces of audio have been analysed. The obtained data have been made available through a Semantic Web server. No ground truth annotations were used to assess the quality of the transcriptions, that were obtained by the method presented in [3].

Manual and automatic annotations of commercial solo flute recordings of Irish traditional music have been conducted in [2], [11] and [13]. The focus in these papers is on style and automatic detection of ornaments in monophonic recordings. Consequently every ornament is finely transcribed in the annotations. An HMM-based transcription algorithm has been developed in [11] to recognise the different ornaments as well as the melody. An attempt was made at identifying players from these stylistic informations. The authors of [13] are planning on sharing the annotation corpus via the Semantic Web, but no data is publicly available yet.

4. PRESENTATION OF THE DATASET

We now describe the design of our corpus. First we present the set of audio recordings included, that was chosen to make the corpus representative of Irish traditional dance music. We then detail the annotation format used to deal with the characteristics of this genre.

4.1 Source of Audio Recordings

Three sources of recordings are included in the corpus:

- session recordings accompanying the Foinn Seisiún books published by the Comhaltas Ceoltóirí Éireann organisation, available with Creative Commons licence. These offer good quality, homogeneous examples of the heterophony inherent to an Irish traditional dance music session.
- Grey Larsen's *MP3s for 300 Gems of Irish Music for All Instruments*, commercially available. These consist of studio quality recordings of tunes played on Irish flute, tin whistle and anglo concertina.
- personal recordings of the second author, a renowned musician, on the Irish flute. These are available together with the annotations.

The corpus comprises thirty tunes in total, which add up to more than thirty minutes of audio. We chose to include solo recordings as a way of comparing the performance of transcription algorithms on monophonic and heterophonic music. This set of tunes has been chosen to be representative of the corpus of Irish music in terms of type and key signature. Table 1 categorises the tunes in our corpus by tune type, key, and performance type.

The complete list of tunes with all relevant metadata is included with the dataset.

4.2 Format of the Annotations

We annotated each audio recording with note events, consisting of three values: onset time, duration and pitch. For the larger goal of obtaining a symbolic transcription, this format of annotation is more useful as well as easier to

	Reel	Jig	Hornpipe	Polka	Slide	Air
Dmaj	2	3	1	1	0	0
Gmaj	2	2	1	2	1	0
Amin	2	2	1	1	0	0
Emin	2	2	1	0	1	1
Bmin	1	0	0	0	1	0
Session	5	5	1	1	1	0
Solo	4	4	3	3	2	1

Table 1: Classification of tunes in our corpus by tune type, key, and performance type

obtain than a continuous pitch track labelling every audio frame. Despite the heterophonic nature of the session performances, there is always a single underlying monophonic melody. It is this melody we are interetsed in. For this reason there is no overlap between the notes, and the resulting transcription we present is monophonic.

Due to the heterophonic nature of Irish traditional music as played during a session, and to the slight tuning differences between the instruments, a single fundamental frequency cannot appropriately describe a note. Therefore we decided to report only MIDI note references instead of frequencies.

In session performances, the playing of ornamentation such as *rolls* and *cuts* [22] often results in several successive onsets for a single long note. Figure 1 shows an example of such a situation, where three different instruments interpret differently the same note (the short notes played by the flute are part of a *roll* and are not melodically significant, therefore they are not transcribed). This makes it difficult, even for experienced musicians and listeners, to know when repeated notes are to be considered as a single ornamented note or distinct notes. Because of this inherent ambiguity, it is not correct to associate one onset with one note in the transcription. For this reason, we decided to merge consecutive notes of identical pitch into one single note. A note in our transcriptions then corresponds to a change of pitch in the melody. For solo performances, there are some clear silences between notes (typically when the flute or whistle player takes a breath). Whenever such a silence occurs, we annotated two distinct notes even if they are of the same pitch. In the solo recordings present in the corpus, a breath typically lasts around 200ms. Notes that are repeated without pauses or cut by an ornament were still reported as a single note, in order to be consistent with the annotations of session recordings.

Manual annotations were made by the first author with the aid of the Tony software [14]. After importing an audio file, Tony offers estimates using the pYIN algorithm (note-level) presented in the next section. These were then manually corrected, by adding new notes, merging repeated notes and adjusting frequencies. The annotations were finally post-processed to convert these frequencies to the closest MIDI note references. With this annotation format, the dataset comprises in total more than 8600 notes.

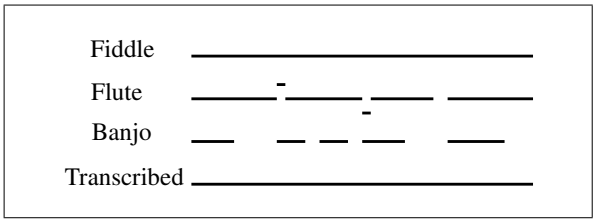


Figure 1: Example of ornamented notes on different instruments

4.3 Open Publication of the Dataset

The dataset is publicly available as a set of csv files. [1] Each file contains the annotation for an entire audio file. Each line represents a note (as onset time, duration, MIDI note). The annotations can be easily used in any evaluation framework as well as with the software Sonic Visualiser [6].

5. EVALUATION OF EXISTING TRANSCRIPTION ALGORITHMS

In order to establish some baselines for melody extraction in recordings of Irish traditional dance music, we evaluate transcriptions of our audio corpus returned by four different algorithms.

5.1 Presentation of the Algorithms

The melody extraction (or more generally automatic transcription) algorithms we use rely on different signal processing approaches, and return estimated melodies in two different formats. Some return frame-level estimates, or continuous pitch tracks, in which case some pre-processing detailed later in 5.2.1 is needed to conduct the evaluations. Others return note-level estimates of the same format as that used for our annotations, sometimes with a continuous pitch track as an intermediate step.

5.1.1 pYIN

pYIN [15] stands for probabilistic YIN, and is based on the standard frequency estimation algorithm YIN [7], used in conjunction with HMM-based pitch tracking. The initial algorithm returns frame-level estimates, but an additional segmentation step based on HMM modelling of note events was introduced in [14]. We evaluate the two versions of the algorithm, both open source and available as a Vamp plug-in. [2]

It is important to note that because we manually annotated our corpus with Tony, offering the note-level estimates from pYIN as first approximations, results of the evaluations might be biased in favour of this method.

5.1.2 Melodia

Melodia [19] first extracts a salience function by detecting peaks in the time/frequency representation of the au-

[1] https://github.com/pierrebeauguitte/tuneset
[2] https://code.soundsoftware.ac.uk/projects/pyin

dio signal and then extracts the best continuous pitch track possible. It is available as a Vamp plug-in. [3]

5.1.3 MATT

MATT [8] stands for Machine Annotation of Traditional Tunes. It returns note-level transcriptions, by estimating the fundamental frequency from the harmonicity of the signal, and segmenting the resulting continuous pitch track according to its continuity as well as the energy level. Although the method used is not at the state of the art of melody extraction algorithms (for an overview, see [18]), the fact that it was designed specifically for and fine-tuned to Irish traditional music makes it of great interest to us. A Java implementation is available online. [4]

5.1.4 Silvet

Silvet [3] is based on Principal Latent Component Analysis. Although it is designed for polyphonic music transcription, obtaining a single melody track is achievable by simply limiting the number of notes occuring at any time to one. It first generates a pitch track by factorising the spectrogram according to predefined templates. This is then post-processed with HMM smoothing, in a similar manner to the pYIN segmentation step. This approach has a much higher computationnal cost due to the complexity of spectrogram factorisation. It is available as an open source Vamp plug-in. [5]

5.2 Evaluations of the Transcriptions

In order to be consistent with our annotation policy (see 4.2), it is necessary to post-process the estimates of these algorithms in the following manner:

- align all frequencies to the closest MIDI note;
- merge consecutive notes of same pitch separated by less than 200ms (only for note-level estimates).

The second step is particularly critical for the note-level metrics of the MIREX Note Tracking task, but will also affect the frame-level metrics of the Melody Extraction tasks for the frames in the filled gaps.

All evaluations are performed with the `mir_eval` open source framework presented in [16]. [6] In order to assess the statistical significance of the differences in scores, we use the Wilcoxon signed rank test (to compare our samples with the performances reported in the original publications), and the Mann-Whitney U-test (to compare between our different samples).

5.2.1 Frame-Level Evaluation: Melody Extraction Task

The MIREX Melody Extraction task evaluates transcriptions at the frame-level. Pre-processing of both the ground truth and the estimates is necessary, and simply consists of aligning both on the same 10ms time grid. The pitch estimate for a frame is considered correct if it is distant from the ground truth by less than a quarter of a tone (50 cents). The metrics also look for voicing errors: a voiced frame is one where a melody pitch is present. Five different metrics are computed for each tune. Results are shown, using boxplots, in Figure 2, (a) for the solo recordings, and (b) for the session recordings.

The original publications introducing frame-level pYIN and MATT do not report these metrics. In [3], Silvet was only evaluated on corpora of polyphonic music, for which these metrics are not suitable. pYIN - notes was evaluated with the audio dataset from, and scores reported in [14] ranged from 0.83 to 0.85 for Overall Accuracy, and from 0.87 to 0.91 for Raw Pitch Accuracy. Evaluations conducted in [19] for Melodia on audio datasets from the MIREX evaluation resulted in Overall Accuracy of 0.77, Raw Chroma Accuracy of 0.83 and Raw Pitch Accuracy of 0.81. Scores obtained on our dataset are significantly lower for all these metrics and samples (p-values < 0.01), for both solo and session recordings. This seems to suggest that the genre of Irish traditional dance music does have some distinct characteristics, not limited to the heterophonic aspect of a session, that can present challenges for automatic transcription.

Comparing the Overall Accuracy on the solo and session subsets, it is interesting to see that MATT and both versions of pYIN have significantly lower scores on the session subset (p-values < 0.01), whereas Melodia and Silvet do not. We believe that this is because the Melodia and Silvet algorithms were specifically designed for polyphonic music analysis.

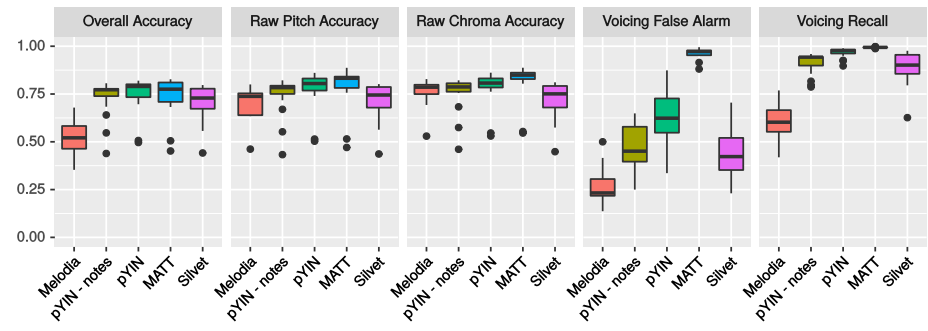
Raw chroma accuracy is, by definition, always higher than raw pitch accuracy. We observe that, on the solo recordings, this difference is only significant for Melodia (p-value < 0.05). On the session subset, it is very significant (p-values < 0.001) for all algorithms except Silvet. This suggests that Silvet is more robust for estimating fundamental frequencies.

5.2.2 Note-Level Evaluation: Note Tracking Task

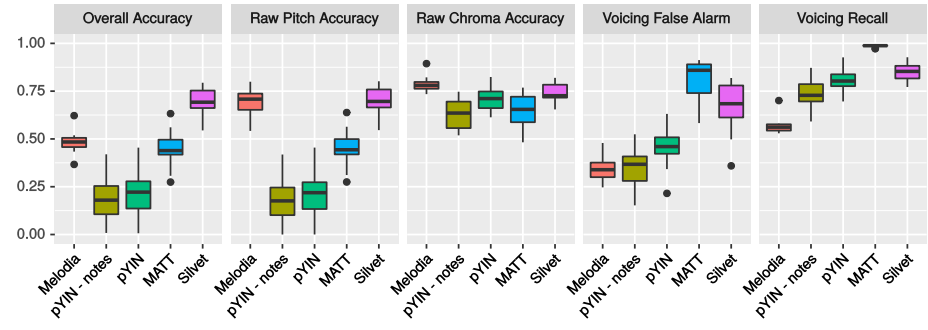
We now present the results of the MIREX Note Tracking task. Although this task is primarily aiming at polyphonic music transription systems, it also applies directly to monophonic music as long as both ground truth annotations and returned estimates are in a note-event format. In our case, this applies to pYIN - notes, MATT and Silvet.

Estimated notes are associated with their closest match from the ground truth, and a note is considered correctly transcribed if its onset is distant from the reference note by less than 50ms and its pitch by less than a quarter of a tone. Another way of evaluating the transcription is to also take the duration of notes into account. Commonly found instruments in Irish music have a wide range of acoustical characteristics: some (like the flute, the fiddle, the uilleann pipes) can be played *legato* or *staccato*, depending on personal or regional styles, or on the type of tunes performed; others (typically the banjo) can only be played *staccato*, with hard onsets and very little sustain. Consequently, the offset of the notes is of little interest for our evaluations,

[3] `http://mtg.upf.edu/technologies/melodia`
[4] `https://github.com/skooter500/matt2`
[5] `https://code.soundsoftware.ac.uk/projects/silvet`
[6] `https://github.com/craffel/mir_eval`



(a) Solo recordings



(b) Session recordings

Figure 2: Scores of the MIREX Melody Extraction task evaluation metrics.

particularly in session recordings where all these instruments can play together. This is why we only report results for the first type of evaluation.

Precision, recall and F-measures are computed with the `mir_eval` framework [16], and plotted in Figure 3, (a) for the solo recordings and (b) for the session recordings. The figures also show the results for the Chroma correctness, where a note is correctly transcribed if its onset ($\pm$50ms) and its pitch class are correct. This is of interest to us because of the heterophonic nature of session performance described in Section 2.

It is interesting to observe that MATT achieves the highest F-measures on the solo recordings. However, only the difference with Silvet is statistically significant (p-values < 0.05). On the session subset, Silvet performs better than the other two algorithms, with high statistical significance (p-values < 0.01).

Precision scores are not significantly different on the solo recordings. On the session recordings, Silvet achieves significantly higher pitch precisions (p-values < 0.001). However, when looking at the chroma precision, the difference with pYIN is no longer significant. Scores for MATT remain significantly lower (p-values < 0.001).

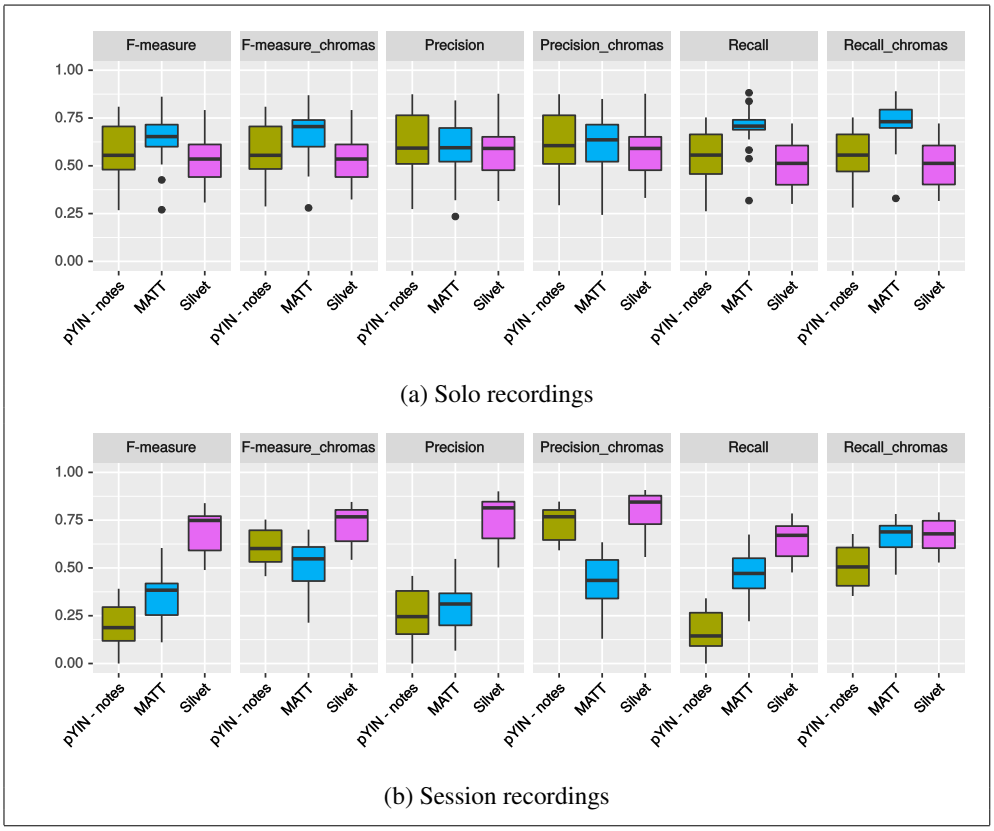
Surprisingly, Silvet has higher F-measures on the session subset than on the solo subset, and this difference is statistically significant (p-value < 0.05). pYIN and MATT both score lower on the session subset. For the pitch F-measure, this difference is highly significant, with p-values < 0.001. For the chroma F-measure, only MATT scores significantly lower (p-value < 0.01).

6. CONCLUSION

In this paper, we introduced a new dataset of annotations of Irish traditional dance music. It is, to our knowledge, the first publicly available corpus of manually transcribed audio recordings in this genre. It covers a representative range of tune types, keys, and performance types. From the results obtained with state of the art automatic transcription algorithms, it appears that the heterophonic nature of this music presents challenges for MIR. It would have been of great interest to evaluate the HMM based algorithm used for flute ornaments recognition in [11], but unfortunately no implementation of it is publicly available.

These findings are good motivation to work towards the development of new methods for automatically transcribing Irish traditional dance music. We believe that this corpus will be of great use for this purpose, both for training data-driven models and to evaluate new algorithms.

We hope to make the corpus larger in the future, so that it includes a wider range of instruments and performance types (violin or *fiddle*, banjo, session recordings from other sources). We are also planning on making use of the Music Ontology [17] in later releases. Adopting the standards of the Semantic Web will hopefully allow more interaction with many resources such as, for example, the Europeana-Sounds database.



(a) Solo recordings

(b) Session recordings

Figure 3: Scores of the MIREX Note Tracking task evaluation metrics.

7. REFERENCES

[1] Samer Abdallah, Aquiles Alencar-Brayner, Emmanouil Benetos, Stephen Cottrell, Jason Dykes, Nicolas Gold, Alexander Kachkaev, Mahendra Mahey, Dan Tidhar, Adam Tovell, Tillman Weyde, and Daniel Wolff. Automatic Transcription and Pitch Analysis of the British Library World and Traditional Music Collections. In *Proceedings of the 5th International Workshop on Folk Music Analysis*, pages 10–12, 2015.

[2] Islah Ali-MacLachlan, Münevver Köküer, Cham Athwal, and Peter Jančovič. Towards the Identification of Irish Traditional Flute Players from Commercial Recordings. In *Proceedings of the 5th International Workshop on Folk Music Analysis*, pages 13–17, 2015.

[3] Emmanouil Benetos and Simon Dixon. A Shift-Invariant Latent Variable Model for Automatic Music Transcription. *Computer Music Journal*, 36(4):81–94, December 2012.

[4] Emmanouil Benetos, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434, July 2013.

[5] Emmanouil Benetos and André Holzapfel. Automatic transcription of Turkish microtonal music. *The Journal of the Acoustical Society of America*, 138(4):2118–2130, 2015.

[6] Chris Cannam, Christian Landone, and Mark Sandler. Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1467–1468. ACM, 2010.

[7] Alain de Cheveigné and Hideki Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917, 2002.

[8] Bryan Duggan. *Machine annotation of traditional Irish dance music*. PhD thesis, Dublin Institute of Technology, June 2009.

[9] Bryan Duggan and Brendan O'Shea. Tunepal: searching a digital library of traditional music scores. *OCLC Systems & Services: International digital library perspectives*, 27(4):284–297, October 2011.

[10] Emilia Gómez and Jordi Bonada. Towards Computer-Assisted Flamenco Transcription: An Experimental Comparison of Automatic Transcription Algorithms As Applied to A Cappella Singing. *Computer Music Journal*, 37(2):73–90, 2013.

[11] Peter Jančovič, Münevver Köküer, and Baptiste Wrena. Automatic Transcription of Ornamented Irish

Traditional Flute Music Using Hidden Markov Models. In *Proceedings of the 16th International Society for Music Information Retrieval Conference*, 2015.

[12] M. Kemal Karaosmanoğlu. A Turkish makam music symbolic database for music information retrieval: SymbTr. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, 2012.

[13] Münevver Köküer, Daithí Kearney, Islah Ali-MacLachlan, Peter Jančovič, and Cham Athwal. Towards the creation of digital library content to study aspects of style in Irish traditional music. In *Proceedings of the 1st International Workshop on Digital Libraries for Musicology*, DLfM '14, pages 1–3. ACM Press, 2014.

[14] Matthias Mauch, Chris Cannam, Rachel Bittner, George Fazekas, Justin Salamon, Jiajie Dai, Juan Bello, and Simon Dixon. Computer-aided Melody Note Transcription Using the Tony Software: Accuracy and Efficiency. In *Proceedings of the First International Conference on Technologies for Music Notation and Representation*, 2015.

[15] Matthias Mauch and Sam Dixon. pYIN: A fundamental frequency estimator using probabilistic threshold distributions. In *Proceedings of the 39th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 659–663. IEEE, 2014.

[16] Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel PW Ellis. mir_eval: A Transparent Implementation Of Common MIR Metrics. In *Proceedings of the 15th International Society for Music Information Retrieval Conference*, 2014.

[17] Yves Raimond, Samer A. Abdallah, Mark B. Sandler, and Frederick Giasson. The Music Ontology. In *Proceedings of the 8th International Society for Music Information Retrieval Conference*, pages 417–422, 2007.

[18] Justin Salamon, Emilia Gomez, Daniel P.W. Ellis, and Guilhem Richard. Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *Signal Processing Magazine, IEEE*, 31(2):118–134, 2014.

[19] Justin Salamon and Emilia Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[20] Li Su and Yi-Hsuan Yang. Escaping from the Abyss of Manual Annotation: New Methodology of Building Polyphonic Datasets for Automatic Music Transcription. In *International Symposium on Computer Music Multidisciplinary Research*, 2015.

[21] Norman Makoto Su and Bryan Duggan. TuneTracker: tensions in the surveillance of traditional music. In *ACM conference on Designing Interactive Systems*, pages 845–854. ACM Press, 2014.

[22] Fintan Vallely, editor. *The Companion to Irish Traditional Music*. Cork University Press, Cork, second edition edition, 2011.

[23] Sean Williams. *Focus: Irish Traditional Music*. Routledge, 2013.

ADAPTIVE FREQUENCY NEURAL NETWORKS FOR DYNAMIC PULSE AND METRE PERCEPTION

Andrew J. Lambert, Tillman Weyde, and Newton Armstrong
City University London
{andrew.lambert.1; t.e.weyde; newton.arstrong.1}@city.ac.uk

ABSTRACT

Beat induction, the means by which humans listen to music and perceive a steady pulse, is achieved via a perceptual and cognitive process. Computationally modelling this phenomenon is an open problem, especially when processing expressive shaping of the music such as tempo change. To meet this challenge we propose Adaptive Frequency Neural Networks (AFNNs), an extension of Gradient Frequency Neural Networks (GFNNs).

GFNNs are based on neurodynamic models and have been applied successfully to a range of difficult music perception problems including those with syncopated and polyrhythmic stimuli. AFNNs extend GFNNs by applying a Hebbian learning rule to the oscillator frequencies. Thus the frequencies in an AFNN adapt to the stimulus through an attraction to local areas of resonance, and allow for a great dimensionality reduction in the network.

Where previous work with GFNNs has focused on frequency and amplitude responses, we also consider phase information as critical for pulse perception. Evaluating the time-based output, we find significantly improved responses of AFNNs compared to GFNNs to stimuli with both steady and varying pulse frequencies. This leads us to believe that AFNNs could replace the linear filtering methods commonly used in beat tracking and tempo estimation systems, and lead to more accurate methods.

1. INTRODUCTION

Automatically processing an audio signal to determine pulse event onset times (beat tracking) is a mature field, but it is by no means a solved problem. Analysis of beat tracking failures has shown that beat trackers have great problems with varying tempo and expressive timing [5, 6].

The neuro-cognitive model of nonlinear resonance models the way the nervous system resonates to auditory rhythms by representing a population of neurons as a canonical nonlinear oscillator [15]. A Gradient Frequency Neural Network (GFNN) is an oscillating neural network model based on nonlinear resonance. The network consists of a number of canonical oscillators distributed across a frequency range. The term 'gradient' is used to refer to this frequency distribution, and should not be confused with derivative-based learning methods in Machine Learning. GFNNs have been shown to predict beat induction behaviour from humans [16, 17]. The resonant response of the network adds rhythm-harmonic frequency information to the signal, and the GFNN's entrainment properties allow each oscillator to phase shift, resulting in deviations from their natural frequencies. This makes GFNNs good candidates for modelling the perception of temporal dynamics in music.

Previous work on utilising GFNNs in an MIR context has shown promising results for computationally difficult rhythms such as syncopated rhythms where the pulse frequency may be completely absent from the signal's spectrum [17, 23], and polyrhythms where there is more than one pulse candidate [2]. However, these studies have placed a focus on the frequencies contained in the GFNN's output, often reporting the results in the form of a magnitude spectrum, and thus omitting phase information. We believe that when dealing with pulse and metre perception, phase is an integral part as it constitutes the difference between entraining to on-beats, off-beats, or something in-between. In the literature, the evaluation of GFNNs' pulse finding predictions in terms of phase has, to our knowledge, never been attempted.

Our previous work has used GFNNs as part of a machine learning signal processing chain to perform rhythm and melody prediction. An expressive rhythm prediction experiment showed comparable accuracy to the state-of-the-art beat trackers. However, we also found that GFNNs can sometimes become noisy, especially when the pulse frequency fluctuates [12, 13].

This paper presents a novel variation on the GFNN, which we have named an Adaptive Frequency Neural Network (AFNN). In an AFNN, an additional Hebbian learning rule is applied to the oscillator frequencies in the network. Hebbian learning is a correlation-based learning observed in neural networks [9]. The frequencies adapt to the stimulus through an attraction to local areas of resonance. A secondary elasticity rule attracts the oscillator frequencies back to their original values. These two new interacting adaptive rules allow for a great reduction in the density of the network, minimising interference whilst also maintaining a frequency spread across the gradient.

© Andrew J. Lambert, Tillman Weyde, and Newton Armstrong. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Andrew J. Lambert, Tillman Weyde, and Newton Armstrong. "Adaptive Frequency Neural Networks for Dynamic Pulse and Metre Perception", 17th International Society for Music Information Retrieval Conference, 2016.

The results of an experiment with a GFNNs and AFNNs are also presented, partially reproducing the results from Velasco and Large's last major MIR application of a GFNN [23], and Large et al.'s more recent neuroscientific contribution [17]. However, we place greater evaluation focus on phase accuracy. We have found that AFNNs can produce a better response to stimuli with both steady and varying pulses.

The structure of this paper is as follows: Section 2 provides a brief overview of the beat-tracking literature and the GFNN model, Section 3 introduces a phase based evaluation method, Section 4 introduces our new AFNN model, Section 5 details the experiments we have conducted and shares the results, and finally Section 6 provides some conclusions and points to future work.

2. BACKGROUND

2.1 Pulse and Metre

Lerdahl and Jackendoff's *Generative Theory of Tonal Music* [19] was one of the first formal theories to put forward the notion of hierarchical structures in music which are not present in the music itself, but perceived and constructed by the listener. One such hierarchy is *metrical structure*, which are layers of beats existing in a hierarchically layered relationship with the rhythm. Each metrical level is associated with its own period, which divides the previous level's period into a certain number of parts.

Humans often choose a common, comfortable metrical level to tap along to, which is known as a *preference rule* in the theory. This common metrical level is commonly referred to as 'the beat', but this is a problematic term since a beat can also refer to a singular rhythmic event or a metrically inferred event. To avoid that ambiguity, we use the term 'pulse' [4].

2.2 Beat Tracking

Discovering the pulse within audio or symbolic data is known as *beat tracking* and has a long history of research dating back to 1990 [1]. There have been many varied approaches to beat tracking over the years, and here we focus on systems relevant to the proposed model. Some early work by Large used a single nonlinear oscillator to track beats in performed piano music [14]. Scheirer used linear comb filters [22], which operate on similar principles to Large and Kolen's early work on nonlinear resonance [18]. A comb filter's state is able to represent the rhythmic content directly, and can track tempo changes by only considering one metrical level. Klapuri et al.'s system builds on Scheirer's design by also using comb filters, and extends the model to three metrical levels [10]. More recently, Böck et al. [3] used resonating feed backward comb filters with a particular type of Recurrent Neural Network called a Long Short-Term Memory Network (LSTM) to achieve a state-of-the-art beat tracking result in the MIR Evaluation eXchange (MIREX) [1] .

[1] http://www.music-ir.org/mirex/

2.3 Nonlinear Resonance

Jones [7] proposed a psychological entrainment theory to address how humans are able to attend temporal events. Jones posited that rhythmic patterns such as music and speech potentially entrain a hierarchy of oscillations, forming an *attentional rhythm*. Thus, entrainment assumes an organisational role for temporal patterns and offers a prediction for future events, by extending the entrained period into the future.

Large then extended this theory with the notion of *nonlinear resonance* [15]. Musical structures occur at similar time scales to fundamental modes of brain dynamics, and cause the nervous system to resonate to the rhythmic patterns. Certain aspects of this resonance process can be described with the well-developed theories of neurodynamics, such as oscillation patterns in neural populations. Through the use of neurodynamics, Large moves between physiological and psychological levels of modelling, and directly links neural activity with music. Several musical phenomena can all arise as patterns of nervous system activation, including perceptions of pitch and timbre, feelings of stability and dissonance, and pulse and metre perception.

The model's basis is the canonical model of Hopf normal form oscillators, which was derived as a model oscillating neural populations [16]. Eqn (1) shows the differential equation that defines the canonical model, which is a Hopf normal form oscillator with its higher order terms fully expanded:

$$\frac{dz}{dt} = z(\alpha + i\omega + (\beta_1 + i\delta_1)|z|^2 + \frac{(\beta_2 + i\delta_2)\varepsilon|z|^4}{1-\varepsilon|z|^2}) + kP(\epsilon, x(t))A(\epsilon, \bar{z}) + \sum_{i \neq j} c_{ij}\frac{z_j}{1-\sqrt{\epsilon}z_j}\cdot\frac{1}{1-\sqrt{\epsilon}\bar{z_i}}, \quad (1)$$

z is a complex valued output where the real and imaginary parts represent excitation and inhibition, $\bar{z}$ is its complex conjugate, and ω is the driving frequency in radians per second. α is a linear damping parameter, and β_1, β_2 are amplitude compressing parameters, which increase stability in the model. δ_1, δ_2 are frequency detuning parameters, and ε controls the amount on nonlinearity in the system. $x(t)$ is a time-varying external stimulus, which is also coupled nonlinearly and consists of passive part, $P(\varepsilon, x(t))$, and an active part, $A(\varepsilon, \bar{z})$, controlled by a coupling parameter k. c_{ji} is a complex number representing phase and magnitude of a connection between the i^{th} and j^{th} oscillator (z_i, z_j). These connections can be strengthened through unsupervised Hebbian learning, or set to fixed values as in [23]. In our experiments presented here, we set c_{ij} to 0.

By varying the oscillator parameters, a wide range of behaviours not encountered in linear models can be induced (see [15]). In general, while the cannonical model maintains an oscillation according to its parameters, it entrains to and resonates with an external stimulus nonlinearly. The α parameter acts as a bifurcation parameter:

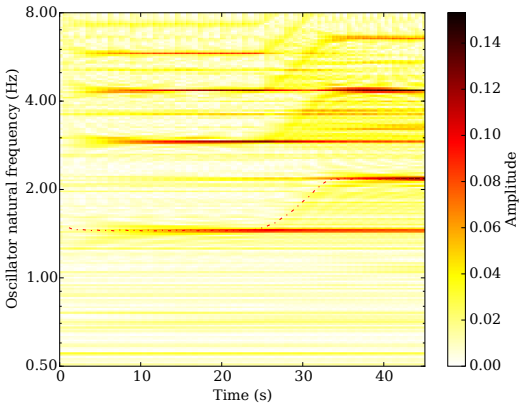


Figure 1. Amplitudes of oscillators over time. The dashed line shows stimulus frequency. The stimulus itself is shown in Figure 2. There is an accelerando after approximately 25s.

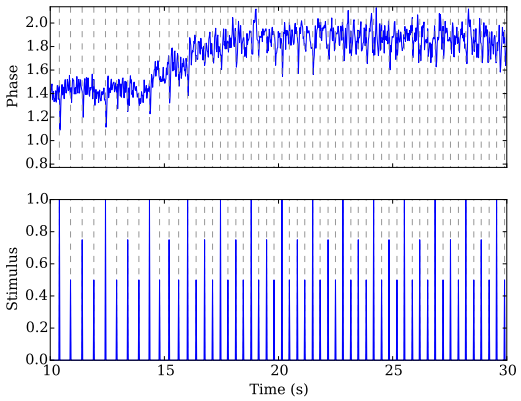


Figure 2. Weighted phase output, Φ, of the GFNN over time. The stimulus is the same as Figure 1.

when $\alpha < 0$ the model behaves as a damped oscillator, and when $\alpha > 0$ the model oscillates spontaneously, obeying a limit-cycle. In this mode, the oscillator is able to maintain a long temporal memory of previous stimulation.

Canonical oscillators will resonate to an external stimulus that contains frequencies at integer ratio relationships to its natural frequency. This is known as mode-locking, an abstraction on phase-locking in which k cycles of oscillation are locked to m cycles of the stimulus. Phase-locking occurs when $k = m = 1$, but in mode-locking several harmonic ratios are common such as 2:1, 1:2, 3:1, 1:3, 3:2, and 2:3 and even higher order integer ratios are possible [17], which all add harmonic frequency information to a signal. This sets nonlinear resonance apart from many linear filtering methods such as the resonating comb filters used in [10] and Kalman filters [8].

2.4 Gradient Frequency Neural Networks

Connecting several canonical oscillators together with a connection matrix forms a *Gradient Frequency Neural Network* (GFNN) [16]. When the frequencies in a GFNN are distributed within a rhythmic range and stimulated with music, resonances can occur at integer ratios to the pulse.

Figure 1 shows the amplitude response of a GFNN to a rhythmic stimulus over time. Darker areas represent stronger resonances, indicating that that frequency is relevant to the rhythm. A hierarchical structure can be seen to emerge from around 8 seconds, in relation to the pulse which is just below 2Hz in this example. At around 24 seconds, a tempo change occurs, which can be seen by the changing resonances in the figure. These resonances can be interpreted as a perception of the hierarchical metrical structure.

Velasco and Large [23] connected two GFNNs together in a pulse detection experiment for syncopated rhythms. The two networks were modelling the sensory and motor cortices of the brain respectively. In the first network, the oscillators were set to a bifucation point between damped and spontaneous oscillation ($\alpha = 0, \beta_1 = -1, \beta_2 = -0.25, \delta_1 = \delta_2 = 0$ and $\varepsilon = 1$). The second network was tuned to exhibit double limit cycle bifurcation behaviour ($\alpha = 0.3, \beta_1 = 1, \beta_2 = -1, \delta_1 = \delta_2 = 0$ and $\varepsilon = 1$), allowing for greater memory and threshold properties. The first network was stimulated by a rhythmic stimulus, and the second was driven by the first. Internal connections were set to integer ratio relationships such as 1:3 and 1:2, these connections were fixed and assumed to have been learned through a Hebbian process. The results showed that the predictions of the model confirm observations in human performance, implying that the brain may be adding frequency information to a signal to infer pulse and metre [17].

3. PHASE BASED EVALUATION

Thus far in the literature, evaluation of GFNNs has not considered phase information. The phase of oscillations is an important output of a GFNN; in relation to pulse it constitutes the difference between predicting at the correct pulse times, or in the worst-case predicting the off-beats. This is concerning in Velasco and Large's evaluation of pulse detection in syncopated beats, which by definition contain many off-beat events [23].

Phase and frequency are interlinked in that frequency can be expressed as a rate of phase change and indeed the canonical oscillators' entrainment properties are brought about by phase shifts. Since the state of a canonical oscillator is represented by a complex number, both amplitude and phase can be calculated instantaneously by taking the magnitude ($r = |z|$), and angle ($\varphi = arg(z)$) respectively. We propose calculating the weighted phase output, Φ, of the GFNN as a whole, shown in (2).

$$\Phi = \sum_{i=0}^{N} r_i \varphi_i \tag{2}$$

Figure 2 shows the weighted phase output, Φ, over time. Even though the amplitude response to the same stimu-

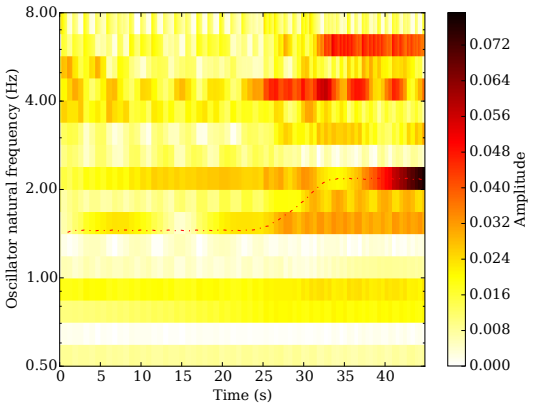


Figure 3. A low density (4opo) GFNN output. The dashed line shows stimulus frequency. Frequency information is not being captured as successfully, as can be observed by the low resonances.

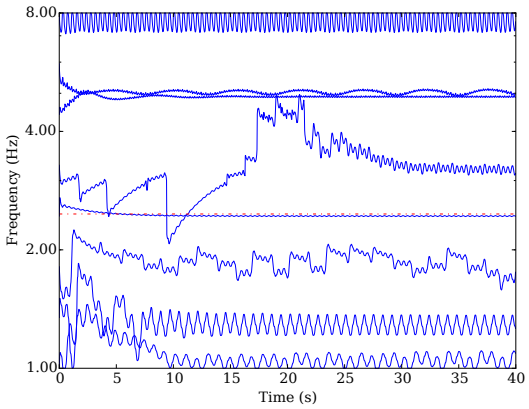


Figure 4. AFNN frequencies adapting to a sinusoidal stimulus. The dashed line shows stimulus frequency.

lus shows a clear corresponding metrical hierarchy (see Figure 1), the phase response remains noisy. This is due to the high density of oscillators required in a GFNN. Velasco and Large used 289 oscillators per layer in their experiment, a density of 48 oscillators per octave (opo). These high densities are often used in GFNNs to capture a wide range of frequencies, but can cause interference in the network. The term 'interference' is used here to mean interacting signals amplifying or cancelling each other when summed. Since each oscillator can only entrain to a narrow range of frequencies, using a lower density not only increases the likelihood missing a relevant frequency, it also stops local frequency populations from reinforcing one another. An example of this can be seen in Figure 3, where frequency information is not being captured as successfully in a 4opo GFNN.

In our previous work, we have addressed this issue by using only the real part of the oscillator as a single mean-field output [11, 12]. This retained a meaningful representation of the oscillation, but ultimately removed important information. A selective filter could also be applied, by comparing each oscillator with the mean amplitude of the GFNN, and only retaining resonating oscillators. However, this is not an ideal solution to the interference problem as it requires an additional, non real-time processing step which cannot be easily incorporated into an online machine learning chain. Furthermore, new frequencies would not be selected until they begin to resonate above the selection threshold, meaning that new resonances in changing tempos may be missed.

4. ADAPTIVE FREQUENCY NEURAL NETWORK

The Adaptive Frequency Neural Network (AFNN) attempts to address both the interference within high density GFNNs, and improve the GFNNs ability to track changing frequencies, by introducing a Hebbian learning rule on the frequencies in the network. This rule is an adapted form of the general model introduced by Righetti et al. [21] shown in (3):

$$\frac{d\omega}{dt} = -\frac{\epsilon}{r}x(t)sin(\varphi) \tag{3}$$

Their method depends on an external driving stimulus ($x(t)$) and the state of the oscillator (r, φ), driving the frequency (ω) toward the frequency of the stimulus. The frequency adaptation happens on a slower time scale than the rest of the system, and is influenced by the choice of ϵ, which can be thought of as a force scaling parameter. ϵ also scales with r, meaning that higher amplitudes are affected less by the rule.

This method differs from other adaptive models such as McAuley's phase-resetting model [20] by maintaining a biological plausibility ascribed to Hebbian learning [9]. It is also a general method that has been proven to be valid for limit cycles of any form and in any dimension, including the Hopf oscillators which form the basis of GFNNs (see [21]).

We have adapted this rule to also include a linear elasticity, shown in (4).

$$\frac{d\omega}{dt} = -\frac{\epsilon_f}{r}x(t)sin(\varphi) - \frac{\epsilon_h}{r}(\frac{\omega - \omega_0}{\omega_0}) \tag{4}$$

The elastic force is an implementation of Hooke's Law, which describes a force that strengthens with displacement. We have introduced this rule to ensure the AFNN retains a spread of frequencies (and thus metrical structure) across the gradient. The force is relative to natural frequency, and can be scaled through the ϵ_h parameter. By balancing the adaptive (ϵ_f) and elastic (ϵ_h) parameters, the oscillator frequency is able to entrain to a greater range of frequencies, whilst also returning to its natural frequency (ω_0) when the stimulus is removed. Figure 4 shows the frequencies adapting over time in the AFNN under sinusoidal input.

The AFNN preserves the architecture of the GFNN; the main difference is the frequency learning procedure. Figure 5 shows the weighted phase output (Φ) of an AFNN

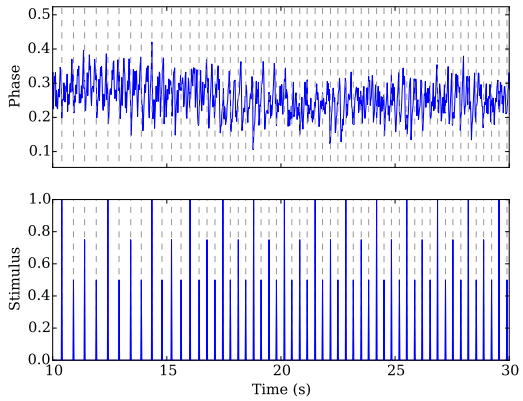


Figure 5. Weighted phase output, Φ, of the AFNN over time. Reduced interference can be seen compared with Figure 2.

stimulated with the same stimulus as in Figure 2. One can observe that a reduced level of interference is apparent.

5. EXPERIMENT

We have conducted a pulse detection experiment designed to test two aspects of the AFNN.

Firstly, we wanted to discover how the output of the AFNN compares with the GFNN presented in [23]. To this end, we are using similar oscillator parameters ($\alpha = 0, \beta_1 = \beta_2 = -1, \delta_1 = \delta_2 = 0$ and $\varepsilon = 1$). This is known as the 'critical' parameter regime, poised between damped and spontaneous oscillation. We are retaining their GFNN density of 48opo, but reducing the number of octaves to 4 (0.5-8Hz, logarithmically distributed), rather than the 6 octaves (0.25-16Hz) used in [23]. This equates to 193 oscillators in total. This reduction did not affect our results and is more in line with Large's later GFNN ranges (see [17]).

The AFNN uses the same oscillator parameters and distribution, but the density is reduced to 4opo, 16 oscillators in total. ϵ_f and ϵ_h were hand-tuned to the values of 1.0 and 0.3 respectively. For comparison with the AFNN, a low density GFNN is also included, with the same density as the AFNN but no adaptive frequencies.

We have selected two of the same rhythms used by Velasco and Large for use in this experiment, the first is an isochronous pulse and the second is the more difficult 'son clave' rhythm. We supplemented these with rhythms from the more recent Large et al. paper [17]. These rhythms are in varying levels of complexity (1-4), varied by manipulating the number of events falling on on-beats and off-beats. A level 1 rhythm contains one off-beat event, level 2 contains two off-beat events and so forth. For further information about these rhythms, see [17]. Two level 1 patterns, two level 2 patterns, two level 3 patterns, and four level 4 patterns were used.

The second purpose of the experiment was to test the AFNN and GFNN's performance on dynamic pulses, therefore we have included two additional stimulus rhythms: an accelerando and a ritardando.

We are additionally testing these rhythms at 20 different tempos selected randomly from a range 80-160bpm. None of the networks tested had any internal connections activated, fixed or otherwise ($c_{ij} = 0$). An experiment to study of the effect of connections is left for future work.

In summary, the experiment consisted of 5 stimulus categories, 20 tempos per category and 3 networks. There are two initial evaluations, one for comparison with previous work with GFNNs, and the second is testing dynamic pulses with accelerando and ritardando. The experiment used our own open-source *PyGFNN* python library, which contains GFNN and AFNN implementations [2] .

5.1 Evaluation

As we have argued above (see Section 2.4), we believe that when predicting pulse, phase is an important aspect to take into account. Phase information in the time domain also contains frequency information, as frequency equates the rate of change in phase. Therefore our evaluation compares the weighted phase output (Φ) with a ground truth phase signal similar to an inverted beat-pointer model [24]. While a beat-pointer model linearly falls from 1 to 0 over the duration of one beat, our inverted signal rises from 0 to 1 to represent phase growing from 0 to 2π in an oscillation. The entrainment behaviour of the canonical oscillators will cause phase shifts in the network, therefore the phase output should align to the phase of the input.

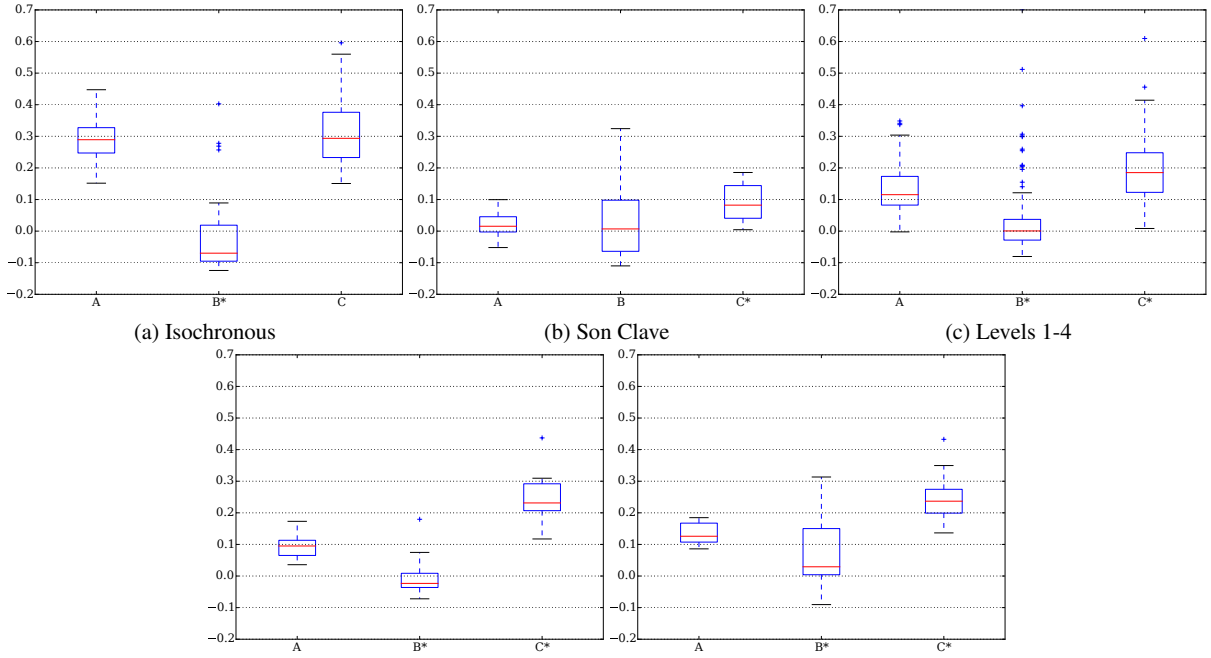
To make a quantitative comparison we calculate the Pearson product-moment correlation coefficient (PCC) of the two signals. This gives a relative, linear, mean-free measure of how close the target and output signals match. A value of 1 represents a perfect correlation, whereas -1 indicates an anti-phase relationship. Since the AFNN and GFNN operate on more than one metrical level, even a small positive correlation would be indicative of a good frequency and phase response, as some of the signal represents other metrical levels.

5.2 Results

Figure 6 shows the results for the pulse detection experiment described above in the form of box plots.

We can observe from Figure 6a that the GFNN (A) is effective for tracking isochronous rhythms. The resonance has enough strength to dominate the interference from the other oscillators. The low density GFNN (B) performs significantly worse with little positive correlation and some negative correlation, showing the importance of having a dense GFNN. The outliers seen can be explained by the randomised tempo; sometimes by chance the tempo falls into an entrainment basin of one or more oscillators. Despite its low density, the AFNN (C) fairs as well as the GFNN, showing a matching correlation to the target signal, especially in the upper quartile and maximum bounds. Exploring more values for ϵ_f and ϵ_h may yield even better results here.

[2] https://github.com/andyr0id/PyGFNN



(a) Isochronous (b) Son Clave (c) Levels 1-4

(d) Accel. (e) Rit.

Figure 6. Box and Whisker plots of the PCC results. A) GFNN, B) Low density GFNN, C) AFNN. Boxes represent the first and third quartiles, the red band is the median, and whiskers represent maximum and minimum non-outliers. *Denotes significance in a Wilcoxon signed rank test ($p < 0.05$).

In the son clave results (Figure 6b) all networks perform poorly. A poorer result here was expected due to the difficulty of this rhythm. However, we can see a significant improvement in the AFNN, which may be due to the reduced interference in the network. In the Large et al. rhythms (Figure 6c) we notice the same pattern.

We can see from the Accelerando and Ritardando rhythms (Figure 6d and 6e) that Φ is poorly correlated, indicating the affect of the interference from other oscillators in the system. The AFNNs response shows a significant improvement, but still has low minimum values. This may be due to the fact that the adaptive rule depends on the amplitude of the oscillator, and therefore a frequency change may not be picked up straight away. Changing the oscillator model parameters to introduce more amplitude damping may help here. Nevertheless the AFNN model still performs significantly better than the GFNN, with a much lower oscillator density.

6. CONCLUSIONS

In this paper we proposed a novel Adaptive Frequency Neural Network model (AFNN) that extends GFNNs with a Hebbian learning rule to the oscillator frequencies, attracting them to local areas of resonance. Where previous work with GFNNs focused on frequency and amplitude responses, we evaluated the outputs on their weighted phase response, considering that phase information is critical for pulse detection tasks. We conducted an experiment partially reproducing Velasco and Large's [23] and Large et al.'s [17] studies for comparison, adding two new rhythm categories for dynamic pulses. When compared with GFNNs, we showed an improved response by AFNNs to rhythmic stimuli with both steady and varying pulse frequencies.

AFNNs allow for a great reduction in the density of the network, which can improve the way the model can be used in tandem with other machine learning models, such as neural networks or classifiers. Furthermore the system functions fully online for use in real time. In future we would like to explore this possibility by implementing a complete beat-tracking system with an AFNN at its core.

We have a lot of exploration to do with regards to the GFNN/AFNN parameters, including the testing values for the adaptive frequency rule, oscillator models and internal connectivity. The outcome of this exploration may improve the results presented here.

The mode-locking to high order integer ratios, nonlinear response, and internal connectivity set GFNNs apart from many linear filtering methods such as the resonating comb filters and Kalman filters used in many signal prediction tasks. Coupled with frequency adaptation we believe that that the AFNN model provides very interesting prospects for applications in MIR and further afield. In future we would like to explore this possibility by implementing a complete beat-tracking system with an AFNN at its core and perform an evaluation with more realistic MIR datasets.

7. ACKNOWLEDGEMENTS

Andrew J. Lambert is supported by a PhD studentship from City University London.

8. REFERENCES

[1] Paul E. Allen and Roger B. Dannenberg. Tracking musical beats in real time. In *Proceedings of the 1990 International Computer Music Conference*, pages 140–3, San Fancisco, CA, 1990.

[2] Vassilis Angelis, Simon Holland, Paul J. Upton, and Martin Clayton. Testing a Computational Model of Rhythm Perception Using Polyrhythmic Stimuli. *Journal of New Music Research*, 42(1):47–60, 2013.

[3] Sebastian Böck, Florian Krebs, and Gerhard Widmer. Accurate tempo estimation based on recurrent neural networks and resonating comb filters. In *Proceedings of the 16th International Society for Music Information Retrieval Conference*, pages 625–31, Malaga, Spain, 2015.

[4] Simon Grondin. *Psychology of Time*. Emerald Group Publishing, 2008.

[5] Peter Grosche, Meinard Müller, and Craig Stuart Sapp. What Makes Beat Tracking Difficult? A Case Study on Chopin Mazurkas. In *in Proceedings of the 11th International Society for Music Information Retrieval Conference, 2010*, pages 649–54, Utrecht, Netherlands, 2010.

[6] Andre Holzapfel, Matthew E.P. Davies, José R. Zapata, João L. Oliveira, and Fabien Gouyon. Selective Sampling for Beat Tracking Evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(9):2539–48, 2012.

[7] Mari R. Jones. Time, our lost dimension: Toward a new theory of perception, attention, and memory. *Psychological Review*, 83(5):323–55, 1976.

[8] Rudolph E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[9] Richard Kempter, Wulfram Gerstner, and J Leo Van Hemmen. Hebbian learning and spiking neurons. *Physical Review E*, 59(4):4498–514, 1999.

[10] Anssi P. Klapuri, Antti J. Eronen, and Jaakko T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–55, 2006.

[11] Andrew Lambert, Tillman Weyde, and Newton Armstrong. Beyond the Beat: Towards Metre, Rhythm and Melody Modelling with Hybrid Oscillator Networks. In *Joint 40th International Computer Music Conference and 11th Sound & Music Computing conference*, Athens, Greece, 2014.

[12] Andrew Lambert, Tillman Weyde, and Newton Armstrong. Studying the Effect of Metre Perception on Rhythm and Melody Modelling with LSTMs. In *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2014.

[13] Andrew J. Lambert, Tillman Weyde, and Newton Armstrong. Perceiving and Predicting Expressive Rhythm with Recurrent Neural Networks. In *12th Sound & Music Computing Conference*, Maynooth, Ireland, 2015.

[14] Edward W. Large. Beat tracking with a nonlinear oscillator. In *Working Notes of the IJCAI-95 Workshop on Artificial Intelligence and Music*, pages 24–31, 1995.

[15] Edward W. Large. Neurodynamics of Music. In Mari R. Jones, Richard R. Fay, and Arthur N. Popper, editors, *Music Perception*, number 36 in Springer Handbook of Auditory Research, pages 201–231. Springer New York, 2010.

[16] Edward W. Large, Felix V. Almonte, and Marc J. Velasco. A canonical model for gradient frequency neural networks. *Physica D: Nonlinear Phenomena*, 239(12):905–11, 2010.

[17] Edward W. Large, Jorge A. Herrera, and Marc J. Velasco. Neural networks for beat perception in musical rhythm. *Frontiers in Systems Neuroscience*, 9(159), 2015.

[18] Edward W. Large and John F. Kolen. Resonance and the Perception of Musical Meter. *Connection Science*, 6(2-3):177–208, 1994.

[19] Fred Lerdahl and Ray Jackendoff. *A generative theory of tonal music*. MIT press, Cambridge, Mass., 1983.

[20] J. Devin McAuley. *Perception of time as phase: Toward an adaptive-oscillator model of rhythmic pattern processing*. PhD thesis, Indiana University Bloomington, 1995.

[21] Ludovic Righetti, Jonas Buchli, and Auke J. Ijspeert. Dynamic hebbian learning in adaptive frequency oscillators. *Physica D: Nonlinear Phenomena*, 216(2):269–81, 2006.

[22] Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. *The Journal of the Acoustical Society of America*, 103(1):588–601, 1998.

[23] Marc J. Velasco and Edward W. Large. Pulse Detection in Syncopated Rhythms using Neural Oscillators. In *12th International Society for Music Information Retrieval Conference*, pages 185–90, Miami, FL, 2011.

[24] Nick Whiteley, Ali T. Cemgil, and Simon J. Godsill. Bayesian Modelling of Temporal Structure in Musical Audio. In *Proceedings of the 7th International Society for Music Information Retrieval Conference*, pages 29–34, Victoria, Canada, 2006.

AN EVALUATION FRAMEWORK AND CASE STUDY FOR RHYTHMIC CONCATENATIVE SYNTHESIS

Cárthach Ó Nuanáin, Perfecto Herrera, Sergi Jordà
Music Technology Group
Universitat Pompeu Fabra
Barcelona
{carthach.onuanain, perfecto.herrera, sergi.jorda}@upf.edu

ABSTRACT

In this paper we present and report on a methodology for evaluating a creative MIR-based application of concatenative synthesis. After reviewing many existing applications of concatenative synthesis we have developed an application that specifically addresses loop-based rhythmic pattern generation. We describe how such a system could be evaluated with respect to its its objective retrieval performance and subjective responses of humans in a listener survey. Applying this evaluation strategy produced positive findings to help verify and validate the objectives of our system. We discuss the results of the evaluation and draw conclusions by contrasting the objective analysis with the subjective impressions of the users.

1. INTRODUCTION

MIR-based applications are becoming increasingly widespread in creative scenarios such as composition and performance [14] [7] [8]. This is commensurate with the prevalence of sampling-based approaches to sound generation, thus the desire is to develop more rich and descriptive understanding of the underlying content being used.

One of the primary difficulties faced with designing instruments for creative and compositional tasks remains the elaboration of an appropriate evaluation methodology. Indeed, this is a trending challenge facing many researchers [2], and numerous papers address this directly with various proposals for methodological frameworks, some drawing from the closely related field of HCI (Human Computer Interaction) [13], [16], [11]. More generally the evaluation of computer composition systems has also been the subject of much discussion in the literature. One frequent benchmark for evaluating algorithmic music systems is a type of Turing test where the success criterion is determined by the inability of human listener to discern between human and computer-generated music. As Hiraga [11] notes, however, these kind of tests can be problematic for two reasons. Firstly, it makes the assumption that the music generated by the algorithm is intended to sound like music produced by humans, rather than something to be treated differently. Secondly it ignores other facets of the system that imperatively needs evaluation, such as the interface and the *experience*. Pachet also finds issue with simplistic Turing test approaches to music evaluation [18]. He repeats, for instance, the view that unlike the traditional Turing test which evaluated the ability to synthesis believable natural language, no such "common-sense" knowledge exists for aspects of music.

We have designed and developed an MIR-driven instrument that uses concatenative synthesis to generate looped rhythmic material from existing content. In terms of evaluation we face the challenge of evaluating an MIR driven software system, thus subject to the same scrutiny facing any information retrieval system that needs to be appraised. We also face the challenge of evaluating the system as a musical composition system that needs to serve the composer and listener alike.

In the next section we will give the reader brief familiarity with the instrument in terms of its implementation and functionality. Subsequently, existing concatenative systems will be reported on in terms of their evaluation methodologies (if any). Section 3 will propose the evaluation framework in questions and the results will be reported. We will conclude the paper with our impressions on what we have learnt and scope for improvement in terms of the system itself and the evaluation methodology.

2. INSTRUMENT DESCRIPTION

Concatenative synthesis builds new sounds by combining together existing ones from a corpus. It is similar to granular synthesis differing only in the order of size of the grains: granular synthesis operates on microsound scales of 20-200ms whereas concatenative synthesis uses musically relevant unit sizes such as notes or even phrases. The process by which these sounds are selected for resynthesis is a fundamentally MIR-driven task. The corpus is defined by selecting sound samples, optionally segmenting them into onsets and extracting a chosen feature set to build descriptions of those sounds. New sounds can finally be synthesised by selecting sounds from the corpus according to

© Cárthach Ó Nuanáin, Perfecto Herrera, Sergi Jordà. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Cárthach Ó Nuanáin, Perfecto Herrera, Sergi Jordà. "An Evaluation Framework and Case Study for Rhythmic Concatenative Synthesis", 17th International Society for Music Information Retrieval Conference, 2016.

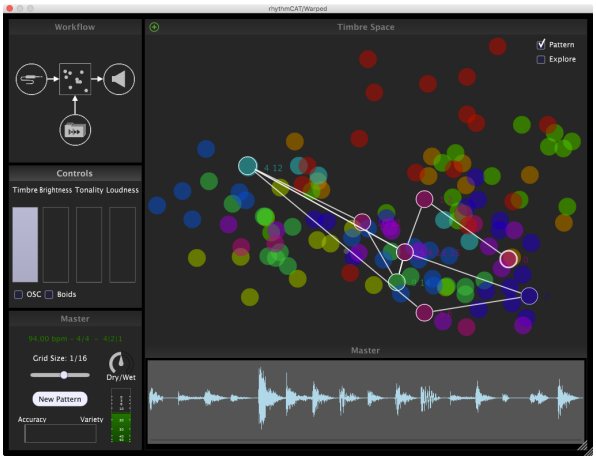


Figure 1: Instrument Interface

Author (Year)	Evaluation
Schwarz (2000)	No
Zils & Pachet (2001)	No
Hazel (2001)	No
Hoskinson & Pai (2001)	No
Xiang (2002)	No
Kobayashi (2003)	No
Cardle et al. (2003)	Videos of use cases
Lazier & Cook (2003)	No
Sturm (2004)	No
Casey (2005)	Retrieval Accuracy
Aucouturier & Pachet, (2005)	User Experiment
Simon et al. (2005)	Algorithm Performance
Jehan (2005)	Algorithmic evaluation
Schwarz (2005)	No
Weiss et al. (2009)	No
Frisson et al. (2010)	No
Hackbarth (2010)	No
Bernardes (2014)	Author's impressions

Table 1: Evaluation in Concatenative Systems

a unit selection algorithm and connecting them in series, maybe applying some cross-fading to smooth disparities in the process.

Concatenative synthesis has a long history of application in speech synthesis [15]. One of the most well-known works in the area of musical concatenative synthesis is CataRT [22] but there are many other systems referenced in the literature including some commercial implementations. Bernardes [3] provides a thorough summary of these based on similar reports in [25] and [21].

Our system (Figure 1) resembles many concatenative synthesis applications that offer a 2D timbre space for exploration. Where it distinguishes itself is in its sound generation strategy and mode of interaction for the user. Implemented as a VST plugin, it is an inherently loop-based instrument. It records and analyses incoming audio from the host as target segments according to a metrical level and concatenates units of sound from the corpus to generate new loops with varying degrees of similarity to the target loop. This similarity is determined by the unit selection algorithm, the central component in concatenative systems.

2.1 Unit Selection

The unit selection algorithm is quite straightforward. For each unit i in the segmented target sequence (e.g. 16-step) and each corpus unit j (typically many more), the concatenation *unit cost* $C_{i,j}$ is calculated by the weighted Euclidean distance of each feature k as given by Equation 1, where a and b are the values of the features in question.

$$C_{i,j} = \sqrt{\sum_{k=1}^{n} w_k (a_k - b_k)^2} \quad (1)$$

In terms of feature selection, after consulting a number of different articles [10], [20] and [27], dealing with feature extraction and rhythm we decided on a combination of MFCCs, loudness, spectral centroid and spectral flatness.

These unit costs are stored in similarity matrix M. Next we create a matrix M' of the indices of the ascendingly sorted elements of M. Finally a concatenated sequence can be generated by returning a vector of indices I from this sorted matrix and playing back the associated sound file. To retrieve the closest sequence V_0 one would only need to return the first row (Equation 3).

$$V_0 = (I_{0,i}, I_{0,i+1}..., I_{0,N}) \quad (2)$$

Returning sequence vectors solely based on the row restricts the possibilities to the number of rows in the matrix and is quite limited. We can extend the number of possibilities to i^{j-T} units if we define a similarity threshold T and return a random index between 0 and $j - T$ for each step i in the new sequence.

3. EVALUATION OF CONCATENATIVE SYNTHESIS

As we were researching existing works in the table presented by Bernardes, [3] we were struck by the absence of discussion regarding evaluation in most of the accompanying articles. This table we reproduce here (Table 1) amended and abridged with our details on the evaluation procedures (if any) that were carried out.

Some of the articles provided descriptions of use cases [4] or at least provided links to audio examples [24]. Notably, many of the articles [23], [9] consistently made references to the role of "user", but only one of those actually conducted a user experiment [1]. By no means is this intended to criticise the excellent work presented by these authors. Rather it is intended to highlight that although evaluation is not always an essential part of such experiments - especially in "one-off" designs for single users such as the author as composer - it is an underexplored aspect that could benefit from some contribution.

We can identify two key characteristics of our research that can inform what kind of evaluation can be carried out. Firstly it's a retrieval system, and can be analysed to determined its ability to retrieve relevant items correctly. Sec-

ondly it is a system that involves users or more precisely, musical users. How do we evaluate this crucial facet?

Coleman has identified and addressed the lack of subjective evaluation factors in concatenative synthesis [5]. In his doctoral thesis he devotes a chapter to a listening survey conducted to determine the quality of a number of different algorithmic components of the system under consideration. He asks the listeners to consider how well the harmony and timbre of the concatenated sequences are retained. In a previous paper [17] we conducted a similar-style listening survey to determine the ability of a genetic algorithm to create symbolic rhythmic patterns that also mimic a target input pattern. Evaluation strategies need to be tailored specifically for systems, but if the system is intended to retrieve items according to some similarity metric, and the material is musical, then a listening survey should be critical. Furthermore, and echoing Coleman's work, we would emphasise that whatever the application of a concatenative system, the evaluation of the *timbral* quality is essential.

In light of these elements we also devised a quantitative listening survey to examine musical output of the system not only in terms of its facility in matching the target content perceptually but also in producing musically pleasing and meaningful content.

4. METHOD

4.1 Evaluation and Experimental Design

Given the rhythmic characteristics of the system we formulated an experiment that evaluated its ability to generate new loops based on acoustic drum sounds. We gathered a dataset of 10 breakbeats ranging from 75 BPM to 142BPM and truncated them to single bar loops. Breakbeats are short drum solo sections from funk music records in the 1970s and exploited frequently as sample sources for hip-hop and electronic music. This practice has been of interest to the scientific community, as evident in work by Ravelli et al. [19], Hockman [12] and Collins [6].

In turn, each of these loops was then used as a seed loop for the system with the sound palette derived from the remaining 9 breakbeats. Four variations were then generated with 4 different distances to the target. These distances correspond to indices into the similarity matrix we alluded to in Section 2, which we normalise by dividing the index by the size of the table. The normalised distances then chosen were at 0.0 (the closest to the target), 1.0 (the furthest from the target) and two random distances in ranges 0.0 - 0.5 and 0.5 - 1.0.

Repeating this procedure 10 times for each target loop in the collection for each of the distance categories, we produced a total of 40 generated files to be compared with the target loop. Each step in the loop was labelled in terms of its drum content, for example the first step might have a kick and a hi-hat. Each segmented onset (a total of 126 audio samples) in the palette was similarly labelled with its corresponding drum sounds producing a total of 169 labelled sounds. The labellings we used were K = Kick, S = Snare, HH = Hi-hat, C = Cymbal and finally X when the content wasn't clear. Figure 2 shows the distribution of onsets by type in the full corpus of segmented units. Another useful statistic is highlighted in Figure 3, which plots the distribution of onsets for each step in the 16 step sequence for the predominant kick, snare and hi-hat for the 10 target loops. Natural trends are evident in these graphs, namely the concentration of the kick on the 1st beat, snares on the 2nd and 4th beat and hi-hats on off beats.

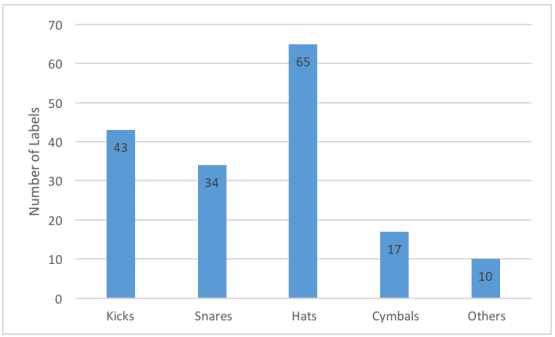


Figure 2: Distribution of Sounds in Corpus

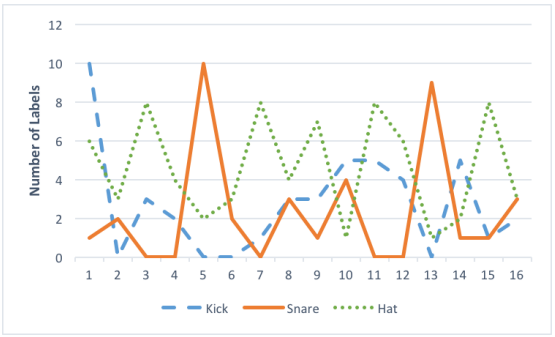


Figure 3: Distribution of Sounds in Target Loops

4.2 Retrieval Evaluation

The aim of the experiment was first to determine how well the system was able to retrieve similarly labelled "units" for each 1/16th step in the seed loop. To evaluate the ability of the algorithm to retrieve correctly labelled sounds in the generated loops we defined the accuracy A by equation 3, based on a similar approach presented in [26]. We make a simplification that corresponding HH and X and C labels also yield a match based on our observation that their noisy qualities are very similar, and some of the target loops used did not have onsets sounding at each 1/16th step.

$$A = \frac{number\ of\ correctly\ retrieved\ labels}{total\ number\ of\ labels\ in\ target\ loop} \quad (3)$$

4.2.1 Retrieval Results

By studying the Pearson's correlation between the retrieval ratings and the distance, we can confirm the tendency of smaller distances to produce more similar patterns by observing the moderate negative correlation (ρ = -0.516, p

<0.001) between increased distance and the accuracy ratings (Figure 4).

An interesting observation is that when we isolate the retrieval accuracy ratings to kick and snare we see this correlation increase sharply to (ρ = -0.826, p <0.001), as can be seen in Figure 5.

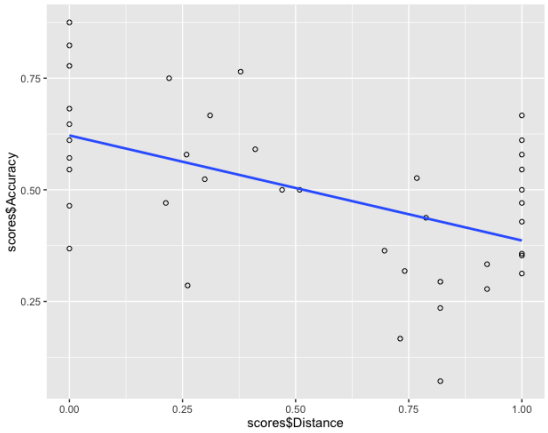


Figure 4: Scatter Plot and Regression Line of Retrieval Accuracy with Distance for all Drum Sounds

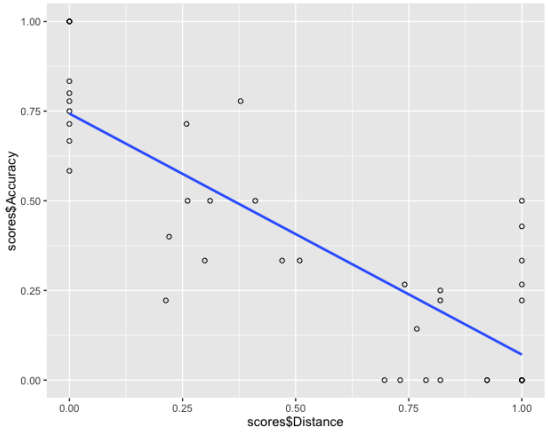


Figure 5: Scatter Plot and Regression Line of Retrieval Accuracy with Distance for Kick and Snare

Delving into the data further, we can identify 3 different categorical groupings that demonstrate predictable trends in terms of the central tendencies and descending retrieval accuracy (Figure 6). We label these categories A, B and C with the breakdown of the number of patterns and their corresponding distance ranges as follows:

- A - 10 patterns - 0.0
- B - 9 patterns - [0.2 - 0.5]
- C - 21 patterns - [0.5 - 1.0]

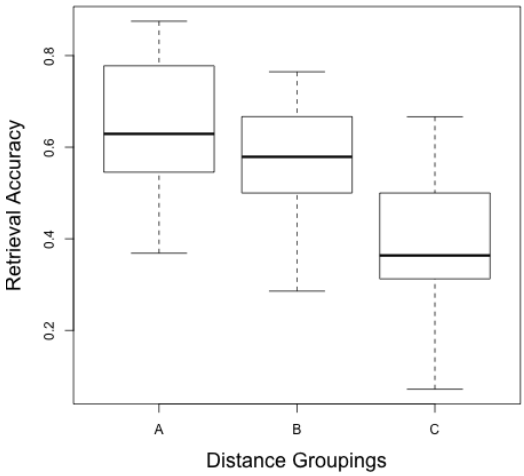


Figure 6: Central Tendencies of Retrieval Ratings for the Similarity/Distance Categories

4.3 Listener Evaluation

The retrieval accuracy gives the objective ratings of the system's capability for retrieving correctly labelled items. This information may not be consistent with the human listener's perceptual impression of similarity, nor does it give any indication whether the retrieved items are musically acceptable or pleasing. To assess the human experience of the sonic output and to compare with the objective ratings of the system, we conducted a listening survey which will be described here.

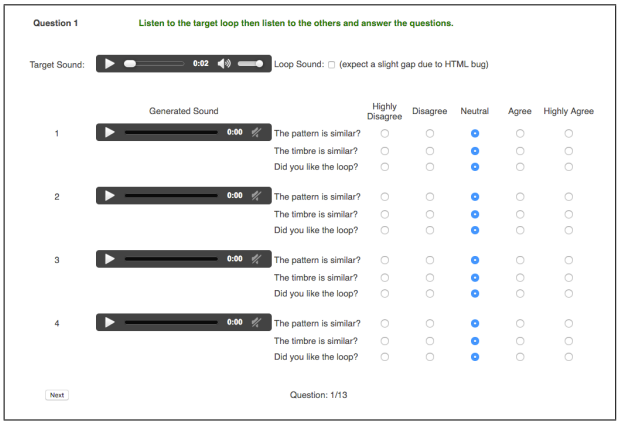


Figure 7: Screenshot of Web Survey

To directly compare and contrast with the retrieval evaluation the same 40 loops generated by the system and used in the retrieval analysis were transferred to the listening survey. The survey itself was web-based (Figure 7) and took roughly 15-20 minutes to complete. Participants were presented with the seed pattern and the generated patterns and could listen as many times as they liked. Using a 5 point Likert scale the participants were then asked to rate their agreement with the following statements:

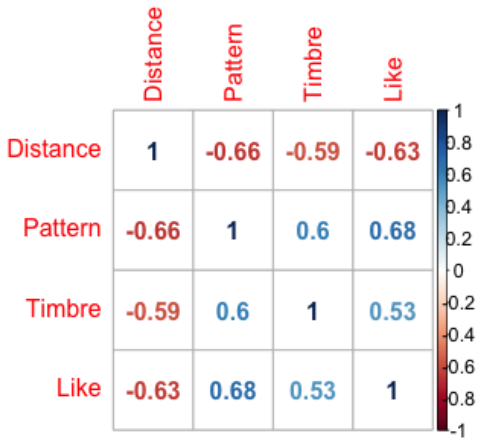


Figure 8: Correlations Between Distance and Subjective Ratings of Pattern Similarity, Timbre Similarity and Liking

- Is the rhythmic pattern similar?
- Is the timbre similar?
- Did you like the loop?

Twenty one participants in all took part in total, drawn from researchers at the authors' institute as well as friends and colleagues with an interest in music. Twenty out of the 21 participants declared they were able to play a musical instrument Ten of the 21 participants specified they played a percussion instrument and 9 reported they could read notation. In the instructions we provided brief explanations of the key terms and audio examples demonstrating contrasting rhythmic patterns and timbres.

4.3.1 Listener Evaluation Results

The survey data was analysed using Spearman's rank correlation on the mode of the participants' responses to each loop stimulus with the associated distance of that loop. We identified a moderate to strong negative correlation for each of the pattern, timbre and "likeness" aspects ($p < 0.01$ in all instances). This can be observed in the red values in the correlation matrix presented in Figure 8.

It should be evident that the subjective listening data conforms quite well to the findings of the objective retrieval rating. Increased distance resulted in decreased retrieval accuracy which in turn corresponded to a decrease in listener ratings for qualities pertaining to pattern similarity and impression of timbral similarity in the sounds themselves. Furthermore, it was revealed that the aesthetical judgement of the generated loops, encapsulated by the "likeness" factor, also followed the trend set out by the objective algorithm. We were curious to establish whether any particular subject did not conform to this preference for similar loops, but examining the individual correlation coefficients revealed all to be negative (all participants preferred more similar sounding patterns).

5. CONCLUSIONS

In this paper we presented a proposal for a framework that evaluates concatenative synthesis systems. Using a system that we developed which specifically generates rhythmic loops as a use case we demonstrated how such a framework could be applied in practice. An application-specific experiment was devised and the objective results and subjective results showed favourably the performance of the similarity algorithm involved. It is hoped that by providing a well-documented account of this process other researchers can be encouraged to adapt comparable evaluation strategies in creative applications of MIR such as concatenative synthesis.

6. ACKNOWLEDGMENTS

This research has been partially supported by the EU-funded GiantSteps project (FP7-ICT-2013-10 Grant agreement nr 610591). [1]

7. REFERENCES

[1] Jean-Julien Aucouturier and François Pachet. Ringomatic: A Real-Time Interactive Drummer Using Constraint-Satisfaction and Drum Sound Descriptors. *Proceedings of the International Conference on Music Information Retrieval*, pages 412–419, 2005.

[2] Jeronimo Barbosa, Joseph Malloch, Marcelo M. Wanderley, and Stéphane Huot. What does " Evaluation " mean for the NIME community? *NIME 2015 - 15th International Conference on New Interfaces for Musical Expression*, page 6, 2015.

[3] Gilberto Bernardes. *Composing Music by Selection: Content-based Algorithmic-Assisted Audio Composition*. PhD thesis, University of Porto, 2014.

[4] Mark Cardle, Stephen Brooks, and Peter Robinson. Audio and User Directed Sound Synthesis. *Proceedings of the International Computer Music Conference (ICMC)*, 2003.

[5] Graham Coleman. *Descriptor Control of Sound Transformations and Mosaicing Synthesis*. PhD thesis, Universitat Pompeu Fabra, 2015.

[6] Nick Collins. *Towards autonomous agents for live computer music: Realtime machine listening and interactive music systems*. PhD thesis, University of Cambridge, 2006.

[7] Matthew E. P. Davies, Philippe Hamel, Kazuyoshi Yoshii, and Masataka Goto. AutoMashUpper: An Automatic Multi-Song Mashup System. *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013*, pages 575—-580, 2013.

[1] http://www.giantsteps-project.eu

[8] Dimitri Diakopoulos, Owen Vallis, Jordan Hochenbaum, Jim Murphy, and Ajay Kapur. 21st century electronica: Mir techniques for classification and performance. In *International Society for Music Information Retrieval Conference*, pages 465–469, 2009.

[9] Benjamin Hackbarth. Audioguide : A Framework for Creative Exploration of Concatenative Sound Synthesis. *IRCAM Research Report*, 2011.

[10] Perfecto Herrera, Amaury Dehamel, and Fabien Gouyon. Automatic labeling of unpitched percussion sounds. In *Audio Engineering Society 114th Convention*, 2003.

[11] Rumi Hiraga, Roberto Bresin, Keiji Hirata, and Haruhiro Katayose. Rencon 2004: Turing Test for Musical Expression. *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 120–123, 2004.

[12] Jason A. Hockman and Matthew E. P. Davies. Computational Strategies for Breakbeat Classification and Resequencing in Hardcore, Jungle and Drum & Bass. In *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, pages 1–6, 2015.

[13] William Hsu and Marc Sosnick. Evaluating interactive music systems: An HCI approach. In *Proceedings of New Interfaces for Musical Expression*, pages 25–28, 2009.

[14] Eric J Humphrey, Douglas Turnbull, and Tom Collins. A brief review of creative MIR. *International Society for Music Information Retrieval*, 2013.

[15] Andrew J. Hunt and Alan W. Black. Unit selection in a concatenative speech synthesis system using a large speech database. *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, 1:373–376, 1996.

[16] Chris Kiefer, Nick Collins, and Geraldine Fitzpatrick. HCI Methodology For Evaluating Musical Controllers: A Case Study. *Proceedings of the 2008 International Conference on New Interfaces for Musical Expression (NIME-08)*, pages 87–90, 2008.

[17] Cárthach Ó Nuanáin, Perfecto Herrera, and Sergi Jorda. Target-Based Rhythmic Pattern Generation and Variation with Genetic Algorithms. In *Sound and Music Computing Conference 2015*, Maynooth, Ireland.

[18] François Pachet and Pierre Roy. (Manufac) Turing Tests for Music. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI 2015), Workshop on "Beyond the Turing Test*, 2015.

[19] Emmanuel Ravelli, Juan P. Bello, and Mark Sandler. Automatic rhythm modification of drum loops. *IEEE Signal Processing Letters*, 14(4):228–231, 2007.

[20] Pierre Roy, François Pachet, and Sergio Krakowski. Analytical Features for the classification of Percussive Sounds: the case of the pandeiro. In *10th Int. Conference on Digital Audio Effects (DAFx-07)*, pages 1–8, 2007.

[21] Diemo Schwarz. Current Research In Concatenative Sound Synthesis. In *Proceedings of the International Computer Music Conference*, pages 9–12, 2005.

[22] Diemo Schwarz, G Beller, B Verbrugghe, and S Britton. Real-Time Corpus-Based Concatenative Synthesis with CataRT. *Proceedings of the 9th International Conference on Digital Audio Effects*, pages 18–21, 2006.

[23] Ian Simon, Sumit Basu, David Salesin, and Maneesh Agrawala. Audio analogies: creating new music from an existing performance by concatenative synthesi. *Proceedings of the International Computer Music Conference*, 2005:65–72, 2005.

[24] Bob L. Sturm. Matconcat: An Application for Exploring Concatenative Sound Synthesis Using Matlab. In *7th International Conference On Digital Audio Effects (DAFx)*, pages 323–326, 2004.

[25] Bob L. Sturm. Adaptive Concatenative Sound Synthesis and Its Application to Micromontage Composition. *Computer Music Journal*, 30(4):46–66, 2006.

[26] Lucas Thompson, Simon Dixon, and Matthias Mauch. Drum Transcription via Classification of Bar-Level Rhythmic Patterns. In *International Society for Music Information Retrieval Conference*, pages 187–192, 2014.

[27] Adam Tindale, Ajay Kapur, George Tzanetakis, and Ichiro Fujinaga. Retrieval of percussion gestures using timbre classification techniques. *Proceedings of the International Conference on Music Information Retrieval*, pages 541–545, 2004.

AN ONTOLOGY FOR AUDIO FEATURES

Alo Allik, György Fazekas, Mark Sandler
Queen Mary University of London
a.allik, g.fazekas, mark.sandler@qmul.ac.uk

ABSTRACT

A plurality of audio feature extraction toolsets and feature datasets are used by the MIR community. Their different conceptual organisation of features and output formats however present difficulties in exchanging or comparing data, while very limited means are provided to link features with content and provenance. These issues are hindering research reproducibility and the use of multiple tools in combination. We propose novel Semantic Web ontologies (1) to provide a common structure for feature data formats and (2) to represent computational workflows of audio features facilitating their comparison. The Audio Feature Ontology provides a descriptive framework for expressing different conceptualisations of and designing linked data formats for content-based audio features. To accommodate different views in organising features, the ontology does not impose a strict hierarchical structure, leaving this open to task and tool specific ontologies that derive from a common vocabulary. The ontologies are based on the analysis of existing feature extraction tools and the MIR literature, which was instrumental in guiding the design process. They are harmonised into a library of modular interlinked ontologies that describe the different entities and activities involved in music creation, production and consumption.

1. INTRODUCTION

Several content based audio feature extraction frameworks and toolsets have been developed over the past decades of MIR research aiming to provide a platform for distributing, sharing or deploying algorithms. While most tools have the potential to become widely adopted common platforms, it is most likely that a plurality of them will continue to be used by the community as well as adopters of MIR technology. However, diverging conceptual organisation of features and different output formats present difficulties when it comes to exchanging and comparing data, or producing annotated datasets. These issues are hindering research reproducibility as well as the use of multiple data or tool sets in a single application or experiment.

A growing demand for shared representations of computational extraction workflows and interoperable data formats is signified by several proposed formats, some of which are associated with feature extractor tools or services [3, 4, 8, 13, 19]. While existing formats for structuring and exchanging content-based audio feature data may satisfy tool or task specific requirements, there are still significant limitations in linking features produced in different data sources, as well as in providing generalised descriptions of audio features that would allow easier identification and comparison of algorithms that produce the data.

Semantic Web technologies facilitate formal descriptions of concepts, terms and relationships that enable implementations of automated reasoning and data aggregation systems to manage large amounts of information within a knowledge domain. Both in research and commercial use cases, it is becoming increasingly important to fuse cultural, contextual and content-based information. This may be achieved by leveraging Linked Data enabled by the use of shared ontologies and unique identification of entities. This not only offers the potential to simplify experiments and increase productivity in research activities traditionally relying on Web scraping, proprietary application programming interfaces or manual data collection, but also enables incorporation of increasingly larger and more complex datasets into research workflows.

We propose a modular approach towards ontological representation of audio features. Since there are many different ways to structure features depending on a specific task or theoretically motivated organising principle, a common representation would have to account for multiple conceptualisations of the domain and facilitate diverging representations of common features. This may be due to the "semantic gap" between low-level computational representations of audio signals and theoretical representations founded in acoustics or musicology. This semantic gap could potentially be bridged using Semantic Web technologies if high-level feature identification can be inferred from computational signatures. However, this functionality is currently beyond the scope of existing technologies. For example, Mel Frequency Cepstral Coefficients (MFCC), which are widely calculated in many tools and workflows, can be categorised as a "timbral" feature in the psychoacoustic or musicological sense, while from the computational point of view, MFCC could be labelled as a "cepstral" or "spectral" representation. The complexity arising from this makes music somewhat unique calling for a robust ontological treatment, although ontological representation of content-based features have been proposed in other domains including image processing [23].

Our framework consists of two separate components to

© Alo Allik, György Fazekas, Mark Sandler. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Alo Allik, György Fazekas, Mark Sandler. " An ontology for audio features ", 17th International Society for Music Information Retrieval Conference, 2016.

distinguish between the abstract concepts describing the audio feature domain and more concrete classes that represent specific audio features and their computational signatures. The Audio Feature Ontology (AFO) is the more abstract component representing entities in the feature extraction process on different levels of abstraction. It describes the structure of processes in feature extraction workflow through phases of conceptualisation, modelling and implementation. The Audio Feature Vocabulary (AFV) then lists existing audio features providing the terms for tool and task specific ontologies without attempting to organise the features into a taxonomy.

2. BACKGROUND

Many recent developments in audio feature data formats employ JavaScript Object Notation (JSON), which is rapidly becoming a ubiquitous data interchange mechanism in a wide range of systems regardless of domain. Particularly, the JSON Annotated Music Specification (JAMS) [8] is a notable attempt to provide meaningful structure to audio feature data while maintaining simplicity and sustainability in representations, which the authors deem as the most crucial factors for wider adoption in the MIR community. A different JSON format for features has been developed in the AcousticBrainz project [19] with the intention of making available low and high level audio features for millions of music tracks. This resource provides the largest feature dataset to date while exposing the extraction algorithms in an open source environment.

It is evident that the simplicity of JSON combined with its structuring capabilities make it an attractive option, particularly compared to preceding alternatives including YAML, XML, Weka Attribute-Relation File Format (ARFF), the Sound Description Interchange Format (SDIF), and various delimited formats. All these formats enable communication between various workflow components and offer varying degrees of flexibility and expressivity. However, even the most recent JSON methods only provide a structured representation of feature data without the facility of linking these concepts semantically to other music related entities or data sources. For example, the JAMS definition does not address methodology that would enable detailed description and comparison of audio features nor does it provide a structured way of linking the feature data to the rest of the available metadata for a particular music track. Admittedly, the AcousticBrainz dataset does provide links to the MusicBrainz [1] database by global identifiers, but there is no mechanism to identify the features or compare them to those available in other extraction libraries. The Essentia library [3] that is used in the AcousticBrainz infrastructure for feature extraction is open source, thus providing access to the algorithms, but there is no formalised description of audio features beyond source code and documentation yet. Other feature extraction frameworks provide data exchange formats designed for particular workflows or specific tools. However, there is no common format shared by all the different tools and libraries. The motley of output formats is well demonstrated in the representations category of a recent evaluation of feature extraction toolboxes [15]. For example, the popular MATLAB MIR Toolbox export function outputs delimited files as well as ARFF, while Essentia provides YAML and JSON and the YAAFE library outputs CSV and HDF5. The MPEG-7 standard, used as benchmarks for other extraction tools provides an XML schema for a set of low-level descriptors, but the deficiencies highlighted above also apply in this case.

Semantic Web technologies provide domain modelling and linking methods considerably beyond the expressivity and interoperability of any of the solutions described above. The OWL family of ontology languages is designed to be flexible enough to deal with heterogeneous Web-based data sources. It is also built on strong logical foundations. It implies a conceptual difference between developing data formats and ontological modelling. The authors of [8] mention a common criticism that RDF-based MIR formats similarly to XML suffer from being non-obvious, verbose or confusing. However, the potential of meaningful representation of audio features and linking ability to other music related information outweighs these concerns. Ontological representation and linking of divergent domains is a difficult task, but should not be discarded lightly in favour of simplicity. The benefits of even relatively limited Semantic Web technologies for MIR research have been demonstrated on a number of occasions. For example, the proof-of-concept system described in [17] enables increased automation and simplification of research workflows and encourages resource reuse and validation by combining several existing ontologies and Semantic Web resources, including the Music Ontology [2], GeoNames [3], DBTune [4], and the Open Archives Initiative Object Reuse and Exchange (OAI-ORE). A system for MIR workflow preservation has been proposed in [12], which emphasises the importance of representing and preserving the context of entire research processes and describes a Context Model of a typical MIR workflow as a Semantic Web ontology.

The original version of the Audio Feature Ontology was created within a framework of a harmonised library of modular music-related Semantic Web ontologies [4], built around the core Music Ontology [20]. This library relies on widely adopted Semantic Web ontologies such as the Friend of a Friend (FOAF) vocabulary, as well as domain specific ontologies for describing intellectual works (FRBR) and complex associations of domain objects with time-based events (Event and Timeline ontologies). The library also provides a set of extensions describing music specific concepts including music similarity [9] and the production of musical works in the recording studio [5]. Since its publication, it has been integrated in several research projects, including the Networked Environment for Music Analysis (NEMA) [24], the Computational Analysis of the Live Music Archive (CALMA) [2] as well as commercial applications, e.g. the BBC and its music Web-

[1] http://musicbrainz.org

[2] http://musicontology.com/

[3] http://www.geonames.org/

[4] http://dbtune.org/

site [5]. This ontology provides a model for structuring and publishing content-derived information about audio recordings and allows linking this information to concepts in the Music Ontology framework. However, it does not provide a comprehensive vocabulary of audio features or computational feature extraction workflows. It also lacks concepts to support development of more specific feature extraction ontologies. Structurally, it conflates musicological and computational concepts to an extent that makes it inflexible for certain modelling requirements as suggested in [7]. In order to address these issues, the updated Audio Feature Ontology separates abstract ontological concepts from more specific vocabulary terminology, supplies methodology for extraction workflow descriptions, and increases flexibility for modelling of task and tool specific ontologies.

3. ONTOLOGY ENGINEERING

In order to gain a better understanding of the MIR domain and user needs, a catalogue of audio features was first compiled based on a review of relevant literature, existing feature extraction tools and research workflows. The first phase of this process involved extracting information about features from journal articles, source code and existing structured data sources. This information was subsequently collated into a linked data resource to serve as a foundation for the ontology engineering process. There was no attempt at explicit classification of features into a hierarchical taxonomy. Source code was parsed from a number of open source feature extraction packages including CLAM [1], CoMIRVA [21] jMIR (jAudio) [13], LibXtract [6], Marsyas [7], Essentia [3] and YAAFE [11]. Existing linked resource representations of the Vamp plugins provided easy access to all the features available for download on the Vamp plugins Website [8]. Manual extraction was used for the packages which did not provide suitable access for automatic parsing or which were reviewed in journal articles, including the Matlab toolboxes (MIR Toolbox and Timbre Toolbox), Aubio [9], Cuidado [18], PsySound3 [10], sMIRk [6], SuperCollider SCMIR toolkit, and some of the more recent MIREX submissions. A simple automatic matching procedure was employed to identify synonymous features using a Levenshtein distance algorithm, which aided the compilation of a feature synonym dictionary.

The catalogue was created in linked data format using the Python RDFLib library [11], which enables quick and easy serialisation of linked data into various formats. The catalogue lists feature objects and their attributes and serves as the foundation for a hybrid ontology engineering process combining manual and semi-automatic approaches. The catalogue identifies approximately 400 distinct features and thereby significantly increases the scope of the original ontology, which supports identifying about 30 entities. The catalogue has been published online [12] and allows querying subsets of popular features computed in feature extraction tools to help define the scope and domain boundaries of the ontology. It also sheds light on the range of classifications of features inherent in different software tools and libraries, as well as conceptualisations of the domain in journal articles. Figure 1 shows three divergent organisations of features from very different sources.

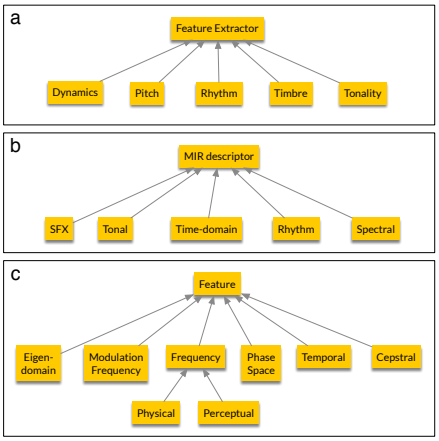


Figure 1. Three different taxonomies of audio features extracted from (a) MIR Toolbox, (b) Essentia, and (c) Mitrovic et al. [14]

The catalogue exemplifies the diversity of viewpoints on classification of features within the community. It is clear that in some cases audio features are categorised according to musicological concepts, such as pitch, rhythm and timbre, while in others, the classification is based on the computational workflows used in calculating the features or a combination of different domains depending on the task. Consequently, there is no need to impose a deeply taxonomical structure on the collected audio features, rather the resulting ontology should be focused on facilitating structured feature data representation that is flexible enough to accommodate all these diverging organisational principles.

4. CORE ONTOLOGY MODEL

The most significant updates to the original ontology model are designed to address a number of requirements determined during the engineering process. The proposed updates are intended to:

- provide comprehensive vocabulary of audio features
- define terms for capturing computational feature extraction workflows
- support development of domain and task specific ontologies for existing extraction tools
- restructure concept inheritance for more flexible and sustainable feature data representation

[5] http://bbc.co.uk/music/
[6] http://libxtract.sourceforge.net
[7] http://marsyas.info
[8] http://www.vamp-plugins.org/
[9] http://aubio.org/
[10] http://psysound.wikidot.com/
[11] https://github.com/RDFLib/rdflib
[12] http://sovarr.c4dm.eecs.qmul.ac.uk/af/catalog/1.0#

- facilitate design of linked data formats that combine strong logical foundations of ontological structuring with simplicity of representations.

The fundamental structure of the ontology has changed in a couple of key aspects. The core structure of the framework separates the underlying classes that represent abstract concepts in the domain from specific named entities. This results in the two main components of the framework defined as Audio Feature Ontology (AFO, `http://w3id.org/afo/onto/1.1#`) and Audio Feature Vocabulary (AFV, `http://w3id.org/afo/vocab/1.1#`). The main differences also include introducing levels of abstraction to the class structure and reorganising the inheritance model. The different layers of abstraction represent the feature design process from conceptualisation of a feature, through modelling a computational workflow, to implementation and instantiation in a specific computational context. For example, the abstract concept of Chromagram is separate from its model which involves a sequence of computational operations like cutting an audio signal into frames, calculating the Discrete Fourier Transform for each frame, etc. (see Section 5.1 for a more detailed example, and [16] for different methods for extracting Chroma-based features). The abstract workflow model can be implemented using various programming languages as components of different feature extraction software applications or libraries. Thus, this layer enables distinguishing a Chromagram implementation as a Vamp plugin from a Chromagram extractor in MIR Toolbox. The most concrete layer represents the feature extraction instance, for example, to reflect the differences of operating systems or hardware on which the extraction occurred. The layered model is shown in Figure 2.

Figure 2. The Audio Feature Ontology core model with four levels of abstraction

The core model of the ontology retains original attributes to distinguish audio features by temporal characteristics and data density. It relies on the Event and Timeline ontologies to provide the primary structuring concepts for feature data representation. Temporal characteristics classify feature data either into instantaneous points in time - e.g. event onsets or tonal change moments - or events with known time duration. Data density attributes allow describing how a feature relates to the extent of an audio file: whether it is scattered and occurs irregularly over the course of the audio signal, or the feature is calculated at regular intervals and fixed duration. The change in the inheritance model removes the music-specific subclassing of **afo:Point**, **afo:Segment**, and **afo:Signal** classes which was claimed to make feature representation less flexible in certain use cases [7]. The Segment Ontology was proposed as a solution to get around these limitations [7], in which the Segment class functions as a music-generic dimension between explicitly temporal and implicitly temporal concepts, thus enabling multiple concurrent domain-specific concepts to be represented. An alternative solution is to subclass **afo:Point**, **afo:Segment**, and **afo:Signal** directly from **afo:AudioFeature**, which, in turn, is a subclass of **event:Event**. In this case, the feature extraction data can be directly linked to the corresponding term in AFV without being constrained by domain or task specific class definitions. This way, it is not necessary to add the Segment Ontology concepts to feature representations, thereby simplifying the descriptions.

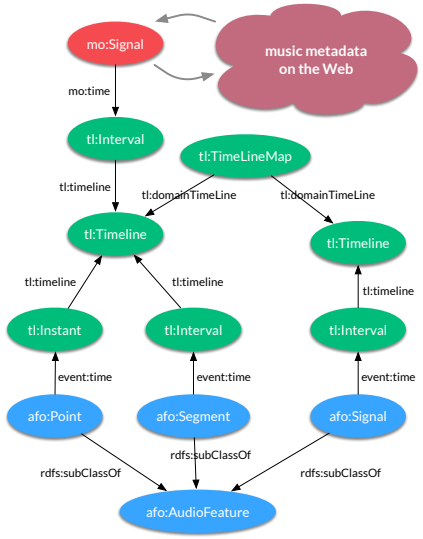


Figure 3. Framework model showing how feature data representation is linked with music metadata resources on the Web using temporal entities defined in the Timeline ontology

Audio features collated from literature and extraction software are defined as subclasses in the AFV. An illustrative Turtle-syntax representation that shows the basic principle of how subclassing **afo:AudioFeature** functions in the context of annotating both sparse and dense features is provided in Section 5.2. The other purpose of the vocabulary is to define computational extraction workflow descriptions, so that features can be more easily identified and compared by their respective computational signatures. The following section delves into this in more detail.

5. CASE STUDIES AND EVALUATION

5.1 Representing computational workflows

AFV defines terms for the tool and task specific ontologies and implements the model layer of the ontology framework. It is a clean version of the catalogue which only lists the features without any of their properties with many duplications of terms consolidated. This enables the definition of tool and task specific feature implementations and leaves any categorisation or taxonomic organisation to be specified in the implementation layer.

The vocabulary also specifies computational workflow

models for some of the features which can be linked to from lower level ontologies. The computational workflow models are based on feature signatures as described in [14]. The signatures represent mathematical operations employed in the feature extraction process with each operation assigned a lexical symbol. It offers a compact description of each feature and enables an easier way of comparing features according to their extraction workflows. Converting the signatures into a linked data format to include them in the vocabulary involves defining a set of OWL classes that handle the representation and sequential nature of the calculations. The operations are implemented as sub-classes of three general classes: transformations, filters and aggregations. For each abstract feature, we define a model property. The OWL range of the model property is a ComputationalModel class in the Audio Feature Ontology namespace. The operation sequence can be defined through this object's operation sequence property. For example, the signature of the Chromagram feature defined in [14] as "f F l Σ", which designates a sequence of (1) windowing (f), (2) Discrete Fourier Transform (F), (3) logarithm (l) and (4) sum (Σ) is expressed as a sequence of RDF statements in Listing 1.

```
afv:Chromagram a owl:Class ;
  afo:model afv:ChromagramModel ;
  rdfs:subClassOf afo:AudioFeature .

afv:ChromagramModel a afo:Model;
  afo:sequence afv:Chromagram_operation_sequence_1 .

afv:Chromagram_operation_sequence_1 a afv:Windowing;
  afo:next_operation
       afv:Chromagram_operation_sequence_2 .

afv:Chromagram_operation_sequence_2 a
     afv:DiscreteFourierTransform;
  afo:next_operation
       afv:Chromagram_operation_sequence_3 .

afv:Chromagram_operation_sequence_3 a afv:Logarithm;
  afo:next_operation
       afv:Chromagram_operation_sequence_4 .

afv:Chromagram_operation_sequence_4 a
     afo:LastOperation, afv:Sum .
```

Listing 1. Description of a chromagram computation.

```
SELECT DISTINCT ?feature
WHERE {
  ?opid a afv:DiscreteCosineTransform .
  ?seqid afo:first_operation ?fopid .
  ?fopid afo:next_operation+ ?opid .

  OPTIONAL {
    ?model afo:operation_sequence ?seqid .
    ?feature afo:model ?model .
  }
}
```

Listing 2. Retrieving feature types involving the DCT.

This structure enables building SPARQL queries to retrieve comparative information on features from the vocabulary. For example, we can inquire which features in the vocabulary employ the Discrete Cosine Transform calculation by executing the query of Listing 2. The query will produce the following result:

```
afv:AutocorrelationMFCCs
afv:BarkscaleFrequencyCepstralCoefficients
afv:MelscaleFrequencyCepstralCoefficients
afv:ModifiedGroupDelay
afv:ModulationHarmonicCoefficients
afv:NoiseRobustAuditoryFeature
afv:PerceptualLinearPrediction
afv:RelativeSpectralPLP
```

5.2 Audio content description

In order to determine how well the AFO framework represents the audio feature extraction domain, we need to test its suitability for representing audio features in the context of particular use cases. We employ a task-based methodology to focus on evaluating the suitability of AFO in a feature extraction workflow. Task-based evaluation is based on having a set of pre-defined requirements and it may offer a measure of practical aspects, such as the human ability to formulate queries using an ontology, or the accuracy of responses provided by the system's inferential component. In order to qualitatively evaluate the AFO framework, we need to define a set of requirements from the perspective of music information retrieval workflows. Reviewing common research workflows, the following main requirements for audio feature annotations have been discovered:

- identify an extracted audio feature by linking it to a corresponding term in the Audio Feature Vocabulary
- identify the computational steps involved in the process
- describe the temporal structure and density of output
- associate audio features with the audio signal timeline
- identify the feature extraction software tools used in the extraction process

Sparse point-like and dense signal-like features of an audio file - such as onsets or MFCC - can be linked directly to their respective classes in AFV in the feature extraction process as shown in Listing 3.

The Turtle representation is but one of the possible means of serialisation. AFO can facilitate development of other data formats that are aligned with linked data principles, including binary RDF representations. One of the goals of the development process has been to look for alternative formats that could be used in different contexts. Due to the wide appeal of JSON, the ontology also enables publishing feature data in its linked data version. JSON-LD is an extension to the standard JSON format that provides an entity-centric representation of RDF/OWL semantics and a means to define a linked data context with URI connections to external ontologies and resources [10]. It has the potential to simplify feature representations while maintaining ontological structuring of the data. The format enables establishing links to ontologies where the structure of the data is defined by using the key word "@context". OWL class types are annotated with "@type" and unique

identifiers are with "@id". The latter functions as a linking mechanism between nodes when an RDF graph is converted into a JSON tree structure. The JSON-LD representation of audio features has been tested in the context of an adaptive music player.

```
@prefix afv: <http://w3id.org/afo/vocab/1.1#> .
@prefix mo: <http://purl.org/ontology/mo/> .
@prefix tl:
     <http://purl.org/c4dm/timeline.owl#> .
@prefix vamp: <http://purl.org/ontology/vamp/> .

:signal_f6261475 a mo:Signal ;
  mo:time [
    a tl:Interval ;
    tl:onTimeLine :timeline_aec1cb82
  ] .

:timeline_aec1cb82 a tl:Timeline .

:transform_onsets a vamp:Transform ;
  vamp:plugin plugbase:qm-onsetdetector ;
  vamp:output
       plugbase:qm-onsetdetector_output_onsets .

:transform_mfcc a vamp:Transform ;
  vamp:plugin plugbase:qm-mfcc ;
  vamp:output
       plugbase:qm-mfcc_output_coefficients .

:event_1 a afv:Onset ;
  event:time [
    a tl:Instant ;
    tl:onTimeLine :timeline_aec1cb82 ;
    tl:at "PT1.98S"^^xsd:duration ;
  ] ;
  vamp:computed_by :transform_onsets .

:feature_1 a afv:MFCC ;
  mo:time [
    a tl:Interval ;
    tl:onTimeLine :timeline_aec1cb82 ;
  ] ;
  vamp:computed_by :transform_mfcc ;
  afo:value ( -26.9344 0.188319 0.106938 ..) .
```

Listing 3. An abbreviated example of linking onsets and MFCC features to AFV and the Music Ontology

5.3 Case study: adaptive music player

Beyond representing audio feature data in research workflows, there are many other practical applications for the ontology framework. One of the test cases is providing data services for an adaptive music player that uses audio features to enrich user experience and enables novel ways to search or browse large music collections. Feature data of the music tracks available in the player is stored in a CouchDB [13] instance in JSON-LD. The data is used by Semantic Web entities called Dynamic Music Objects (dymos) [22] that control the audio mixing functionality of the player. Dymos make song selections and determine tempo alignment for cross-fading based on features. Listing 4 shows an example of JSON-LD representation of a track used in the system linked to feature annotations.

6. CONCLUSIONS

The Audio Feature Ontology and Vocabulary provide a framework for representing audio features using Semantic Web methods and linked data technologies. It provides terminology to facilitate task and tool specific ontology development and serves as a descriptive framework

[13] http://couchdb.apache.org/

```
{
   "@context": {
      "foaf": "http://xmlns.com/foaf/0.1/",
      "afo": "http://w3id.org/afo/onto/1.1#",
      "afv": "http://w3id.org/afo/vocab/1.1#",
      "mo": "http://purl.org/ontology/mo/",
      "dc": "http://purl.org/dc/elements/1.1/",
      "tl": "http://purl.org/NET/c4dm/timeline.owl#",
      "vamp": "http://purl.org/ontology/vamp/"
   },
   "@type": "mo:Track",
   "dc:title": "Open My Eyes",
   "mo:artist": {
      "@type": "mo:MusicArtist",
      "foaf:name": "The Nazz"
   },
   "mo:available_as": "/home/snd/250062-15.01.wav",
   "mo:encodes": {
      "@type": "mo:Signal",
      "mo:time": {
         "@type": "tl:Interval",
         "tl:duration": "PT163S",
         "tl:timeline": {
            "@type": "tl:Timeline",
            "@id": "98cfa995.."
   }}},
   "afo:features": [
      {
         "@type": "afv:Key",
         "vamp:computed_by": {
            "@type": "vamp:Transform",
            "vamp:plugin_id":
               "vamp:qm-vamp-plugins:qm-keydetector"
         },
         "afo:values": [
            { "tl:at": 1.4 , "rdfs:label": "C# minor",
               "tl:timeline": "98cfa995.."
            },
            { "tl:at": 5.9 , "rdfs:label": "D minor",
               "tl:timeline": "98cfa995.."
            }
         ]
      }]}
```

Listing 4. JSON-LD representation of an audio feature linked with track metadata

for audio feature extraction. The updates to the original ontology for audio features strive to simplify feature representations and make them more flexible while maintaining ontological structuring and linking capabilities. JSON-LD has been shown to function as a linked data format that enables converting RDF graph structures to key-value representation. This could also apply for other similar data formats and NoSQL database systems. The ontology engineering process has produced example ontologies for existing tools including MIR Toolbox, Essentia, Marsyas and others available from the ontology Website http://w3id.org/afo/onto/1.1#.

7. ACKNOWLEDGEMENTS

This work was part funded by the FAST IMPACt EPSRC Grant EP/L019981/1 and the European Commission H2020 research and innovation grant AudioCommons (688382). Sandler acknowledges the support of the Royal Society as a recipient of a Wolfson Research Merit Award.

8. REFERENCES

[1] X. Amatriain, P. Arumi, and D. Garcia. Clam: A framework for efficient and rapid development of cross-

platform audio applications. In *proc. 14th ACM International Conference on Multimedia*, pages 951–954, New York, USA, 2006.

[2] S. Bechhofer, S. Dixon, G. Fazekas, T. Wilmering, and K. Page. Computational analysis of the live music archive. In *proc. 15th International Conference on Music Information Retrieval (ISMIR)*, 2014.

[3] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X. Serra. Essentia: an audio analysis library for music information retrieval. In *proc. 14th International Society for Music Information Retrieval Conference (ISMIR'13)*, pages 493–498, Curitiba, Brazil, 2013.

[4] G. Fazekas, Y. Raimond, K. Jakobson, and M. Sandler. An overview of semantic web activities in the OMRAS2 project. *Journal of New Music Research (JNMR)*, 39(4), 2010.

[5] G. Fazekas and M. Sandler. The studio ontology framework. In *proc. 12th International Society for Music Information Retrieval (ISMIR'11) conference, Miami, USA, 24-28 Oct.*, pages 24–28, 2011.

[6] R. Fiebrink, G. Wang, and Perry R. Cook. Support for MIR prototyping and real-time applications in the chuck programming language. In *proc. 9th International Conference on Music Information Retrieval, Philadelphia, PA, USA, September 14-18*, 2008.

[7] B. Fields, K. Page, D. De Roure, and T. Crawford. The segment ontology: Bridging music-generic and domain-specific. In *proc. IEEE International Conference on Multimedia and Expo, Barcelona, Spain, 11-15 July*, 2011.

[8] E. J. Humphrey, J. Salamon, O. Nieto, J. Forsyth, R. Bittner, and J. P. Bello. JAMS: A JSON annotated music specification for reproducible MIR research. In *15th Int. Soc. for Music Info. Retrieval Conf.*, pages 591–596, Taipei, Taiwan, Oct. 2014.

[9] K. Jacobson, Y. Raimond, and M. Sandler. An ecosystem for transparent music similarity in an open world. In *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe, Japan, October 26-30*, 2009.

[10] M. Lanthaler and C. Gütl. On using JSON-LD to create evolvable restful services. In *proc. 3rd International Workshop on RESTful Design at WWW'12*, 2012.

[11] B. Mathieu, B. Essid, T. Fillon, J. Prado, and G. Richard. YAAFE, an easy to use and efficient audio feature extraction software. In *proc. 11th International Conference on Music Information Retrieval (ISMIR), Utrecht, Netherlands, August 9-13*, 2010.

[12] R. Mayer and A. Rauber. Towards time-resilient mir processes. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR), Porto, Portugal, October 8-12*, 2012.

[13] C. McKay and I. Fujinaga. Improving automatic music classification performance by extracting features from different types of data. In *proc. of the International Conference on Multimedia Information Retrieval*, pages 257–266, New York, USA, 2010.

[14] D. Mitrovic, M. Zeppelzauer, and C. Breiteneder. Features for content-based audio retrieval. *Advances in Computers*, 78:71–150, 2010.

[15] D Moffat, D Ronan, and J. D. Reiss. An evaluation of audio feature extraction toolboxes. In *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, Trondheim, Norway, 2015.

[16] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *proc. 12th International Society for Music Information Retrieval (ISMIR'11) conference, Miami, USA*, 2011.

[17] K. Page, B. Fields, B. Nagel, G. O'Neill, D. De Roure, and T. Crawford. Semantics for music analysis through linked data: How country is my country? In *Proceedings of the 6th International Conference on e-Science, Brisbane, Australia, 7-10 Dec.*, 2010.

[18] G. Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Tech. rep., IRCAM, 2004.

[19] A. Porter, D. Bogdanov, R. Kaye, R. Tsukanov, and X. Serra. Acousticbrainz: a community platform for gathering music information obtained from audio. In *proc. 16th International Society for Music Information Retrieval (ISMIR) Conference*, 2015.

[20] Y Raimond, S. Abdallah, M. Sandler, and F. Giasson. The music ontology. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria, September 23-27*, 2007.

[21] Markus Schedl. The CoMIRVA Toolkit for Visualizing Music-Related Data. Technical report, Dept. of Computational Perception, Johannes Kepler Univ., 2006.

[22] Florian Thalmann, Alfonso Perez Carillo, György Fazekas, Geraint A. Wiggins, and Mark Sandler. The mobile audio ontology: Experiencing dynamic music objects on mobile devices. In *proc. 10th IEEE International Conference on Semantic Computing, Laguna Hills, CA, USA*, 2016.

[23] M. Vacura, V. Svátek, C. Saathoff, T. Ranz, and R. Troncy. Describing low-level image features using the COMM ontology. In *proc. 15th International Conference on Image Processing (ICIP), San Diego, California, USA*, pages 49–52, 2008.

[24] K. West, A. Kumar, A. Shirk, G. Zhu, J. S. Downie, A. F. Ehmann, and M. Bay. The networked environment for music analysis (NEMA). In *proc. 6th World Congress on Services, Miami, USA, July 5-10*, 2010.

ANALYSIS AND CLASSIFICATION OF PHONATION MODES IN SINGING

Daniel Stoller
Queen Mary University of London
d.stoller@qmul.ac.uk

Simon Dixon
Queen Mary University of London
s.e.dixon@qmul.ac.uk

ABSTRACT

Phonation mode is an expressive aspect of the singing voice and can be described using the four categories *neutral*, *breathy*, *pressed* and *flow*. Previous attempts at automatically classifying the phonation mode on a dataset containing vowels sung by a female professional have been lacking in accuracy or have not sufficiently investigated the characteristic features of the different phonation modes which enable successful classification. In this paper, we extract a large range of features from this dataset, including specialised descriptors of pressedness and breathiness, to analyse their explanatory power and robustness against changes of pitch and vowel. We train and optimise a feed-forward neural network (NN) with one hidden layer on all features using cross validation to achieve a mean F-measure above 0.85 and an improved performance compared to previous work. Applying feature selection based on mutual information and retaining the nine highest ranked features as input to a NN results in a mean F-measure of 0.78, demonstrating the suitability of these features to discriminate between phonation modes. Training and pruning a decision tree yields a simple rule set based only on *cepstral peak prominence* (CPP), *temporal flatness* and *average energy* that correctly categorises 78% of the recordings.

1. INTRODUCTION

Humans have the extraordinary capability of producing a wide variety of sounds by manipulating the complex interaction between the vocal folds and the vocal tract. This flexibility also manifests in the singing voice, giving rise to a large number of different types of expression such as vibrato or glissando. In this paper, we focus on *phonation mode* as one of these expressive elements. Sundberg [22] defines four phonation modes within a two-dimensional space spanned by *subglottal pressure* and *glottal airflow*: *Neutral* and *breathy* phonations involve less subglottal pressure than *pressed* and *flow* phonations, while neutral and pressed phonations have lower glottal airflow than breathy and flow phonations.

The phonation mode is an important part of singing and can be seen as an expressive dimension along with pitch and loudness [23]. This additional degree of control allows more room for interpretation and expression - a breathy voice for example can be used to portray sweetness and sexuality, while pressed voices can seem forceful and tense [20]. In addition to individual differences between singers, the phonation mode tends to vary depending on the musical style, as shown in a study with four different genres by Borch and Sundberg [5]. Automatically detecting the phonation mode could help diagnose certain vocal disorders such as the hypofunction and hyperfunction of the glottis [10]. Because many singing students in particular exhibit varying degrees of these malfunctions throughout the course of their studies, teachers could be assisted to correct this behaviour during lessons. Apart from music, phonation modes also play an important role in speech. For the task of speaker-independent emotion recognition, phonation mode is one of the features of voice quality that can be useful for reliably detecting emotion in speech [16].

2. RELATED WORK

Several studies have investigated phonation modes from a physiological and a signal processing perspective.

By using direct body measurements, Grillo and Verdolini [11] showed that laryngeal resistance as the ratio of subglottal pressure and average glottal airflow can reliably account for the difference between pressed, neutral and breathy phonation, although not between neutral and resonant voice. Subglottal pressure was also found to correlate with the amount of phonatory pressedness in a similar study, along with the closing quotient of the glottis and the difference in amplitudes of the first two harmonics in the voice source spectrum [17]. Without direct body measurements however, it is difficult to estimate subglottal pressure based only on auditory information.

As a result, signal-based feature descriptors have been developed to estimate the degree of pressedness. Most notably, the *normalised amplitude quotient* (NAQ) describes the glottal closing phase and was shown to be more robust than the closing quotient when separating breathy, neutral and pressed spoken vowels [1, 3]. This capability apparently transfers to the singing voice: Given vocal recordings featuring the four different phonation modes rated by a panel of experts, the NAQ accounted for 73% of the variation in the ratings of perceived pressedness [24]. Other descriptors have been proposed for discriminating breathy from tense voices, such as the *peak slope* [14] and the *maxima dispersion quotient* (MDQ) [15]. The *cepstral*

© Daniel Stoller, Simon Dixon. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Daniel Stoller, Simon Dixon. "Analysis and classification of phonation modes in singing", 17th International Society for Music Information Retrieval Conference, 2016.

peak prominence (CPP) [12] feature was shown to correlate strongly with ratings of perceived breathiness. In the context of singing however, the suitability of these features to capture the characteristics of all four phonation modes remains largely unknown and is investigated in this paper.

For the automatic detection of phonation modes, a dataset containing vowels sung by a female professional was created [21]. On this dataset, an automatic classification method [20] based on modelling the human vocal tract and estimating the glottal source waveform was developed. The physiological nature of the model can give insight into how humans produce phonation modes by interpreting optimised model parameters. However, only moderate accuracies between 60% and 75% were achieved, despite training the model on each vowel individually, resulting in a less general classification problem where vowel-dependent effects do not have to be taken into account. Another classification attempt on the same dataset using features derived from *linear predictive coding* (LPC) such as formant frequencies achieved a mean F-measure of 0.84 [13] with a *logistic model tree* as classifier. However, the accuracy may be high partly due to not excluding the higher pitches in the dataset, which the singer was only able to produce in breathy and neutral phonation. As a result, only two instead of four classes have to be distinguished in the higher pitch range, incentivising the classifier to extract pitch-related information to detect this situation. Although the authors identify CPP and the difference between the first two harmonics of the voice source as useful features, they do not systematically analyse how their features allow for successful classification to derive an explanation for phonation modes as an acoustic phenomenon.

This paper focuses on finding the features that best explain the differences between phonation modes in the context of singing. We investigate whether individual features, especially descriptors such as NAQ and MDQ, can directly distinguish some of the phonation modes. Different sets of features are constructed and used for the automatic classification of phonation modes to compare their explanatory power. In its optimal configuration, our classifier significantly outperforms existing approaches.

3. DATASET

We use the dataset provided by [21], which contains single sustained vowels sung by a female professional recorded at a sampling frequency of 44.1 KHz. Every phonation mode is reproduced with each of the nine vowels A, AE, I, O, U, UE, Y, OE and E and with pitches ranging from A3 to G5. However, pitches above B4 do not feature the phonation modes flow and pressed. To create a balanced dataset, where all four classes are approximately equally represented for each combination of pitch and vowel, we only use pitches between A3 and B4 and also exclude alternative recordings of the same phonation mode. If not stated otherwise, the balanced dataset called *DS-Bal* is used in this study. The full dataset *DS-Full* is only used to enable a comparison with classification results from previous work [13].

No.	Feature	No.	Feature
F1	MFCC40B	F15	Harmonic 1-6 amp.
F2	MFCC80B	F16	HNR 500
F3	MFCC80B0	F17	HNR 1500
F4	MFCC80BT	F18	HNR 2500
F5	Temp. Flatness	F19	HNR 3500
F6	Spec. Flatness	F20	HNR 4500
F7	ZCR	F21	Formant 1-4 amp.
F8	Spec. Flux Mean	F22	Formant 1-4 freq.
F9	Spec. Flux Dev.	F23	Formant 1-4 bandw.
F10	Spec. Centroid	F24	CPP
F11	HFE1	F25	NAQ
F12	HFE2	F26	MDQ
F13	F0 Mean	F27	Peak Slope
F14	F0 Dev.	F28	Glottal Peak Slope

Table 1. List of features used in this paper. A detailed explanation can be found in section 4.

4. FEATURES

A large number of features listed in table 1 is extracted to facilitate an extensive comparison and evaluation. Apart from the first three features, *trimmed* audio samples containing only the centre 600 ms of every recording are used for extraction to remove potential silences and note transients, keeping the stable part of the phonation and ensuring the reliability of LPC-derived features. For time-dependent features, frames of 50 ms with a Hanning window and 50% overlap are used for extraction before the mean of all frames is calculated, unless otherwise noted. In addition to common spectral features, we include features introduced in section 2 specifically designed to estimate phonatory pressedness or breathiness, because they should be particularly useful in this task.

Mel-frequency cepstral coefficients (MFCCs, F1-F4) are timbre descriptors used widely in MIR and speech research. In this paper, the n-th coefficient of an MFCC vector will be denoted as MFCC(n). The first feature MFCC40B (F1) is a 40-dimensional MFCC vector using the standard number of 40 Mel bands for the summarisation of the spectrum and including the 0-th coefficient representing energy. Presumably, the lower coefficients capture information more relevant to phonation modes, as they encode timbral properties that are independent of pitch. Therefore, we additionally include MFCC80B (F2), which is the first 40 coefficients of the MFCCs computed with 80 instead of 40 Mel bands, giving increased resolution in the lower coefficients. To determine the importance of energy as a feature for successful classification, MFCC80B0 (F3) is introduced as a variant of MFCC80B that is also 40-dimensional, but does not include the 0-th coefficient. As an additional variant of MFCC80B (F2), we extract MFCC80BT (F4), not from the full but from the trimmed recordings of the sung vowels, to investigate the importance of timbral information at the vowel onset and release.

Although MFCCs represent the audio signal very efficiently, as they encode most of the energy in the spec-

trum in the lower coefficients using the discrete cosine transform, they are hard to interpret as they mostly lack an intuitive description. We add a range of spectral features (F5-F12) to allow for a more comprehensible explanation of the phonation mode as an acoustic phenomenon. More specifically, temporal flatness (F5) and spectral flatness (F6) compute the ratio between the geometric and the arithmetic mean of the audio signal in the time and in the frequency domain, respectively, and describe whether the signal is smooth or spiky. The spectral flux is summarised by its mean (F8) and standard deviation (F9). As an estimation of high-frequency energy (HFE), HFE1 (F11) determines the frequency above which only 15% of the total energy resides and HFE2 (F12) calculates the amount of energy present above a frequency of 1500 Hz. We apply the pitch tracking algorithm from [9] and compute the mean (F13) and standard deviation (F14) of the resulting series of pitches to determine the amplitudes of the first six harmonics (F15). As a potential discriminator for the breathy voice, the harmonic-to-noise ratio (HNR) designed for speech signals [8] is extracted for the frequencies below 500 (F16), 1500 (F17), 2500 (F18), 3500 (F19) and 4500 (F20) Hz. Using LPC with a filter of order $\frac{f}{1000} + 2$ and f as the sampling frequency in Hz, we retain the amplitudes (F21), frequencies (F22) and bandwidths (F23) of the first four formants. We further include CPP (F24), NAQ (F25) and MDQ (F26) introduced in section 2. Finally, the peak slope is computed by determining the slope of a regression line that is fitted to the peaks in the spectrum of the audio signal (F27) and the glottal waveform (F28) obtained by the *iterative adaptive inverse filtering algorithm* [2].

5. FEATURE ANALYSIS

5.1 MFCC Visualisation

In contrast to most of the other features listed in table 1, MFCCs (F1-F4) can be difficult to interpret. To make sense of this high-dimensional feature space and how it potentially differentiates phonation modes, we normalise the coefficients in MFCC40B to have zero mean and unit standard deviation across the dataset. The resulting coefficients are visualised in Figure 1, where the recordings on the horizontal axis are grouped by phonation mode and sorted in ascending order of pitch within each group. Within each phonation mode, multiple diagonal lines extending across the 10-th and higher coefficients imply a dependency of these feature coefficients on pitch, which a classifier would have to account for to reach high accuracy. The first 10 coefficients on the other hand do not exhibit this behaviour and also partly differ between phonation modes, especially when comparing breathy to non-breathy phonation. In particular, MFCC40B(0) as a measure of average energy increases in value from breathy, neutral, pressed to flow phonation. Although this visualisation does not reveal dependencies on vowel, it demonstrates the importance of the lower MFCCs and motivates the usage of MFCC80B, as more Mel bands increase the resolution in this range.

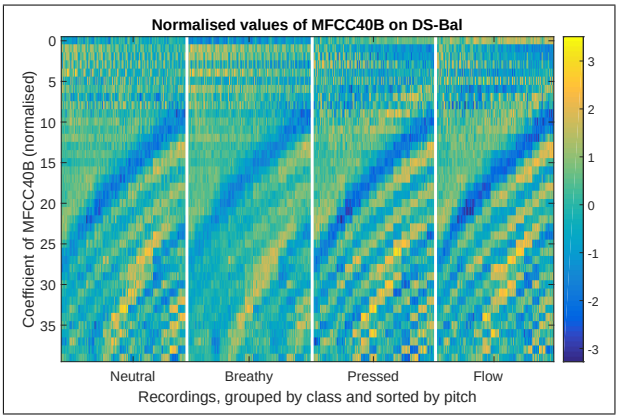


Figure 1. Visualisation of normalised MFCC40B values. Vertical gaps separate the different phonation modes. For each phonation mode, corresponding recordings are sorted in ascending order of pitch.

5.2 Class separation

In this section, we will investigate whether individual feature coefficients can directly separate some of the phonation modes by using an *analysis of variance* (ANOVA). Assuming that of all MFCC variants, MFCC80B is best suited for phonation mode detection, we subject MFCC80B and all remaining features (F5-F28) to an ANOVA with feature coefficients as dependent variables and the four phonation modes as independent categorical variables. The resulting *F-Ratio* equates to the ratio of variance between classes to the variance within classes, indicating how clearly phonation modes are separated by a particular feature.

The ten features with the highest resulting F-Ratios in descending order from $F = 441$ to $F = 213$ are CPP, temporal flatness, MFCC80B(0), MFCC80B(1), spectral flatness, HFE1, MDQ, spectral flux mean and deviation, and MFCC80B(3). However, these features are all mutually correlated with absolute correlation coefficients above 0.5 (mean: 0.77), indicating a large degree of redundancy.

In Figure 2, the distribution of feature values depending on phonation mode is shown for CPP, temporal flatness and MFCC80B(0) as they reach the highest F-Ratios, and for MDQ because it correlates least with the three aforementioned features. Figure 2 (a) demonstrates that CPP separates not only breathy from all other phonation modes significantly, as expected due to its design as a measure of breathiness [12], but can also distinguish neutral from pressed and flow phonation and to some degree pressed from flow phonation. Regardless of its simplicity, temporal flatness shown in Figure 2 (b) manages to clearly separate neutral and breathy from pressed and flow phonation. MFCC80B(0) shown in Figure 2 (c) confirms the finding from section 5.1 that each phonation mode features a different loudness on average. Interestingly, MDQ plotted in Figure 2 (d) behaves similar to temporal flatness shown in Figure 2 (b) and does not separate the classes more clearly despite its comparatively complex design intended to directly quantify the degree of pressedness.

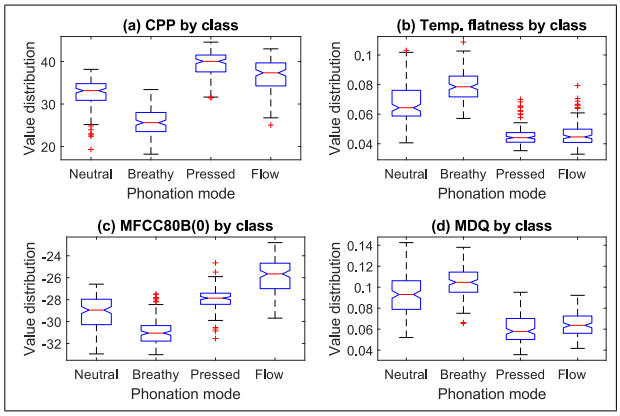


Figure 2. Distributions of (a) CPP, (b) temporal flatness, (c) MFCC80B(0) and (d) MDQ for the four phonation modes.

Regarding the other features, HNR behaves as expected and successfully separates breathy phonation from all other phonation modes, with a cut-off of 2500 Hz (F18) achieving the best F-Ratio of 97.8, but does not differentiate between the remaining phonation modes. Contrary to its purpose of estimating pressedness, NAQ surprisingly exhibits only small differences in mean values and large overlaps of the distributions between different phonation modes ($F = 6.48$), possibly because it was originally proposed for speech [3]. Apart from slightly higher values of peak slope for breathy voices, the feature proves to be uninformative ($F = 22.75$), despite obtaining good results on speech excerpts [14]. Finally, distributions of glottal peak slope for the phonation modes are not significantly different ($F = 0.42$).

In general, separating breathy and neutral phonation from pressed and flow phonation is more readily achieved by individual features than distinguishing pressed from flow phonation. Therefore, we perform the same analysis with only pressed and flow phonation as possible categories of the independent variable to find features that make this particularly difficult distinction. As a result, MFCC80B(0) achieves by far the largest separation ($F = 183.23$), which is hinted at in Figure 2 (c), followed by MFCC80B(1) ($F = 44.51$). Apart from CPP ($F = 38.58$) shown in Figure 2 (a) and MFCC80B(10) ($F = 31.72$), all remaining features exhibit F-Ratios below 15.

5.3 Robustness against pitch and vowel changes

We investigate the robustness of individual features against changes of pitch and vowel by performing an ANOVA with pitch and vowel respectively as independent variables, with one class for each unique pitch or vowel present in the dataset. As well as the formant-based features (F21-F23), the lower MFCC80B coefficients between approximately 4 and 17 are dependent on vowel, a dependency not immediately visible in Figure 1. HFE2 with an F-Ratio of 32.03 is more dependent on vowel than the alternative HFE1 feature ($F = 9.16$), further corroborating the superiority of HFE1 over HFE2 for phonation mode detection. Other particularly vowel-dependent features are the amplitude of the third harmonic ($F = 26.68$) and peak slope ($F = 45.04$). Regarding pitch, dependencies were found in MFCC80B confirming the interpretation of Figure 1, starting with coefficient 18 and increasing in F-Ratio until coefficient 30, where it remains constant for the coefficients 30 to 40. Except for F0 Mean as an estimate of pitch, no other significant dependencies were found, allowing for the construction of a classifier that is mostly robust against pitch changes.

5.4 A simple rule set to explain phonation modes

In this section, possible interactions between features that could explain differences in the phonation modes are analysed to derive a comprehensible rule set that correctly categorises most of the recordings. We construct a decision tree with *Gini's diversity index* [6] as split criterion and prune it so it has only three decision nodes. The result is the following set of rules using only temporal flatness, CPP and the MFCC80B(0) for distinguishing the phonation modes:

- Neutral and breathy phonation have higher temporal flatness (greater than $0.055 = 47$th percentile) than pressed and flow phonation
- Neutral phonation has higher CPP (greater than $29.97 = 30$th percentile) than breathy phonation
- Flow phonation has a higher MFCC80B(0) (greater than $-26.37 = 84$th percentile) than pressed

The above rules assign the correct class to 78% of the recordings in the dataset, thus offering a simple explanation for the main differences between the phonation modes.

6. CLASSIFICATION

6.1 Feature Sets

The eight feature sets listed in table 2 are constructed for training the classifier. The first four feature sets exclusively use the MFCC variants (F1-F4) from section 4. FS5 contains all features except the MFCC variants (F1-F4), while FS6 combines the MFCC variant yielding the best classification accuracy (F2) with all other features (F5-F28).

In the search for a low-dimensional feature representation, we apply *Principal component analysis* (PCA) to the features in FS6. The resulting principal components sorted in descending order of their eigenvalues constitute feature set FS7, for which classification performance will be assessed when including only the first D dimensions. Principal components can be difficult to interpret, because each represents a combination of different features. Therefore, we employ a feature selection method based on mutual information [19] to retrieve a ranking of the dimensions in FS6, enabling the construction of an optimal feature set of dimensionality D with the D highest-ranked feature coefficients. As a result, FS8 contains all feature dimensions from FS6 sorted in descending order of rank.

Name	List of features	Dimensions
FS1	MFCC40B (F1)	40
FS2	MFCC80B (F2)	40
FS3	MFCC80B0 (F3)	40
FS4	MFCC80BT (F4)	40
FS5	Features 5 to 28	38
FS6	FS2 and FS5	78
FS7	FS6, PCA-transformed	78
FS8	FS6, sorted by feature selection	78

Table 2. Feature sets used for classification.

6.2 Method

Feed-forward neural networks (NNs) are used for classification, as they are robust against noise and correlated inputs. We use one hidden layer with a variable number of neurons N, and a *soft-max* output layer. Cross-validation is employed that splits the dataset into 10 evenly distributed subsets, using every combination of two subsets as test and validation set with the remaining 8 subsets as training data, resulting in $10 \cdot 9$ iterations. For training, *stochastic gradient descent* is used to minimise *cross-entropy error* after semi-randomly initialising the network weights with the *Nguyen-Widrow initialisation method* [18].We describe the overall performance with the mean F-measure obtained over all cross-validation iterations.

To find the optimal number of hidden neurons N for every feature set, we determine the mean F-measure achieved for every $N \in \{1, \ldots, 40\}$. To obtain a compact set of features that yields high accuracy, we also optimise the mean F-measure achieved when using only the first D dimensions in the feature sets FS1 to FS4 as well as FS7 and FS8, resulting in a grid search with the number of neurons N and the number of features D as parameters.

6.3 Results

The classification results obtained with varying numbers of hidden neurons N and dimensions D using the feature sets FS1, FS2, FS7 and FS8 are visualised in Figure 3. We excluded the feature sets FS3 and FS4 due to their similar behaviour compared to FS1 and FS2, and FS5 as well as FS6 because only the number of neurons N was varied. With $N < 4$ neurons in the hidden layer, mean F-measures remain at low levels for every feature set regardless of the number of dimensions. Performance with more neurons N improves gradually when using an increasing number of MFCCs, as Figures 3 (a) and (b) demonstrate. The rate of this increase becomes less pronounced for higher MFCCs, implying that the differences in phonation mode are mostly encoded by approximately the first 20 MFCCs. FS2 containing MFCC80B shown in Figure 3 (b) however reaches significantly higher mean F-measures with the same number of coefficients than FS1 comprised of MFCC40B in Figure 3 (a). An increased frequency resolution of the cepstrum representation could be an explanation, as it leads to a more precise description of the relevant low-frequency components in the spectrum. Applying PCA does not lead

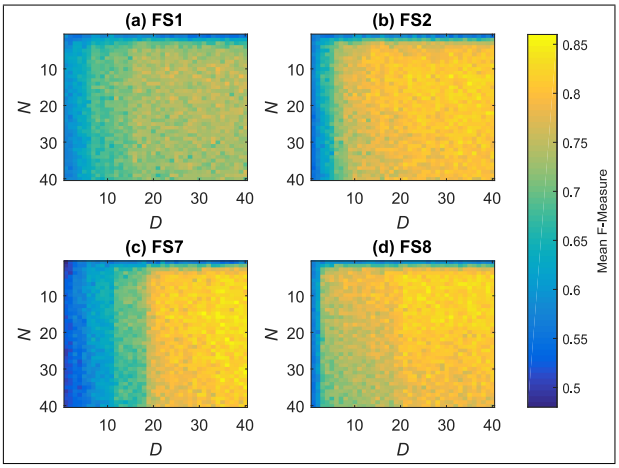


Figure 3. Mean F-measures when using a different number of neurons N and the first D dimensions in the feature sets (a) FS1, (b) FS2, (c) FS7 and (d) FS8.

to a drastically reduced dimensionality of the feature space without a major degradation in performance: Including the first D principal components of feature set FS7 only results in moderate performance for $D < 19$. One reason could be an intrinsically high dimensionality of the feature space, corroborated by the requirement of 32 principal components to explain 95% of the variance. Additionally, the first principal components could encode mostly pitch- and vowel-dependent variances in feature values instead of changes induced by different phonation modes. In contrast, feature selection considers how informative each feature dimension is for classification. As a result, including only the first few dimensions of FS8, which were ranked highest by feature selection, yields high mean F-measures as shown in Figure 3 (d).

Generally, the mean F-measure is subject to considerable variance due to the random selection of subsets performed by cross-validation. Because this impedes the robust selection of the optimal parameters, we interpolate the mean F-measures using *locally weighted regression* [7] with a *span* of 0.1, meaning 10% of the data points along each dimension nearest to an interpolated point determine its position. Intended as a trade-off between classification accuracy and model complexity, we define the optimal combination of parameters N and D as

$$(N_{\text{opt}}, D_{\text{opt}}) = \underset{(N,D)}{\operatorname{argmin}}\{N + D \mid (N, D) \in \mathcal{C}\}, \quad (1)$$

where $\mathcal{C}$ is the set of parameter configurations for which neither adding a dimension nor a hidden neuron increases the smoothed F-measure $s(N, D)$ more than a threshold t:

$$\mathcal{C} = \{(N, D) \mid s(N + 1, D) - s(N, D) < t \quad (2)$$
$$\wedge\, s(N, D + 1) - s(N, D) < t\}. \quad (3)$$

For $t = 0.001$, the mean F-measures for the optimised parameter settings are shown in table 3, including the 95% confidence for the maximum deviation of the mean in both directions, calculated as 1.96 times the standard error of

Feature set	N_{opt}	D_{opt}	Mean F-m.	$1.96 \cdot$ SEM
FS1	10	18	0.7403	0.027
FS2	8	15	0.7965	0.026
FS3	12	17	0.7948	0.027
FS4	9	21	0.7358	0.028
FS5	9	-	0.7681	0.023
FS6	9	-	0.8501	0.024
FS7	9	26	0.8050	0.025
FS8	9	24	0.8302	0.026

Table 3. Classification results for each feature set after optimising the number of neurons N and dimensions D.

the mean (SEM), to determine whether two classification results are significantly different from each other. For every feature set, at least moderate performance is achieved with only a low number of neurons. The usage of 80 Mel bands in MFCC80B (FS2) results in a significantly higher F-measure compared to the standard MFCC40B feature (FS1) with 40 Mel bands. Although FS3 exhibits lower performance with only very few coefficients, removing MFCC80B(0) does not decrease accuracy compared to FS2, perhaps because its correlation with higher MFCCs can be used to gain very similar information. The MFCCs appear to capture relevant timbral information present at the vowel onsets and releases, as the decreased performance for FS4 using the trimmed recordings shows. The interpretable, 38-dimensional feature set FS5 obtains moderate accuracy, but the best mean F-measure of 0.8501 is reached with the 78-dimensional feature set FS6. Application of PCA (FS7) and feature selection based on mutual information (FS8) on feature set FS6 lead to only slightly reduced performance with fewer dimensions. Feature selection is particularly successful, allowing us to construct a NN with only four hidden neurons and the MFCC80B(0), MFCC80B(1), MFCC80B(12), spectral flux mean and deviation, HNR 3500 and temporal flatness as features that still achieves a mean F-measure of 0.78 (SEM = 0.027).

To compare performance, we use the full dataset DS-Full on which the best mean F-measure of 0.84 was achieved by [13], and train a NN in the same manner as described in section 6.2. FS6 is chosen for this experiment, as it exhibits the best performance on the dataset DS-Bal. With $N = 13$ neurons, a mean F-measure of 0.868 with an SEM of 0.015 is obtained, leading to a 95% confidence interval of $[0.846, 0.890]$ for the F-measure and proving a significant improvement over the previously achieved mean F-measure of 0.84.

7. DISCUSSION AND OUTLOOK

Although designed specifically to only determine the amount of breathiness, CPP manages to separate each phonation mode best out of all features ($F = 441$). MDQ reliably distinguishes pressed and flow phonation from neutral and breathy phonation ($F = 310$), but has a correlation of 0.66 with temporal flatness, which achieves a better direct class separation ($F = 394$) with a simpler approach. Peak slope ($F = 22.75$) and glottal peak slope ($F = 0.65$) show weak discriminative power, in contrast to previous work [14]. The same applies to NAQ ($F = 6.48$), which contradicts previous literature demonstrating its suitability to measure the degree of pressedness [1, 3, 24] and warrants further investigation.

Contrary to the assumption in [20] that MFCCs are inapt for phonation mode classification, the lower coefficients alone lead to commensurate performance despite their dependence on vowel. Our classifier is mostly able to account for these effects, but an investigation into how class separation is exactly achieved, for example by using rule extraction from NNs [4], remains for future work. The increase in performance with MFCCs when including the full recording (MFCC80B) instead of an excerpt (MFCC80BT) demonstrates the relevance of timbral information at the vowel onset and release, but a more detailed analysis is needed to find the underlying cause. We show that using 80 Mel bands further increases performance, revealing the importance of optimising this parameter in future work. Every phonation mode features a different loudness, as indicated by MFCC80B(0) as a measure of average energy ($F = 352$). Loudness could vary strongly in more realistic singing conditions and between different singers, therefore making loudness-based phonation mode detection not very generalisable and adaptive to other scenarios.

Overall, the dataset has severe limitations, which reduces the generalisability of classifiers trained on this data: Because it only contains one singer, detection could be using singer-specific effects leading to decreased performance when confronted with other singers. Classification also has to be extended to work on full recordings of vocal performances instead of only isolated vowels. Finally, the recordings in the dataset are monophonic unlike many real-world music pieces, for which performance could be reduced due to the additionally required singing voice separation. Considering this is the only publically available dataset known to the authors that includes annotations of phonation mode, the development of larger, more comprehensive datasets for phonation mode detection seems critical for future progress on this task.

8. CONCLUSION

In this paper, we investigated the discriminative and explanatory power of a large number of features in the context of phonation mode detection. CPP, temporal flatness and MFCC80B(0) representing average energy were found to separate the phonation modes best, as they can correctly explain the phonation mode present in 78% of all recordings. Contrary to previous work, NAQ, peak slope and glottal peak slope did not separate phonation modes well. MFCCs lead to good classification accuracy using NNs as shown in section 6, particularly when using 80 instead of 40 Mel bands. The highest mean F-measure of 0.85 is achieved on the balanced dataset DS-Bal when using all features, demonstrating their explanatory power and the success of our classifier. On the dataset DS-Full, we attain an F-measure of 0.868, thereby significantly outperforming the best classifier from previous work [13].

9. REFERENCES

[1] Matti Airas and Paavo Alku. Comparison of multiple voice source parameters in different phonation types. In *8th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1410–1413, 2007.

[2] Paavo Alku. An automatic method to estimate the time-based parameters of the glottal pulseform. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 29–32, 1992.

[3] Paavo Alku, Tom Bäckström, and Erkki Vilkman. Normalized amplitude quotient for parametrization of the glottal flow. *The Journal of the Acoustical Society of America*, 112(2):701–710, 2002.

[4] Robert Andrews, Joachim Diederich, and Alan B. Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*, 8(6):373–389, 1995.

[5] Daniel Z. Borch and Johan Sundberg. Some phonatory and resonatory characteristics of the rock, pop, soul, and swedish dance band styles of singing. *Journal of Voice*, 25(5):532 – 537, 2011.

[6] Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. *Classification and regression trees*. CRC press, 1984.

[7] William S. Cleveland and Susan J. Devlin. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American statistical association*, 83(403):596–610, 1988.

[8] Guus de Krom. A cepstrum-based technique for determining a harmonics-to-noise ratio in speech signals. *Journal of Speech, Language, and Hearing Research*, 36(2):254–266, 1993.

[9] Thomas Drugman and Abeer Alwan. Joint robust voicing detection and pitch estimation based on residual harmonics. In *12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1973–1976, 2011.

[10] Emil Froeschels. Hygiene of the voice. *Archives of Otolaryngology*, 38(2):122–130, 1943.

[11] Elizabeth U. Grillo and Katherine Verdolini and. Evidence for distinguishing pressed, normal, resonant, and breathy voice qualities by laryngeal resistance and vocal efficiency in vocally trained subjects. *Journal of Voice*, 22(5):546 – 552, 2008.

[12] James Hillenbrand, Ronald A. Cleveland, and Robert L. Erickson. Acoustic correlates of breathy vocal quality. *Journal of Speech, Language, and Hearing Research*, 37(4):769–778, 1994.

[13] Léonidas Ioannidis, Jean-Luc Rouas, and Myriam Desainte-Catherine. Caractérisation et classification automatique des modes phonatoires en voix chantée. In *XXXèmes Journées d'études sur la parole*, 2014.

[14] John Kane and Christer Gobl. Identifying regions of non-modal phonation using features of the wavelet transform. In *12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 177–180, 2011.

[15] John Kane and Christer Gobl. Wavelet maxima dispersion for breathy to tense voice discrimination. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(6):1170–1179, 2013.

[16] Marko Lugger and Bin Yang. The relevance of voice quality features in speaker independent emotion recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages IV–17–IV–20, April 2007.

[17] Moa Millgård, Tobias Fors, and Johan Sundberg. Flow glottogram characteristics and perceived degree of phonatory pressedness. *Journal of Voice. Article in press. DOI: http://dx.doi.org/10.1016/j.jvoice.2015.03.014*, 2015.

[18] Derrick Nguyen and Bernard Widrow. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In *IJCNN International Joint Conference on Neural Networks*, pages 21–26 vol.3, June 1990.

[19] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[20] Polina Proutskova, Christophe Rhodes, Tim Crawford, and Geraint Wiggins. Breathy, resonant, pressed automatic detection of phonation mode from audio recordings of singing. *Journal of New Music Research*, 42(2):171–186, 2013.

[21] Polina Proutskova, Christophe Rhodes, Geraint A. Wiggins, and Tim Crawford. Breathy or resonant - A controlled and curated dataset for phonation mode detection in singing. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, pages 589–594, 2012.

[22] Johan Sundberg. *The science of the singing voice*. Illinois University Press, 1987.

[23] Johan Sundberg. What's so special about singers? *Journal of Voice*, 4(2):107 – 119, 1990.

[24] Johan Sundberg, Margareta Thalén, Paavo Alku, and Erkki Vilkman. Estimating perceived phonatory pressedness in singing from flow glottograms. *Journal of Voice*, 18(1):56–62, 2004.

ANALYSIS OF VOCAL IMITATIONS OF PITCH TRAJECTORIES

Jiajie Dai, Simon Dixon
Centre for Digital Music, Queen Mary University of London, United Kingdom
{j.dai, s.e.dixon}@qmul.ac.uk

ABSTRACT

In this paper, we analyse the pitch trajectories of vocal imitations by non-poor singers. A group of 43 selected singers was asked to vocally imitate a set of stimuli. Five stimulus types were used: a constant pitch (*stable*), a constant pitch preceded by a pitch glide (*head*), a constant pitch followed by a pitch glide (*tail*), a pitch *ramp* and a pitch with *vibrato*; with parameters for main pitch, transient length and pitch difference. Two conditions were tested: singing simultaneously with the stimulus, and singing alternately, between repetitions of the stimulus. After automatic pitch-tracking and manual checking of the data, we calculated intonation accuracy and precision, and modelled the note trajectories according to the stimulus types. We modelled pitch error with a linear mixed-effects model, and tested factors for significant effects using one-way analysis of variance. The results indicate: (1) Significant factors include stimulus type, main pitch, repetition, condition and musical training background, while order of stimuli, gender and age do not have any significant effect. (2) The *ramp*, *vibrato* and *tail* stimuli have significantly greater absolute pitch errors than the *stable* and *head* stimuli. (3) Pitch error shows a small but significant linear trend with pitch difference. (4) Notes with shorter transient duration are more accurate.

1. INTRODUCTION

Studying the vocal imitations of pitch trajectories is extremely important because most of the human produce a musical tone by imitation rather than absolute. Only .01% of the general population can produce a musical tone without the use of an external reference pitch [22]. Although sing in tone is the primary element of singing performance, the research of vocal imitations with unstable stimuli has not been explored. It is significant to distinguish the influence factors and to quantise them, fill the gap between response and stimuli, as well as create knowledge to help the future music education and entertainment.

The accuracy of pitch in playing or singing is called intonation [8, 20]. Singing in tune is extremely important for solo singers and choirs because they must be accurate and blend well with accompaniments and other vocal parts [1]. However, it is a practical challenge when the singers have to sing with an unstable reference pitch or other vocal parts without instrumental accompaniment [17, Ch. 12, p. 151]. Nevertheless, most singers rely on their sense of relative pitch and their teammates who provide reference pitches which help them maintain tuning, as the initial tonal reference can be forgotten over time [9, 11]. Pfordresher *et al.* [16] distinguish between pitch accuracy, the average difference between the sung pitch and target pitch, and pitch precision, the standard error of sung pitches.

As for vocal reference pitch (stimulus of imitation in this paper), it usually does not have a fixed pitch for each note which is different from percussion instruments with a stable shape [4, 7, 11]. Instead, vocal notes typically fluctuate around the target pitch. When singing with a stable reference pitch, the singer will voluntarily adjust their vocal output until the auditory feedback matches the intended note [28]. This adjustment especially at the beginning of the note, they may sing with vibrato, and they may not sustain the pitch at the end of the note [27]. Although singers make fewer errors when singing in unison or with stable accompaniment [24], the response of unstable stimulus or notes with transient parts is still obscure.

A transient is part of a signal (often at the beginning) during which its properties are rapidly changing and thus unpredictable. For most musical tones, a short transient segment is followed by a much longer steady state segment, but for singing, such a segmentation is difficult, as the signal never reaches a steady state. At the beginning of a tone, a pitch glide is often observed as the singer adjusts the vocal cords from their previous state (the previous pitch or a relaxed state). Then the pitch is adjusted as the singer uses perceptual feedback to correct for any error in the pitch. Possibly at the same time, vibrato may be applied, which is an oscillation around the central pitch, which is close to sinusoidal for skilled singers, but asymmetric for unskilled singers [7]. At the end of the tone, the pitch often moves in the direction of the following note, or downward (toward a relaxed vocal cord state) if there is no immediately following note.

To investigate the response of singers to time-varying pitch trajectories, we prepared a controlled experiment using synthetic stimuli, in order to test the following hypotheses:

- The stimulus type will have a significant effect on intonation accuracy.
- A greater duration or extent of deviation from the

© Jiajie Dai, Simon Dixon. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jiajie Dai, Simon Dixon. "Analysis of vocal imitations of pitch trajectories", 17th International Society for Music Information Retrieval Conference, 2016.

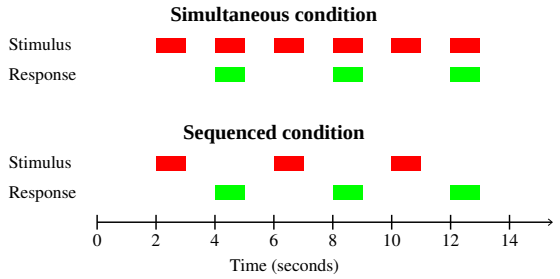


Figure 1: Experimental design showing timing of stimuli and responses for the two conditions.

main pitch will increase intonation error.

- The direction of any deviation in the stimulus from the main pitch determines the direction of any error in the response.
- Singing simultaneously with the stimulus will result in a lower error than alternating the response with the stimulus.

We extract the fundamental frequency (f_0) [5, 10] and convert to a logarithmic scale, corresponding to non-integer numbers of equal-tempered semitones from the reference pitch (A4, 440Hz). We model responses according to stimulus types in order to compute the parameters of observed responses. The significance of factors (stimulus type, stimulus parameters and order of stimuli, as well as participants' musical background, gender and age) was evaluated by analysis of variance (ANOVA) and linear mixed-effects models.

2. MATERIALS AND METHODS

2.1 Experimental Design

The experiment consisted of 75 trials in each of two conditions. In each trial, the participant imitated the stimulus three times (see Figure 1). Each stimulus was one second in duration. In the *simultaneous* condition, the stimulus was repeated six times, with one second of silence between the repetitions, and the participants sang simultaneously with the 2nd, 4th and 6th instances of the stimulus. The *sequenced* condition was similar in that the responses occurred at the same times as in the simultaneous case, but the stimulus was not played at these times. There was a three second pause after each trial. The trials of a given condition were grouped together, and participants were given visual prompts so that they knew when to respond. Each of the 75 trials within a condition used a different stimulus, taken from one of the five stimulus types described in Section 2.2, and presented in a random order. The two conditions were also presented in a random order.

2.2 Stimuli

Unlike previous imitation experiments which have used fixed-pitch stimuli, our experimental stimuli were synthesised from time-varying pitch trajectories in order to provide controlled conditions for testing the effect of specific deviations from constant pitch. Five stimulus types were chosen, representing a simplified model of the components of sung tones (constant pitch, initial and final glides, vibrato and pitch ramps). The pitch trajectories of the stimuli were generated from the models described below and synthesised by a custom-made MATLAB program, using a monotone male voice on the vowel /a:/.

The five different stimulus types considered in this work are: constant pitch (*stable*), a constant pitch preceded by an initial quadratic pitch glide (*head*), a constant pitch followed by a final quadratic pitch glide (*tail*), a linear pitch ramp (*ramp*), and a pitch with sinusoidal vibrato (*vibrato*). The stimuli are parametrised by the following variables: p_m, the main or central pitch; d, the duration of the transient part of the stimulus; and p_D, the extent of pitch deviation from p_m. For *vibrato* stimuli, d represents the period of vibrato. Values for each of the parameters are given in Table 1 and the text below.

By assuming an equal tempered scale with reference pitch A4 tuned to 440 Hz, pitch p and fundamental frequency f_0 can be related as follows [11]:

$$p = 69 + 12 \log_2 \frac{f_0}{440} \tag{1}$$

such that for integer values of p the scale coincides with the MIDI standard. Note that pitch is not constrained to integer values in this representation.

For the *stable* stimulus, the pitch trajectory $p(t)$ is defined as follows:

$$p(t) = p_m, \qquad 0 \leqslant t \leqslant 1. \tag{2}$$

The *head* stimulus is represented piecewise by a quadratic formula and a constant:

$$p(t) = \begin{cases} at^2 + bt + c, & 0 \leqslant t \leqslant d \\ p_m, & d < t \leqslant 1. \end{cases} \tag{3}$$

The parameters a, b and c are selected to make the curve pass through the point $(0, p_m + p_D)$ and have its vertex at (d, p_m). The *tail* stimulus is similar, with $p(t) = p_m$ for $t < 1 - d$, and the transient section being defined for $1 - d \leqslant t \leqslant 1$. In this case the parameters a, b and c are chosen so that the curve has vertex $(1 - d, p_m)$ and passes through the point $(1, p_m + p_D)$.

The *ramp* stimuli are defined by:

$$p(t) = p_m + p_D \times (t - 0.5), \qquad 0 \leqslant t \leqslant 1. \tag{4}$$

Finally, the equation of *vibrato* stimuli is:

$$p(t) = p_m + p_D \sin\left(\frac{2\pi t}{d}\right), \qquad 0 \leqslant t \leqslant 1. \tag{5}$$

There is a substantial amount of data on the fundamental frequency of the voice in the speech of speakers who differ in age and sex [23]. We chose three pitch values according to gender to fall within a comfortable range for most singers. The pitches C3 ($p = 48$), F3 ($p = 53$) and B♭3

($p = 58$) were chosen for male singers and C4 ($p = 60$), F4 ($p = 65$) and B♭4 ($p = 70$) for female singers. For the *vibrato* stimuli, we set the vibrato rate according to a reported mean vibrato rate across singers of 6.1 Hz [18], and the extent or depth of vibrato to ±0.25 or 0.5 semitones, in accordance with values reported by [21]. Because intonation accuracy is affected by the duration of the note [4, 6], we used a fixed one-second duration for all stimuli in this experiment.

Table 1: Parameter settings for each stimulus type. The octave for the pitch parameter was dependent on sex (3 for male, 4 for female).

Type	p_m	d	p_D	Count
stable	{C, F, B♭}	{0.0}	{0.0}	3
head	{C, F, B♭}	{0.1, 0.2}	{±1, ±2}	24
tail	{C, F, B♭}	{0.1, 0.2}	{±1, ±2}	24
ramp	{C, F, B♭}	{1.0}	{±1, ±2}	12
vibrato	{C, F, B♭}	{±0.32}	{0.25, 0.5}	12

2.3 Participants

A total of 43 participants (27 female, 16 male) took part in the experiment. 38 of them were recorded in the studio and 5 were distance participants from the USA, Germany, Greece and China (2 participants). The range of ages was from 19 to 34 years old (mean: 25.1; median: 25; std.dev.: 2.7). Apart from 3 participants who did not complete the experiment, most singers recorded all the trials.

We intentionally chose non-poor singers as our research target. "Poor-pitch singers" are defined as those who have a deficit in the use of pitch during singing [15, 25], and are thus unable to perform the experimental task. Participants whose pitch imitations had on average at least one semitone absolute error were categorised as poor-pitch singers. The data of poor-pitch singers is not included in this study, apart from one singer who occasionally sang one octave higher than the target pitch.

Vocal training is an important factor for enhancing the singing voice and making the singer's voice different from that of an untrained person [12]. To allow us to test for the effects of training, participants completed a questionnaire containing 34 questions from the Goldsmiths Musical Sophistication Index [13] which can be grouped into 4 main factors for analysis: active engagement, perceptual abilities, musical training and singing ability (9, 9, 7 and 7 questions respectively).

2.4 Recording Procedure

A tutorial video was played before participation. In the video, participants were asked to repeat the stimulus precisely. They were not told the nature of the stimuli. Singers who said they could not imitate the time-varying pitch trajectory were told to sing a stable note of the same pitch.

The experimental task consisted of 2 conditions, each containing 75 trials, in which participants sang three one-second responses in a 16-second period. It took just over one hour for participants to finish the experiment. 22 singers took the simultaneous condition first and 21 singers took the sequenced condition first. Although the synthetic stimulus simulated the vowel /a:/, participants occasionally chose other vowels that felt comfortable.

We used an on-line system to record and manage the experiment. After sign-up, participants completed the unfinished tests guided by a graphical interface. After singing each trial, the system automatically uploaded the recordings to a server and the annotation results were simultaneously generated. All responses were labelled with singer ID, condition, trial, order and repetition.

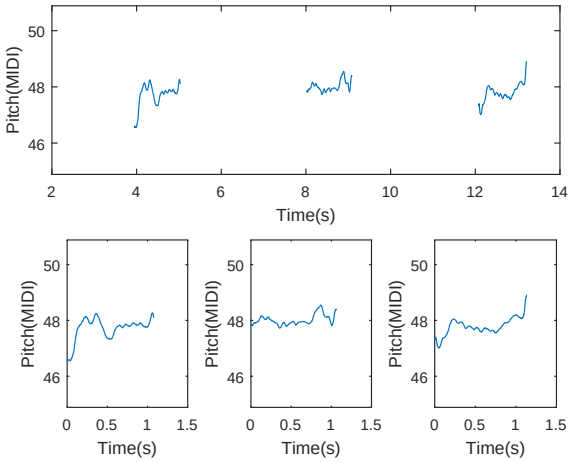


Figure 2: Example of extracted pitch and annotation for *head* stimulus ($p_m = 48$, $p_D = 1$, $d = 0.1$). The upper panel shows the results for pitch extraction by YIN, and the three lower panels show the segmented responses.

2.5 Annotation

Each recording file contains three responses, from which we extract pitch information using the YIN algorithm (version 28th July 2003) [5]. This outputs the pitch trajectory $p(t)$ from which we compute the median pitch $\bar{p}$ for each response. The segmentation into individual responses is based on the timing, pitch and power. If participants sang more than 3 repetitions we choose the three responses that have the longest duration and label them with the recording order. Any notes having a duration less than 0.1 seconds were excluded. Any remaining notes with a duration less than 0.4 seconds were flagged and checked manually. Most of these deficient notes were due to participants making no response. Figure 2 shows an example of pitch extraction and segmentation.

The main pitch $\bar{p}$ of response was calculated by removing the first 10% and last 10% of the response duration, and computing the median of the remaining pitch track. The pitch error e^p is calculated as the difference between the main pitch of the stimulus p_m and that of the response $\bar{p}$:

$$e^p = \bar{p} - p_m \tag{6}$$

For avoiding bias due to large errors we exclude any responses with $|e^p| > 2$ (4% of responses). Such errors arose

when participants sang the pitch of the previous stimulus or one octave higher than the stimulus. The resulting database contains 18572 notes, from which the statistics below were calculated.

The mean pitch error (MPE) over a number of trials measures the tendency to sing sharp (MPE > 0) or flat (MPE < 0) relative to the stimulus. The mean absolute pitch error (MAPE) measures the spread of a set of responses. These can be viewed respectively as inverse measures of accuracy and precision (cf. [16]).

To analyse differences between the stimulus and response as time series, pitch error $e^p_f(t)$ is calculated frame-wise: $e^p_f(t) = p_r(t) - p_s(t)$, for stimulus $p_s(t)$ and response $p_r(t)$, where the subscript f distinguishes frame-wise results. For frame period T and frame index i, $0 \leqslant i < M$, we calculate summary statistics:

$$\mathrm{MAPE}_f = \frac{1}{M} \sum_{i=0}^{M-1} |e^p_f(iT)| \tag{7}$$

and MPE_f is calculated similarly. Equation 7 assumes that the two sequences $p_r(t)$ and $p_s(t)$ are time-aligned. Although cross-correlation could be used to find a fixed offset between the sequences, or dynamic time warping could align corresponding features if the sequences proceed at different or time-varying rates, in our case we consider singing with the correct timing to be part of the imitation task, and we align the stimulus to the beginning of the detected response.

3. RESULTS

We first report pitch error (MPE: 0.0123; std.dev.: 0.3374), absolute pitch error (MAPE: 0.2441; std.dev.: 0.2332) and frame-wise absolute pitch error (MAPE_f: 0.3968; std.dev.: 0.2238) between all the stimuli and responses. 71.1% of responses have an absolute error less than 0.3 semitones. 51.3% of responses are higher than the stimulus ($e^p > 0$). All the singers' information, questionnaire responses, stimulus parameters and calculated errors were arranged in a single table for further processing. We first analyse the factors influencing absolute pitch error in the next two subsections, and then consider pitch error in section 3.3 and the modelling of responses in the following two subsections.

3.1 Influence of stimulus type on absolute pitch error

We performed one-way independent samples analysis of variance (one-way ANOVA) with the fixed factor stimulus type (five levels: *stable*, *head*, *tail*, *ramp* and *vibrato*) and the random factor participant. There was a significant effect of stimulus type ($[F(4, 18567) = 72.3, p < .001]$).

Post hoc comparisons using the Tukey HSD test indicated that the absolute e^p for *tail*, *ramp* and *vibrato* stimuli were significantly different from that of the *stable* stimuli, while the *head* stimuli showed no significant difference from *stable* stimuli (see Table 2). Thus *tail*, *ramp* and *vibrato* stimuli do have an effect on pitch precision. Table 2 also shows the 95% confidence intervals for each stimulus type. Effect sizes were calculated by a linear mixed-effects model comparing with *stable* stimulus results.

Stimulus	MAPE	Confidence interval	Effect size
stable	0.1977	[0.1812, 0.2141]	–
head	0.1996	[0.1938, 0.2054]	0.2 cents
tail	0.2383	[0.2325, 0.2441]*	4.1 cents
ramp	0.3489	[0.3407, 0.3571]***	15.1 cents
vibrato	0.2521	[0.2439, 0.2603]***	5.5 cents

Table 2: Mean absolute pitch error (MAPE) and 95% confidence intervals for each stimulus type (***p $<$ .001; **p $<$ .01; *p $<$ .05).

3.2 Factors of influence for absolute pitch error

The participants performed a self-assessment of their musical background with questions from the Goldsmiths Musical Sophistication Index [14] covering the four areas listed in Table 3, where the general factor is the sum of other four factors. An ANOVA F-test found that all background factors are significant for pitch accuracy (see Table 3). The task involved both perception and production, so it is to be expected that both of these factors (perceptual and singing abilities) would influence results. Likewise most musical training includes some ear training which would be beneficial for this experiment.

Factor	Test Results
General factor	$F(30, 18541) = 54.4$ ***
Active engagement	$F(21, 18550) = 37.3$ ***
Perceptual abilities	$F(22, 18549) = 57.5$ ***
Musical training	$F(24, 18547) = 47.2$ ***
Singing ability	$F(20, 18551) = 69.8$ ***

Table 3: Influence of background factors.

We used R [19] and *lme4* [2] to perform a linear mixed-effects analysis of the relationship between factors of influence and $|e^p|$. The factors stimulus type, main pitch, age, gender, the order of stimuli, trial condition, repetition, duration of pitch deviation d, extent of pitch deviation p_D, observed duration and the four factors describing musical background were added separately into the model, and a one-way ANOVA between the model with and without the factor tested whether the factor had a significant effect. Table 4 shows the p-value of ANOVA results after adding each factor.

We created a fixed model with factors stimulus type, main pitch, repetition and trial condition. As a random effect, we had the factor of the singer. Visual inspection of residual plots did not reveal any obvious deviations from homoscedasticity or normality. The p-values were obtained by likelihood ratio tests of the full model with the effect in question against the model without the effect in question [26].

According to the modelling results on $|e^p|$, significant effects were found for the factors stimulus type, main pitch

Table 4: Significance and effect sizes for tested factors based on ANOVA results.

Factors	p-value	Effect size (cents)
Stimulus type	2.2e-16***	See Table 2
p_m	5.4e-7***	-0.19
Age	0.51	
Gender	0.56	
Order of stimuli	0.13	
Trial condition	2.2e-16***	3.2
Repetition	2.2e-16***	-1.8
Duration of transient d	2.2e-16***	11.4
sign(p_D)	5.1e-6***	0.8
abs(p_D)	8.3e-12***	1.9
Observed duration	3.3e-4***	-5.4
Active engagement	6.9e-2	
Perceptual abilities	0.04*	-0.3
Musical training	6.2e-5***	-0.5
Singing ability	8.2e-2	

p_m (effect size: -2.35 cents per octave), trial condition, repetition, musical background, duration of pitch deviation (effect size: 11.4 cents per second), direction of pitch deviation, magnitude of pitch deviation (effect size: 1.7 cents per semitone) and observed duration (effect size: -5.4 cents per second). The remaining factors (singer, age, gender and the order of stimuli) did not have any significant effect on $|e^p|$ in this model. The LME models gave different results for the background questionnaire factors than the one-way ANOVA, with only two of the factors, perceptual abilities and musical training, having a significant effect.

Contrary to our hypothesis, singing simultaneously (MAPE: 0.26; std.dev.: 0.25) is 3.2 cents less accurate than the sequenced condition (MAPE: 0.23; std.dev: 0.21). Despite the large spread of results, the standard errors in the means are small and the difference is significant. Recall also that responses with $|e^p|$ over 2 semitones were excluded.

Other significant factors were repetition, where we found that MAPE decreases 1.8 cents for each repetition (that is, participants improved with practice), and observed duration and main pitch, which although significant, had very small effect sizes for the range of values they took on.

3.3 Effect of pitch deviation on pitch error

We now look at specific effects on the direction of pitch error, to test the hypothesis that asymmetric deviations from main pitch are likely to lead to errors in the direction of the deviation. For the *stable*, *head* and *tail* stimuli, a correlation analysis was conducted to examine the relationship between pitch deviation and MPE. The result was significant on MPE ($F(4, 12642) = 8.4, p = 9.6e{-}7$) and MAPE ($F(4, 12642) = 8.2, p = 1.3e{-}6$). A significant regression equation was found, with $R^2 = 2.5e-3$, modelling pitch error as $e^P = 0.033 + 0.01 p_D$. Pitch error increased 1 cent for each semitone of p_D, a significant but small effect, as shown in Figure 3.

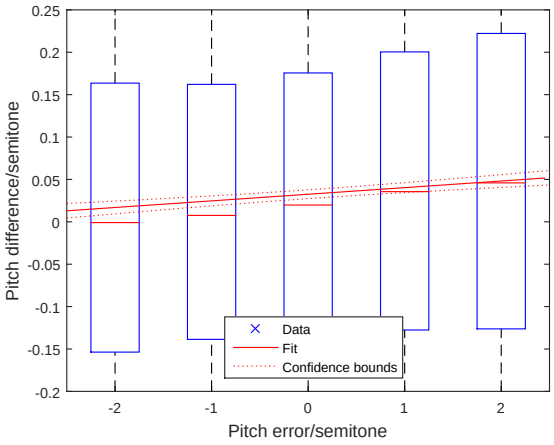


Figure 3: Boxplot of MPE for different p_D, showing median and interquartile range, regression line (red, solid) and 95% confidence bounds (red, dotted). The regression shows a small bias due to the positively skewed distribution of MPE.

3.4 Modelling

In this section, we fit the observed pitch trajectories to a model defined by the stimulus type, to better understand how participants imitated the time-varying stimuli. The *head* and *tail* stimuli are modelled by a piecewise linear and quadratic function. Given the break point, corresponding to the duration of the transient, the two parts can be estimated by regression. We perform a grid search on the break point and select the optimal parameters according to the smallest mean square error. Figure 4 shows an example of *head* response modelling.

The *ramp* response is modelled by linear regression. The model p_m of a *stable* response is the median of $p(t)$ for the middle 80% of the response duration. The *vibrato* responses were modelled with the MATLAB nlinfit function using Equation 5 and initialising the parameters with the parameters of the stimulus.

For the absolute pitch error between modelling results and stimuli, 66.5% of responses have an absolute error less than 0.3 semitones, while only 29.3% of trials have an absolute error less than 0.3 semitones between response and stimulus. We observed that some of the *vibrato* models did not fit the stimulus very well because the singer attempted to sing a stable pitch rather than imitate the intonation trajectory.

3.5 Duration of transient

As predicted, the duration d of the transient has a significant effect on MPE ($F(5, 18566) = 51.4, p < .001$). For the *stable*, *head* and *tail* stimuli, duration of transient influences MAPE ($F(2, 12644) = 31.5, p < .001$), where stimuli with smaller transient length result in lower MAPE. The regression equation is $MAPE = 0.33 + 0.23d$ with $R^2 = 0.208$. MAPE increased 23.2 cents for each second of transient. This matches the result from the linear mixed-effects model, where effect size is 23.8 cents per second.

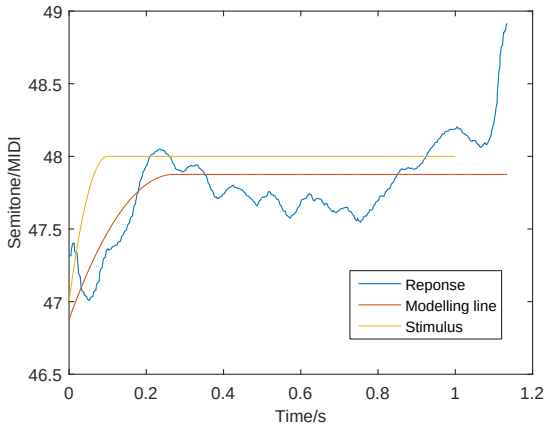


Figure 4: Example of modelling the response to a *head* stimulus with parameters $\mathrm{d} = 0.1$, $\mathrm{p_D} = -1$ and $\mathrm{p_m} = 48$. The response model has $\mathrm{d} = 0.24$, $\mathrm{p_D} = -0.997$ and $\mathrm{p_m} = 47.87$. The forced fit to the stimulus model treats as noise response features such as the final rising intonation.

Based on the modelling results, we observed that transient length in responses was longer than in the corresponding stimuli. 74.2% of *head* and *tail* responses have transient length longer than that of the stimulus. Stimulus transients are 0.1 or 0.2 seconds, but 65.5% of *head* and 72.0% of *tail* responses have a transient longer than 0.2 seconds.

4. DISCUSSION

Since we intentionally chose non-poor singers, most participants imitated with small error. 88.5% of responses were sung with intonation error less than half a semitone. The responses are characterised far more by imprecision than inaccuracy. That is, there is very little systematic error in the results (MPE $= 0.0123$), whereas the individual responses exhibit much larger errors in median pitch (MAPE $= 0.2441$) and on a frame-wise level within notes ($\mathrm{MAPE_f} = 0.3968$). The results for MAPE are within the range reported for non-poor singers attempting known melodies (19 cents [11], 28 cents [4]), and thus is better explained by limitations in production and perception rather than by any particular difficulty of the experimental task.

The *stable* stimuli gave rise to the lowest pitch errors, although the *head* responses were not significantly different. The larger errors observed for the *tail*, *ramp* and *vibrato* stimuli could be due to a memory effect. These three stimulus types have in common that the pitch at the end of the stimulus differs from $\mathrm{p_M}$. Thus the most recent pitch heard by the participant could distract them from the main target pitch. The *ramp* stimuli, having no constant or central pitch, was the most difficult to imitate, and resulted in the highest MAPE.

It was hypothesised that the simultaneous condition would be easier than the sequenced condition, as singing tends to be more accurate when accompanied by other singers or instruments. We propose two reasons why this experiment might be exceptional. Firstly, in the sequenced condition, the time between stimulus and response was short (1 second), so it would be unlikely that the participant would forget the reference pitch. Secondly, the stimulus varied more quickly than the auditory feedback loop, the time from perception to a change in production (around 100ms [3]), could accommodate. Thus the feedback acts as a distractor rather than an aid. Singing in practice requires staying in tune with other singers and instruments. If a singer takes their reference from notes with large pitch fluctuations, especially at their ends, this will adversely affect intonation.

5. CONCLUSIONS

We designed a novel experiment to test how singers respond to controlled stimuli containing time-varying pitches. 43 singers vocally imitated 75 instances of five stimulus types in two conditions. It was found that time-varying stimuli are more difficult to imitate than constant pitches, as measured by absolute pitch error. In particular, stimuli which end on a pitch other than the main pitch (*tail*, *ramp* and *vibrato* stimuli) had significantly higher absolute pitch errors than the *stable* stimuli, with effect sizes ranging from 15 cents (*ramp*) to 4.1 cents (*tail*).

Using a linear mixed-effects model, we determined that the following factors influence absolute pitch error: stimulus type, main pitch, trial condition, repetition, duration of transient, direction and magnitude of pitch deviation, observed duration, and self-reported musical training and perceptual abilities. The remaining factors that were tested had no significant effect, including self-reported singing ability, contrary to other studies [11].

Using one-way ANOVA and linear regression, we found a positive correlation between extent of pitch deviation (pitch difference, $\mathrm{p_D}$) and pitch error. Although the effect size was small, it was significant and of similar order to the overall mean pitch error. Likewise we observed that the duration d of the transient proportion of the stimulus correlated with absolute pitch error. Contrary to expectations, participants performed 3.2 cents worse in the condition when they sang simultaneously with the stimulus, although they also heard the stimulus between singing attempts, as in the sequenced condition.

Finally, we extracted parameters of the responses by a forced fit to a model of the stimulus type, in order to describe the observed pitch trajectories. The resulting parameters matched the stimuli more closely than the raw data did. Many aspects of the data remain to be explored, but we hope that the current results take us one step closer to understanding interaction between singers.

6. DATA AVAILABILITY

There is the tutorial video which show participants how to finish the experiment before they start: `https://www.youtube.com/watch?v=xadECsaglHk`. The annotated data and code to reproduce our results are available in an open repository at: `https://code.soundsoftware.ac.uk/projects/stimulus-intonation/repository`.

7. REFERENCES

[1] Per-Gunnar Alldahl. *Choral Intonation*. Gehrmans, 2008.

[2] Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software*, 67(1):1–48, 2015.

[3] T. A. Burnett, M. B. Freedland, C. R. Larson, and T. C. Hain. Voice F0 Responses to Manipulations in Pitch Feedback. *Journal of the Acoustical Society of America*, 103(6):3153–3161, 1998.

[4] Jiajie Dai, Matthias Mauch, and Simon Dixon. Analysis of Intonation Trajectories in Solo Singing. In *Proceedings of the 16th ISMIR Conference*, volume 421, 2015.

[5] Alain De Cheveigné and Hideki Kawahara. YIN, A Fundamental Frequency Estimator for Speech and Music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[6] J. Fyk. Pitch-matching Ability In Children As A Function of Sound Duration. *Bulletin of the Council for Research in Music Education*, pages 76–89, 1985.

[7] David Gerhard. Pitch Track Target Deviation in Natural Singing. In *ISMIR*, pages 514–519, 2005.

[8] Joyce Bourne Kennedy and Michael Kennedy. *The Concise Oxford Dictionary of Music*. Oxford University Press, 2004.

[9] Peggy A Long. Relationships Between Pitch Memory in Short Melodies and Selected Factors. *Journal of Research in Music Education*, 25(4):272–282, 1977.

[10] Matthias Mauch and Simon Dixon. pYIN: A Fundamental Frequency Estimator Using Probabilistic Threshold Distributions. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2014)*, 2014.

[11] Matthias Mauch, Klaus Frieler, and Simon Dixon. Intonation in Unaccompanied Singing: Accuracy, Drift, and a Model of Reference Pitch Memory. *The Journal of the Acoustical Society of America*, 136(1):401–411, 2014.

[12] Ana P Mendes, Howard B Rothman, Christine Sapienza, and WS Brown. Effects of Vocal Training on the Acoustic Parameters of the Singing Voice. *Journal of Voice*, 17(4):529–543, 2003.

[13] Daniel Müllensiefen, Bruno Gingras, Jason Musil, Lauren Stewart, et al. The Musicality of Non-musicians: An Index for Assessing Musical Sophistication in the General Population. *PloS one*, 9(2):e89642, 2014.

[14] Daniel Müllensiefen, Bruno Gingras, Lauren Stewart, and J Musil. The Goldsmiths Musical Sophistication Index (Gold-MSI): Technical Report and Documentation v0.9. *London: Goldsmiths, University of London. URL: http://www.gold.ac.uk/music-mind-brain/gold-msi*, 2011.

[15] Peter Q Pfordresher and Steven Brown. Poor-pitch Singing in the Absence of "Tone Deafness". *Music Perception: An Interdisciplinary Journal*, 25(2):95–115, 2007.

[16] Peter Q Pfordresher, Steven Brown, Kimberly M Meier, Michel Belyk, and Mario Liotti. Imprecise Singing is Widespread. *The Journal of the Acoustical Society of America*, 128(4):2182–2190, 2010.

[17] John Potter, editor. *The Cambridge Companion to Singing*. Cambridge University Press, 2000.

[18] Eric Prame. Measurements of the Vibrato Rate of Ten Singers. *The Journal of the Acoustical Society of America*, 96(4):1979–1984, 1994.

[19] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008.

[20] John Andrew Simpson, Edmund S.C. Weiner, et al. *The Oxford English Dictionary*, volume 2. Clarendon Press Oxford, 1989.

[21] J. Sundberg. Acoustic and Psychoacoustic Aspects of Vocal Vibrato. Technical Report STL-QPSR 35 (2–3), pages 45–68, Department for Speech, Music and Hearing, KTH, 1994.

[22] Annie H Takeuchi and Stewart H Hulse. Absolute Pitch. *Psychological bulletin*, 113(2):345, 1993.

[23] Hartmut Traunmüller and Anders Eriksson. The Frequency Range of the Voice Fundamental in the Speech of Male and Female Adults. *Consulté le*, 12(02):2013, 1995.

[24] Allan Vurma and Jaan Ross. Production and Perception of Musical Intervals. *Music Perception: An Interdisciplinary Journal*, 23(4):331–344, 2006.

[25] Graham F Welch. Poor Pitch Singing: A Review of the Literature. *Psychology of Music*, 7(1):50–58, 1979.

[26] Bodo Winter. Linear Models and Linear Mixed Effects Models in R with Linguistic Applications. *arXiv preprint arXiv:1308.5499*, 2013.

[27] Yi Xu and Xuejing Sun. How Fast Can We Really Change Pitch? Maximum Speed of Pitch Change Revisited. In *INTERSPEECH*, pages 666–669, 2000.

[28] Jean Mary Zarate and Robert J Zatorre. Experience-dependent Neural Substrates Involved in Vocal Pitch Regulation During Singing. *Neuroimage*, 40(4):1871–1887, 2008.

AUTOMATIC MUSIC RECOMMENDATION SYSTEMS: DO DEMOGRAPHIC, PROFILING, AND CONTEXTUAL FEATURES IMPROVE THEIR PERFORMANCE?

Gabriel Vigliensoni and Ichiro Fujinaga
Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
McGill University, Montréal, QC, Canada
[gabriel,ich]@music.mcgill.ca

ABSTRACT

Traditional automatic music recommendation systems' performance typically rely on the accuracy of statistical models learned from past preferences of users on music items. However, additional sources of data such as demographic attributes of listeners, their listening behaviour, and their listening contexts encode information about listeners, and their listening habits, that may be used to improve the accuracy of music recommendation models.

In this paper we introduce a large dataset of music listening histories with listeners' demographic information, and a set of features to characterize aspects of people's listening behaviour. The longevity of the collected listening histories, covering over two years, allows the retrieval of basic forms of listening context. We use this dataset in the evaluation of accuracy of a music artist recommendation model learned from past preferences of listeners on music items and their interaction with several combinations of people's demographic, profiling, and contextual features. Our results indicate that using listeners' self-declared *age*, *country*, and *gender* improve the recommendation accuracy by 8 percent. When a new profiling feature termed *exploratoryness* was added, the accuracy of the model increased by 12 percent.

1. LISTENING BEHAVIOUR AND CONTEXT

The context in which people listen to music has been the object of study of a growing number of publications, particularly coming from the field of music psychology. Konečni has suggested that the act of music listening has vacated the physical spaces devoted exclusively to music performance and enjoyment long ago, and that music nowadays is listened to in a wide variety of contexts [13]. As music increasingly accompanies our everyday activities, the music and the listener are not the only factors, as the context of listening has emerged as another variable that influences, and is influenced, by the other two factors [11]. It has been also observed that people consciously understand these interactions [6] and use them when choosing music for daily life activities [23]. The context of music listening seems to influence the way in which people chooses music, and so music recommenders should suggest music items to fit the situation and needs of each particular listener.

Modelling the user needs was identified by Schedl et al. as one key requirement for developing user-centric music retrieval systems [20]. They noted also that *personalized systems* customize their recommendations by using additional user information, and *context-aware systems* use dynamic aspects of the user context to improve the quality of the recommendations. The need for contextual and environmental information was highlighted by Cunningham et al. and others [5, 12, 16]. They hypothesized that listeners' location, activity, and context were probably correlated with their preferences, and thus should be considered when developing music recommendation systems. As a result, frameworks for abstracting the context of music listening by using raw features such as environmental data have been proposed in the literature [16, 22]. While some researchers have reported that context-aware recommendation systems perform better than traditional ones [15, 22, 24], others have shown only minor improvements [10]. Finally, others have carried out experiments with only the most highly-ranked music items, probably leading to models biased by popularity [15, 25].

We will now discuss the impact of using listeners' demographic and profiling characteristics— hereafter referred to as *user-side* features [19]—in improving the accuracy of a music recommendation model. User-side features were extracted from self-declared demographics data and a set of custom-built profiling features characterizing the music listening behaviour of a large amount of users of a digital music service. Their music listening histories were disaggregated into different time spans to evaluate if the accuracy of models changed using different temporal contexts of listening. Finally, models based on latent factors were learned for all listening contexts and all combinations of user-side features. Section 2 presents the dataset collection, Section 3 introduces a set of custom-built features to profile listeners' listening behaviour, and Section 4 describes the experimental set up and presents the results.

© Gabriel Vigliensoni and Ichiro Fujinaga. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Gabriel Vigliensoni and Ichiro Fujinaga. "Automatic music recommendation systems: do demographic, profiling, and contextual features improve their performance?", 17th International Society for Music Information Retrieval Conference, 2016.

2. DATASET

We are interested in evaluating the impact of using demographic and profiling features, as well as contextual information, for a large number of people, on the prediction accuracy of a music artist recommendation model. A few publicly available datasets for music listening research provide information relating people and music items. Dror et al. presented a dataset of 1M people's aggregated ratings on music items [7]. McFee et al. introduced a dataset of song playcounts of 1M listeners [17]. Neither of these two datasets, however, provided timestamps of the music logs or demographic information about the listeners. Celma provided a dataset of playcounts with listeners' demographic data for 360K listeners and a set of listening histories with full time-stamped logs; however this last dataset only included logs for 1K listeners [3]. Cantador et al. presented another small dataset with song playcounts for 2K users [2]. Finally, EMI promised a dataset of 1M interviews about people's music appreciation, behaviour, and attitudes [9], but only partial information was made available.

None of the aforementioned datasets provide, at the same time and for a large amount of listeners, access to full music listening histories as well as people's demographic data. This means it is not possible to extract all of the user-side features that we were interested in, and so we decided to collect our own dataset made with music listening histories from Last.fm. Last.fm stands out from most online digital music services because it not only records music logs of songs played back within its own ecosystem, but also from more than 600 media players.

Next, we will present the criteria and acquisition methods used to collect a large number of music listening histories from the Last.fm service.

2.1 Data criteria, acquisition, and cleaning

Aggregating people's music listening histories requires collapsing their music logs into periods of time. In order to obtain data evenly across aggregated weeks, months, seasons, or years, we searched for listeners with an arbitrary number of at least two years of activity submitting music logs since they started using the system, and also with an average of ten music logs per day. These two restrictions forced our data-gathering crawler to search for listeners with a minimum of 7,300 music logs submitted to the Last.fm database. Also, these constraints assured us that we would collect listening histories from active listeners with enough data to perform a good aggregation over time.

Data acquisition was performed by means of using several machines calling the Last.fm API during a period of two years (2012–2014). We collected listening histories by using the Last.fm's API method `user.getRecentTracks()`. This API call allowed us to obtain full listening histories. Along with this data, we also stored all available metadata for each listener, including the optional self-declared demographic features: *age*, *country*, and *gender*.

We performed several processes of data filtering and cleaning in order to avoid noisy data. For example, we realized that there were listeners with numerous duplicated music logs (i.e., same timestamp for many music item IDs), and listening histories with a great deal of music logs that were too close in time (i.e., less than 30 seconds apart, which is the minimum that Last.fm requires to consider a played track as a valid music log). Hence, we decided to filter out all duplicated logs as well as logs that were less than 30 seconds apart in time.

2.2 Dataset demographics

Our dataset consists of 27 billion music logs taken from 594K users' music listening histories. This large repository of music listening records accounts for the interaction of listeners with more than 555K different artists, 900K albums, and 7 million tracks. There are music listening histories from people in 239 self-declared different countries, with listeners from all time zones represented. However, listeners from Africa, South Asia, and East Asia are under-represented in our dataset. In fact, the 19 "top countries" combined account for more than 85 percent of the total number of listeners in the dataset. Table 1 summarizes some of the overall and demographic characteristics of users in the dataset.

Items	No.	Demographic	%	Age groups	%
Logs	27MM	Age	70.5	15–24	57.5
Tracks	7M	Country	81.8	25–34	35.8
Albums	900K	Gender	81.6	35–44	5.5
Artists	555K			45–54	1.2
Listeners	594K				

Table 1. Dataset summary (Demographic: the percentage of people who provided demographic information)

Table 1 shows that large proportion of listeners self-declared their age, gender, and country. Previous research on online profiles concluded that people usually want to be well typified by their online profiles [4], and so we assumed there is a high degree of truth in these demographic features. Listeners from all ages are not equally represented in the dataset. The age distribution is biased towards young people, with an average age of 25 years old.

3. FEATURES FOR LISTENER PROFILING

We hypothesized that by better understanding the listening behaviour of people, we will be able to more accurately model the user needs. Hence, the recommendation can be tailored to each listener and the prediction accuracy will probably improve.

A set of computational features that attempt to describe some aspects of music listening behaviour in relation to musical artists was already proposed in previous research [21]. However, the ranking of the music items was not take into consideration and feature values were binned into categories. In our approach we try to represent similar characteristics of listening behaviour but we also consider

the position of the music items within each listener's ranking as well as using normalized feature values to express the precise value of a certain listening behaviour characteristic in relation to a music item.

3.1 Feature design

We restricted ourselves to designing three novel features to describe listener behaviours: *exploratoryness*, *mainstreamness*, and *genderness*. Values for these features were computed for the three types of *music items* in the dataset: *tracks*, *albums*, and *artists*. Therefore, each listener's listening behaviour was described by a vector of nine values.

We will describe the goals behind each one of these features, give details about their implementation, visualize data patterns, and provide some analysis about the results.

3.1.1 Exploratoryness

To represent how much a listener explores different music instead of listening to the same music repeatedly we developed the *exploratoryness* feature.

For each user x's listening history, let L be the number of submitted music logs, S_k be all submitted music items of type k, where k=$\{$tracks, albums, artists$\}$, $s_{k,i}$ be the number of music logs for the given music item key k at ranking i. We computed the exploratoryness $e_{x,k}$ for listener x on a given music item of type k as:

$$e_{x,k} = 1 - \frac{1}{L}\sum_{i=1}^{S_k}\frac{s_{k,i}}{i} \quad (1)$$

Exploratoryness returns a normalized value, with values closer to 0 for users listening to the same music item again and again, and values closer to 1 for users with more exploratory listening behaviour.

3.1.2 Mainstreamness

With the goal of expressing how similar a listener's listening history is to what everyone else listened to, we developed the *mainstreamness* feature. It analyses a listener's ranking of music items, and compares it with the overall ranking of artists, albums, or tracks, looking for the position of co-occurrences.

For each user x's listening history, let N be the number of logs of the music item ranked first in the overall ranking, L be the number of submitted music logs, S_k be all submitted music items of type k, where k=$\{$tracks, albums, artists$\}$, $s_{k,i}$ be the number of music logs for the given music item key k at ranking i, and $o_{k,i}$ be the number of music logs in the overall ranking of music item type k ranked at position i. We defined the mainstreamness feature $m_{x,k}$ for listener x on a given music item of type k as:

$$m_{x,k} = \frac{1}{NL}\sum_{i=1}^{S_k} s_{k,i}o_{k,i} \quad (2)$$

Listening histories of people with a music item's ranking similar to the overall ranking receive mainstreamness values closer to 1. Listeners' mainstreamness whose ranking differ more from the overall ranking receive values closer to 0.

3.1.3 Genderness

With the aim of expressing how close a listener's listening history is to what females or males are listening to, we developed the *genderness* feature. The genderness feature computation basically relies on mainstreamness, but instead of computing just one overall ranking from all listeners, it uses two rankings: one made with music logs from listeners self-declared as female, and another one from male data.

For each user x's listening history and music item of type k, let $m_{x,k,male}$ be the mainstreamness computed with the male ranking, $m_{x,k,female}$ be the mainstreamness calculated with the female ranking.

We defined the feature genderness $g_{x,k}$ for listener x on a given music item of type k as:

$$g_{x,k} = m_{x,k,male} - m_{x,k,female} \quad (3)$$

3.2 Profiling listeners

To illustrate how the features we developed can be used to profile listeners, we calculated exploratoryness, mainstreamness, and genderness of users in our dataset. In order to not violate the homogeneity of variance we binned listeners into four age groups with balanced number of samples for each group. To obtain balanced groups, we drew a random sample of 100 people of each age, and created 10-year groups with 1000 people each. We then bootstrapped these groups with 1000 replications of the original sample and calculated 95 percent CI error bars. Although we quantified these characteristics in the relation of listeners with *artists*, *albums*, and *tracks*, and their interaction with listeners' age group, preliminary tests indicated that the interaction with artists was most significant. Therefore, for the rest of the paper we present only the results of the interaction between listeners and artists.

Figure 1 shows feature means by age group as well as 95 percent CI bars. In terms of artist exploratoryness, Figure 1(a) shows that while younger listeners in our dataset tend to listen more often to the same performers than adults, older listeners tend to explore more artists. Also, the rise in exploratoryness tends to stabilize in the mid-thirties. Figure 1(b) shows that while younger people listen more to the same artists that everyone is listening to, older people tend to listen to less common performers. This effect could be generated by the behaviour of older people or the fact that there are fewer older people in the original dataset, and so the artists they listen to are less represented in the overall ranking. Figure 1(c) shows artist genderness by age and gender. Listeners self-declared as male tend to listen more to music that is ranked higher in the male ranking, in all age groups, however their preference for the male ranking diminishes with age. Females, on the contrary, listen more to artists ranked higher in the female ranking when they are young, but adult women listen more to artists ranked higher in the male ranking. Overall, men and women have opposite trends of *genderness* in the different age groups, which seem to stabilize as they mature.

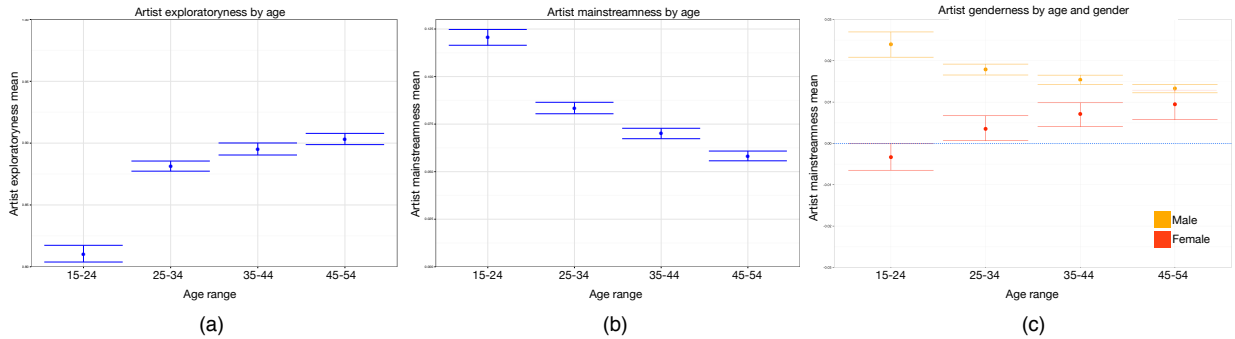


Figure 1. Feature means and 95% CI bars for a random group of listeners in our dataset. Each age group has 1K listeners. Error bars were calculated by taking 1K populations replicated from the original sample using bootstrap. (a) Artist exploratoryness by age group of listeners, (b) artist mainstreamness by age group of listeners, and (c) artist genderness by listeners' age group and gender.

We hoped that the aforementioned features captured some information about people's listening behaviour and will help to improve the accuracy of a music recommender model. However, as genderness was derived directly from mainstreamness, we did not use it in the experimental procedure for evaluating a music recommendation model with user-side data.

4. EXPERIMENTAL PROCEDURE

Our goal is to evaluate if demographics, behavioural profiles, and the use of observations from different contexts improve the accuracy of a recommendation model. Our sources of data involve a matrix of user preferences on artists derived from implicit feedback, a set of three categorical demographic features for each user: *age*, *country*, and *gender*, and a set of two continuous-valued features for describing people's listening behaviour: *exploratoryness* and *mainstreamness*. Preference matrixes were generated by considering full week of music listening histories data, as well as data coming from music logs submitted on weekdays and weekends only.

We followed a similar approach to Koren et al., in which a matrix of implicit feedback values expressing preferences of users on items is modelled by finding two lower dimensional matrixes of rank f $X_{n\times f}$ and $Y_{m\times f}$, which product approximates the original preference [14]. The goal of this approach is to find the set of values in X and Y that minimize the RMSE error between the original and the reconstructed matrixes. However, this conventional approach of matrix factorization for evaluating the accuracy of recommendation models using latent factors does not allow the researcher to incorporate additional features, such as *user-side* features. In order to incorporate latent factors as well as user-side features into one single recommendation model, we used the *Factorization Machines* method for matrix factorization and singular value decomposition [18]. In this approach, interactions between all latent factors as well as additional features are computed within a single framework, with a computational complexity that is linear to the number of extra features.

In order to perform a series of experiments with different sets of model parameters and user-side features in a timely fashion, we randomly sampled 10 percent of per-user music listening histories in the dataset, and we split this new subset into two disjoint sets: training (90 percent) and testing (10 percent) datasets. The training dataset had more than 60M observations from 59K users on 432K artists, with a density of observations of about 0.24 percent. We aggregated each dataset of listening histories by creating `<user, artist, playcounts>` triples. Then, we transformed the number of playcounts in each triple into a 1–5 Likert scale value by means of calculating the complementary cumulative distribution of artists per listener [3]. Hence, artists in each distribution quintile were assigned with a preference value according to how much each user listened to them.

In order to learn the best set of parameters of the recommendation model, we performed a grid search on the λ regularization parameter as well as the f number of latent factors with no user-side data, just using plain matrix factorization for the matrix of preferences of users on artists. Finding a good λ value helps to avoid overfitting the observed data by penalizing the magnitudes of the learned parameters. Finding the best f number of factors helps to obtain a better recommendation accuracy while also providing a set of to-be-interpreted latent factors. We used the Graphlab Create framework [1] to search over the number of latent factors in the range $[50, 200]$, and regularization values in the range $[1\times10^{-5}, 1\times10^{-8}]$. The best combination of parameters was achieved for λ=1×10^{-7} and f=50 latent factors. We used the Adaptive Stochastic Gradient Descent optimization algorithm [8] and set the maximum number of iterations at 50.

4.1 Demographic and profiling features

With these model parameter values, we evaluated the recommendation accuracy in the testing dataset of models

[1] `https ://pypi.python.org/pypi/GraphLab-Create`

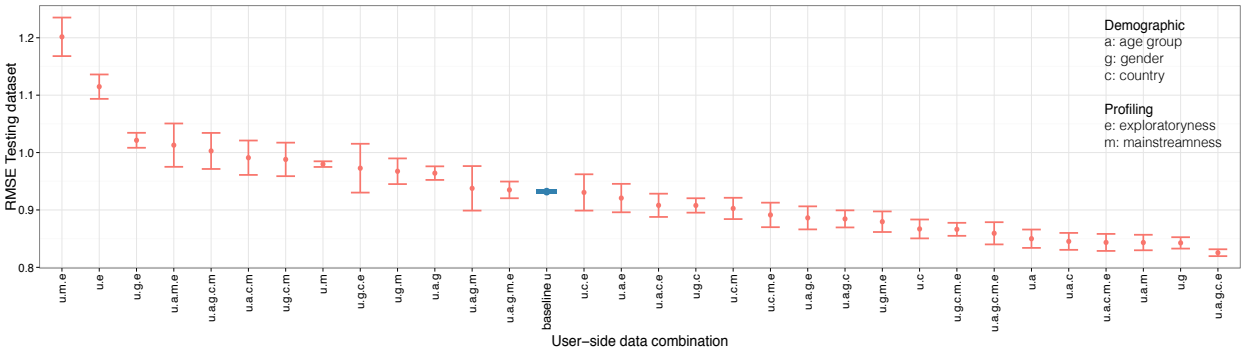


Figure 2. Root mean square error means and 95 percent CI bars for learned models evaluated in the testing dataset, with 32 combinations of the user-side features: *age*, *gender*, *country*, *exploratoryness*, and *mainstreamness* , ranked in decreasing order. Feature combinations are labelled according to the first letter of the word they represent. Baseline for comparison is combination *u*: user's preferences only, without any user-side features.

learned from the training data for all combinations of user-side demographic and profiling features. Since we had five user-side features: age, gender, country, exploratoryness, and mainstreamness, there were 32 different combinations.

Learning a model using an optimization algorithm can sometimes cause the results to converge into local minima instead of the global minimum. We informally evidenced that the variance in results of the optimization algorithm was larger than the variance in using different samples of the dataset. Hence, we repeated the process of learning and testing the accuracy of the learned models 10 times for each user-side feature combination. Using this procedure, we also wanted to compare and evaluate if results in model error were similar throughout several trials. The experiment baseline was established as the approach in which plain matrix factorization was used to estimate the recommendation accuracy of the learned models by just using the matrix of preferences of listeners on artists, without any user-side feature combination. By using this approach, we will be able to compare if the use of any feature combination resulted in a decrease in RMSE error, thus indicating an increase in the accuracy of the model.

Figure 2 summarizes the results of all trials. It shows all feature combination means, ranked in decreasing order, with 95 percent CI error bars generated from a bootstrap sample of 100 replications of the original sample. Feature combinations are labelled according to the first letter of the word they represented. For example, user preference data with age, gender, and exploratoryness is labelled *u.a.g.e*; user data with no user-side feature combinations is just labelled *u*. It can be seen that *u*, the baseline without user-side features, achieved an average RMSE value of .931 and exhibited a small variability, indicating that models in this setup were stable across all trials. All feature combinations to the right of the *u* show a smaller RMSE error, thus providing evidence for an increase in the learned accuracies of those models. Several feature combinations achieved better accuracy than the baseline. In particular, those combinations using just one of the demographic features: country (*u.c*), age (*u.a*), or gender (*u.g*) achieved improvements of about 7, 8, and 9 percent, respectively. Also, the combination of only demographic features (*u.a.g.c*), and all demographic and profiling features (*u.a.g.c.e.m*) improved the baseline model by almost 8 percent. However, the combination of features that achieved the best result was all demographic features together, plus the listener profiling feature of exploratoryness (*u.a.g.c.e*), exhibiting an improvement of about 12 percent above the baseline. The small variability in the model error of this combination throughout all trials suggested that models based on this user-side feature combination were quite stable. On the other hand, the combination of profiling features (*u.m.e)* achieved the worst performance, with a 29 percent increase in error, and a large variability in the estimated model error throughout trials. The variability in the results with these features suggests that the data topology using only profiling features is complex, probably making the iterative process of optimization converge into non-optimal local minima in the data.

4.2 Listening preferences in the contexts of entire week, weekdays only, and weekends only

We hypothesized that if people listen to different music during the weekdays than on weekends, we could create more accurate models by using data from only the weekdays or weekends, respectively. To test this hypothesis, we performed the same experimental approach that we carried out with the full-week dataset. However, this time we created two additional preference matrices of listeners for artists. The first additional matrix was made by using only music logs submitted during weekdays, and the second matrix was made by using only weekend music logs. Therefore, two extra sub-datasets were created using the original full-week dataset: *weekday* and *weekend* datasets. We then followed the same procedure described before: we split the data into training and testing datasets, we learned models from the training dataset for all 32 possible combinations of user-side features, and evaluated the accuracy of those models in the testing dataset. The number of observations, listeners, and artists, and also each of the matrix densities are shown in Table 2.

Dataset	Observations	Listeners	Artists	Density
Full-week	61M	59K	432K	0.237%
Weekdays	54M	59K	419K	0.216%
Weekends	35M	59K	379K	0.154%

Table 2. Number of observations, listeners, artists, and density for each context-based preference matrix.

As expected, the number of observations decreased in the datasets with partial data in relation to the full-week dataset. The number of listeners remained constant, which implies that most listeners in the dataset submitted music logs during weekdays as well as on weekends. Interestingly, the total number of artists for which there were submitted music logs on weekdays and weekends decreased between 3 and 12 percent in relation to the full week data, which implies that many artists in the dataset were only listened during one of the two weekly periods.

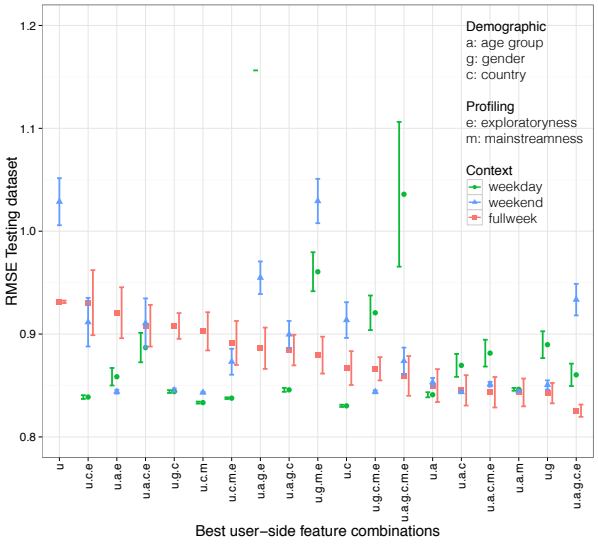


Figure 3. Root mean square error means and 95 percent CI bars for learned models with weekday, weekend, and full-week data. Only those feature combinations with a better RMSE value than the baseline for full-week data are shown.

Figure 3 summarizes model accuracies obtained using music log data from the three aforementioned contexts. Many of the models made with weekly-split listening data achieved better performance than those using full-week data. For example, models learned with weekday as well as weekend data using feature combinations *u.a.e*, *u.g.c*, and *u.c.m* achieved improvements in accuracy of about 7 percent in comparison to the model created with full-week data. They also showed smaller variability meaning more stability in model estimation. However, while the best RMSE value was obtained using the user-side feature combination *u.a.g.c.e* with full-week data, the same feature combination achieved worse performances by using listening data from weekdays and weekends only.

5. CONCLUSIONS AND FUTURE WORK

We have evaluated the impact of listeners' demographic and profiling features as well as basic forms of listening context, namely weekday and weekend versus full-week listening, on recommendation accuracy. We described our requirements for a dataset of music listening histories, explaining why none of the available datasets met our needs and how we ended up collecting our own data. We then formalized a set of profiling features that account some aspects of music listening behaviour. We also explained how we split our dataset of listening histories into weekdays and weekend listening histories to evaluate if having data from different sets of listening histories improved the accuracy of recommendation. Finally, we described how we set experiments that evaluated all combinations of user-side data features in the different contexts of listening. We found that the feature combination that achieved the smallest error was all demographic features together plus listener's exploratoryness, obtaining 12 percent improvement over the baseline of not using any user-side feature data. Although for some feature combinations the use of split listening data improved the recommendation, the best combination of features did benefit from having full-week data.

The results, in particular the many low RMSE values for several feature combinations using split listening data, seem to indicate that these error values are close to the limit in the minimum achievable error. This characteristic has already been described in the literature as a "magic barrier" in recommender systems design [1], referring to the upper bound in rating prediction accuracy due to inconsistencies in user's ratings. However, since we are mapping the number of submitted music listening logs into ratings, we do not see how these inconsistencies can explain this barrier. It would be interesting to perform a narrower grid search in order to investigate if we are actually hitting a wall in accuracy, or if there is a better set of model parameters that allows us to create more stable models and better performances throughout many trials. In comparison with previous research [21], the results are not comparable since different metrics are used. Also, our experiment directly integrated the profiling features into the matrix factorization algorithm. Finally, although these results show an improvement in the accuracy of a recommendation model based on listeners' past listening histories, we might require an online, user-centred study to measure people's actual satisfaction with the learned model.

6. ACKNOWLEDGEMENTS

This research has been supported by BecasChile Bicentenario, Comisión Nacional de Ciencia y Tecnología, Gobierno de Chile, and the Social Sciences and Humanities Research Council of Canada. Important parts of this work used ComputeCanada's High Performance Computing resources. The authors would like to thank Emily Hopkins for her thorough proofreading of this paper.

7. REFERENCES

[1] A. Bellogín, A. Said, and A. P. de Vries. The magic barrier of recommender systems—no magic, just ratings. In *User Modeling, Adaptation, and Personalization*, pages 25–36. Springer, 2014.

[2] I. Cantador, P. Brusilovsky, and T. Kuflik. Last.fm web 2.0 dataset. In *2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011)*, RecSys 2011, Chicago, IL, 2011.

[3] Ò. Celma. *Music Recommendation and Discovery: the Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.

[4] S. Counts and K. B. Stecher. Self-presentation of personality during online profile creation. In *Proceedings of the Third International ICWSM Conference*, pages 191–4, 2009.

[5] S. Cunningham, S. Caulder, and V. Grout. Saturday night or fever? Context aware music playlists. In *Proceedings of the Audio Mostly Conference*, pages 1–8, Piteå, Sweden, 2008.

[6] T. DeNora. *Music in Everyday Life*. Cambridge University Press, 2000.

[7] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The Yahoo! music dataset and KDD-cup'11. *Journal of Machine Learning Research*, 18:3–18, 2011.

[8] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–59, 2011.

[9] EMI. EMI Million Interview Dataset, 2012. `http://musicdatascience.com/emi-million-interview-dataset/`. Accessed 18 February 2016.

[10] M. Gillhofer and M. Schedl. Iron Maiden while jogging, Debussy for dinner? an analysis of music listening behavior in context. In X. He, S. Luo, D. Tao, C. Xu, J. Yang, and M. Abul Hasan, editors, *MultiMedia modeling*, volume 8936 of *Lecture Notes in Computer Science*, pages 380–91, Switzerland, 2015. Springer.

[11] D. J. Hargreaves, R. MacDonald, and D. Miell. How do people communicate using music. In D. Miell, R. MacDonald, and D. J. Hargreaves, editors, *Musical Communication*, chapter 1, pages 1–25. Oxford University Press, 2005.

[12] P. Herrera, Z. Resa, and M. Sordo. Rocking around the clock eight days a week: an exploration of temporal patterns of music listening. In *1st Workshop on Music Recommendation and Discovery*, Barcelona, Spain, 2010.

[13] V. J. Konečni. Social interaction and musical preference. In D. Deutsch, editor, *The Psychology of Music*, pages 497–516. Academic Press, New York, NY, 1982.

[14] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–7, 2009.

[15] D. Lee, S. E. Park, M. Kahng, S. Lee, and S. Lee. Exploiting contextual information from event logs for personalized recommendation. In R. Lee, editor, *Computer and Information Science*, pages 121–39. Springer-Verlag, 2010.

[16] J. S. Lee and J. C. Lee. Context awareness by case-based reasoning in a music recommendation system. In *Ubiquitous Computing Systems*, pages 45–58. Springer, 2007.

[17] B. McFee, T. Bertin-Mahieux, D. P. W. Ellis, and G. R. G. Lanckriet. The million song dataset challenge. In *Proceedings of the 21st International Conference on World Wide Web*, pages 909–16, Lyon, France, 2012.

[18] S. Rendle. Factorization machines. In *IEEE 10th International Conference on Data Mining*, pages 995–1000, Sydney, Australia, 2010. IEEE.

[19] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 635–44, Beijing, China, 2011.

[20] M. Schedl, A. Flexer, and J. Urbano. The neglected user in music information retrieval research. *Journal of Intelligent Information Systems*, 41(3):523–39, 2013.

[21] M. Schedl and D. Hauger. Tailoring music recommendations to users by considering diversity, mainstreaminess, and novelty. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 947–50, 2015.

[22] D. Shin, J. Lee, J. Yeon, and S. Lee. Context-aware recommendation by aggregating user context. In *2009 IEEE Conference on Commerce and Enterprise Computing*, pages 423–30, 2009.

[23] J. A. Sloboda, A. Lamont, and A. Greasley. Choosing to hear music: motivation, process and effect. In S. Hallam, I. Cross, and M. Thaut, editors, *The Oxford Handbook of Music Psychology*, chapter 40, pages 431–40. Oxford University Press, Oxford, UK, 2009.

[24] J. Su, H. Yeh, P. Yu, and V. Tseng. Music recommendation using content and context information mining. *IEEE Intelligent Systems*, 25(1):16–26, 2010.

[25] X. Wang. *Interactive Music Recommendation: Context, Content and Collaborative Filtering*. PhD thesis, National University of Singapore, 2014.

AUTOMATIC OUTLIER DETECTION IN MUSIC GENRE DATASETS

Yen-Cheng Lu[1] **Chih-Wei Wu**[2] **Chang-Tien Lu**[1] **Alexander Lerch**[2]

[1] Department of Computer Science, Virginia Tech, USA
[2] Center for Music Technology, Georgia Institute of Technology, USA
kevinlu@vt.edu, cwu307@gatech.edu, ctlu@vt.edu, alexander.lerch@gatech.edu

ABSTRACT

Outlier detection, also known as anomaly detection, is an important topic that has been studied for decades. An outlier detection system is able to identify anomalies in a dataset and thus improve data integrity by removing the detected outliers. It has been successfully applied to different types of data in various fields such as cyber-security, finance, and transportation. In the field of Music Information Retrieval (MIR), however, the number of related studies is small. In this paper, we introduce different state-of-the-art outlier detection techniques and evaluate their viability in the context of music datasets. More specifically, we present a comparative study of 6 outlier detection algorithms applied to a Music Genre Recognition (MGR) dataset. It is determined how well algorithms can identify mislabeled or corrupted files, and how much the quality of the dataset can be improved. Results indicate that state-of-the-art anomaly detection systems have problems identifying anomalies in MGR datasets reliably.

1. INTRODUCTION

With the advance of computer-centric technologies in the last few decades, various types of digital data are being generated at an unprecedented rate. To account for this drastic growth in digital data, exploiting its (hidden) information with both efficiency and accuracy became an active research field generally known as Data Mining.

Outlier detection, being one of the most frequently studied topics in Data Mining, is a task that aims to identify abnormal data points in the investigated dataset. Generally speaking, an outlier often refers to the instance that does not conform to the expected behavior and should be highlighted. For example, in a security surveillance system an outlier could be the intruder, whereas in credit card records, an outlier could be a fraud transaction.

Many algorithms have been proposed to identify outliers in different types of data, and they have been proven successful in fields such as cyber-security [1], finance [4], and transportation [17]. Outlier detection techniques can also be used as a pre-processing step to remove anomalous data. In the work of Smith and Martinez [24], a set of outlier detection methods are used to remove anomalies from the dataset, followed by several widely-used classification methods in order to compare the performance before and after outlier removal. The result indicates that removing outliers could lead to statistically significant improvements in the training quality as well as the classification accuracy for most of the cases.

Music datasets offer similar challenges to researchers in the field of MIR. Schedl et al. point out that many MIR studies require different datasets and annotations depending on the task [22]. However, since the annotation of music data is complex and subjective, the quality of the annotations created by human experts varies from dataset to dataset. This inaccuracy may potentially introduce errors to the system and decrease the resulting performance.

One MIR task known for this issue is Music Genre Recognition (MGR). According to Sturm [26], the most frequently used dataset in MGR is GTZAN [29], and many of the existing systems are evaluated based on their performance on this dataset. Sturm points out that this dataset contains corrupted files, repeated clips, and misclassified genre labels. These are undesirable for the proper training and testing of a MGR system.

To address the problem of identifying such anomalies in music datasets, an investigation into existing outlier detection algorithms is a good starting point. The goal of this paper is to assess the viability of state-of-the-art outlier detection methods in the context of music data. The contribution of this paper can be summarized as follows: first, this early stage investigation provides a systematic assessment of different outlier detection algorithms applied to a music dataset. Second, the use of standard audio features reveals the capability as well as the limitations of this feature representation for outlier detection. Third, we provide insights and future directions for related studies.

This paper is structured as follows. In Sect. 2, the related work of outlier detection in music data is summarized. The methods used in this paper, such as feature extraction and different outlier detection algorithms, are described in Sect. 3. Section 4 presents the results and discusses the experiments. Finally, the conclusion and future work are given in Sect. 5.

© Yen-Cheng Lu, Chih-Wei Wu, Chang-Tien Lu, Alexander Lerch. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Yen-Cheng Lu, Chih-Wei Wu, Chang-Tien Lu, Alexander Lerch. "Automatic Outlier Detection in Music Genre Datasets", 17th International Society for Music Information Retrieval Conference, 2016.

2. RELATED WORK

Outlier detection methods are typically categorized into five groups: (a) *distance-based* [14, 19], (b) *density-based* [3, 16], (c) *cluster-based* [30], (d) *classification-based* [8, 21], and (e) *statistics-based* [5, 6, 13, 18, 20, 28] methods.

The first group (*distance-based*), proposed by Knorr et al. [14], computes the distances between samples and detects outliers by setting a distance threshold. Methods in this category are usually straightforward and efficient, but the accuracy is compromised when the data is sparse or unevenly distributed. The basic idea was extended by combining the distance criterion with the k-nearest neighbor (KNN) based method [19], which adapts the distance threshold by the k-nearest neighboring distances.

The second group (*density-based*) estimates the local densities around the points of interest in order to determine the outliers. Different variations use different methods to determine the local density, for example, the local outlier factor (LOF) [3] and the local correlation integral (LOCI) [16]. These approaches are popular and have been widely used in different fields.

The third group (*clustering-based*), as proposed in [30], first applies a clustering algorithm to the data, and then labels the wrongly clustered instances as outliers.

The fourth group (*classification-based*) assumes that the designation of anomalies can be learned by a classification algorithm. Here, classification models are applied to classify the instances into inliers and outliers. This is exemplified by Das et al. [8] with a one-class Support Vector Machine (SVM) based approach and Roth [21] with Kernel Fisher Discriminants.

The fifth group (*statistics-based*) assumes the data has a specific underlying distribution, and the outliers can be identified by finding the instances with low probability densities. A number of works apply the similar concept with variations, including techniques based on the robust Mahalanobis distance [20], direction density ratio estimation [13], and minimum covariance determinant estimator [6]. One of the main challenges of these approaches is the reduction of masking and swamping effects: outliers can bias the estimation of distribution parameters, yielding biased probability densities. This effect could result in a biased detector identifying normal instances as outliers, and outliers as normal, respectively. Recent advances have generally focused on applying robust statistics to outlier detection [5, 18, 28]. This is usually achieved by adopting a robust inference technique to keep the model unbiased from outliers in order to capture the normal pattern correctly.

Although the above mentioned approaches have been applied to different types of data, the number of studies on music datasets is relatively small. Flexer et al. [10] proposed a novelty detection approach to automatically identify new or unknown instances that are not covered by the training data. The method was tested on a MGR dataset with 22 genres and was shown to be effective in a cross-validation setting. However, in real-world scenarios, the outliers are usually hidden in the dataset, and an outlier-free training dataset may not be available. As a result, the proposed method might not be directly applicable to other music datasets. Hansen et al. [12] proposed the automatic detection of anomalies with a supervised method based on parzen-window and kernel density estimation. The proposed algorithm was evaluated on a 4-class MGR dataset, which consisted of audio data recorded from radio stations. A commonly used set of audio features, the Mel Frequency Cepstral Coefficients (MFCCs), was extracted to represent the music signals. This approach, nevertheless, has two underlying problems. First, the dataset used for evaluation does not have a ground truth agreed on by human experts. Second, while MFCCs are known to be useful in a multitude of MIR tasks, they might not be sufficient to represent music signals for outlier detection tasks.

To address these issues, two approaches have been taken in this paper: First, for evaluation, a commonly-used MGR dataset with reliable ground truth is used. In Sturm's analysis [26], a set of outliers (i.e., repeated, distorted, and mislabeled music clips) were identified manually in the popular GTZAN [29] dataset. This analysis provides a solid ground for the evaluation of an anomaly detection system in the MGR dataset. Second, we extend the set of descriptors for the music data. In addition to the MFCCs, audio features that are commonly used in MIR tasks are also extracted in order to evaluate the compatibility of current audio features with the existing outlier detection methods.

3. METHOD

3.1 Feature Extraction

Feature extraction is an important stage that transforms an audio signal into a vector-based representation for further data analysis. In an early study of automatic music genre classification, Tzanetakis and Cook proposed three feature sets that characterized any given music signal based on its timbral texture, rhythmic content and pitch content [29]. These features have shown their usefulness in music genre classification, and have been used in many music-related tasks. Although many studies presented more sophisticated features (e.g., [11]) with higher classification accuracy on the GTZAN dataset, the original set of features still seem to provide a good starting point for representing music data. Therefore, a set of baseline features based on Tzanetakis and Cook's features [29] is extracted to allow for easier comparison with prior work. The extracted features can be divided into three categories: spectral, temporal and rhythmic. All of the features are extracted using a block-wise analysis method. To begin with, the audio signal is down-mixed to a mono signal. Next, a Short Time Fourier Transform (STFT) is performed using a block size of 23 ms and a hop size of 11 ms with a Hann window in order to obtain the time-frequency representation. Finally, different instantaneous features are extracted from every block. Spectral features are computed using the spectrum of each block. Temporal features are computed from the time domain signal of each block directly. The rhythmic features are extracted from the beat histogram of the entire time domain signal. The extracted features are (for the details of the implementations, see [15]):

1. **Spectral Features** ($d = 16$): Spectral Centroid (SC), Spectral Roll-off (SR), Spectral Flux (SF), 13 Mel Frequency Cepstral Coefficients (MFCCs)

2. **Temporal Features** ($d = 1$): Zero Crossing Rate (ZCR)

3. **Rhythmic Features** ($d = 8$): Period0 (P0), Amplitude0 (A0), RatioPeriod1 (RP1), Amplitude1 (A1), RatioPeriod2 (RP2), Amplitude2 (A2), RatioPeriod3 (RP3), Amplitude3 (A3).

All of the features are aggregated into texture vectors following the standard procedure as mentioned in [29]; the length of the current texture block is 0.743 s. The mean and standard deviation of the feature vectors within this time span will be computed to create a new feature vector. Finally, all the texture blocks will be summarized again by the mean and standard deviation of these blocks, generating one feature vector to represent each individual recording in the dataset.

3.2 Outlier Detection Methods

3.2.1 Problem Definition

Given N music clips that have been converted into a set of feature vectors $X = \{X_1, ..., X_N\}$ with the corresponding genre label $Y = \{Y_1, ..., Y_N\}$, where each Y_n is belong to one of the M genres (i.e., $Y_n \in \{C_1, ..., C_M\}$), the objective is to find the indices of the abnormal instances which have an incorrect label Y_n.

For this study, we choose 6 well-known outlier detection methods from different categories as introduced in Sect. 2 and compare their performances on a MGR dataset. The methods are described in details in the following sections:

3.2.2 Method 1: Clustering

Clustering is a *cluster-based* approach as described in Sect. 2. In our implementation of this method, we apply k-means to cluster the data into 10 groups. Based on the assumption that normal data are near the cluster centroids while the outliers are not [7, 25], the anomalous score of a given instance is defined by the distance between the point and the centroid of the majority within the same class.

3.2.3 Method 2: KNN

KNN method is a *distance-based* approach that typically defines the anomalous score of each instance by its distance to the k nearest neighbors [9]. It can be expressed in the following equation:

$$k\text{–}distance(P) = d(P, knn(P)) \tag{1}$$

where knn is the function that returns the k-th nearest neighbor of a point P, and d is the function that calculates the distance between two points. Finally, we may compute the outlier score as:

$$\frac{1}{k} \sum_{p \in neighbors_k(P)} k\text{–}distance(p) \tag{2}$$

Setting k to a larger number usually results in a model more robust against outliers. When k is small, the anomalous score given by this method may be biased by a small group of outliers. In our implementation of this method, we apply $k = 6$ in order to maintain a balance between robustness and efficiency.

3.2.4 Method 3: Local Outlier Factor

The Local Outlier Factor (LOF) [3] is a *density-based* approach that extends the KNN method with a calculation of the local densities of the instances. It is one of the most popular anomaly detection methods. It starts with the definition of k-reachability distance:

$$k\text{–}reachDist(P, O) = max(k\text{–}distance(P), d(O, P)) \tag{3}$$

This represents the distance from O to P, but not less than the $k\text{–}distance$ of P. The local reachability density of a given sample is defined by the inverse of the average local rechability distances of k-nearest neighbors:

$$lrd(P) = 1/\left(\frac{\sum_{P_0 \in neighbors_k(P)} k\text{–}reachDist(P, P_0)}{|neighbors_k(P)|}\right) \tag{4}$$

Finally, the lof calculates the average ratio of the local reachability densities of the k-nearest neighbors against the point P:

$$lof(P) = \frac{\sum_{P_0 \in neighbors_k(P)} lrd(P_0)}{lrd(P)|neighbors_k(P)|} \tag{5}$$

In a dataset that is densely distributed, a point may have shorter average distance to its neighbors, and vice versa. Since LOF uses the ratio instead of the distance as the outlier score, it is able to detect outliers in clusters with different densities.

3.2.5 Method 4: One-Class SVM

The One-Class SVM [23] is a *classification-based* approach that identifies outliers with a binary classifier. Given a genre $m \in \{1, ..., M\}$, every sample in m can be classified as in-class or off-class, and the off-class instances are most likely to be the outliers. A One-Class SVM solves the following quadratic programming problem:

$$\begin{aligned} \min_{w, \xi_i, \rho} \quad & \tfrac{1}{2}||w||^2 + \tfrac{1}{\nu N} \textstyle\sum_i \xi_i - \rho \\ \text{subject to} \quad & (w\Phi(x_i)) \geq \rho - \xi_i, i = 1...N \\ & \xi_i \geq 0, i = 1...N \end{aligned} \tag{6}$$

where ξ, w, and ρ are the parameters to construct the separation hyperplane, ν is a parameter that serves as the upper bound fraction of outliers and a lower bound fraction of samples used as support vectors, and Φ is a function that maps the data into an inner product space such that the projected data can be modeled by some kernels such as a Gaussian Radial Basis Function (RBF). By optimizing the above objective function, a hyperplane is then created to separate the in-class instances from the outliers. In the experiment, we construct a One-Class SVM for each of the genres, and identify the music clips that are classified as off-class instances as outliers.

3.2.6 Method 5: Robust PCA

Robust PCA [5] is a *statistics-based* approach that considers the problem of decomposing a matrix X into the superposition of a low-rank matrix L_0 and a sparse matrix S_0, such that:

$$X = L_0 + S_0$$

The problem can be solved by the convex optimization of the Principal Component Pursuit [5]:

$$minimize||L||_* + \lambda||S||_1 \quad (7)$$

$$\text{subject to } L + S = X \quad (8)$$

where $||L||_*$ is the nuclear norm of L, and λ is the sparsity constraint that determines how sparse S would be.

The matrix S_0 is a sparse matrix consisting of mostly zero entries with a few non-zero entries being the outliers. In the experiment, we apply this method to the music data and calculate the sparse matrices for every genre. Next, we normalize the features using a standard Z-score normalization process, and identify the outliers by finding the instances with a maximum sparse matrix value that is 3 times greater than the unity standard deviation.

3.2.7 Method 6: Robust Categorical Regression

Robust Categorial Regression (RCR) is another *statistics-based* method that identifies outliers based on a regression model. First, we formulate the relation of Y and X based on a linear input-output assumption:

$$g(Y) = X\beta + \varepsilon, \quad (9)$$

where g is the categorical link function, β is the regression coefficient matrix, and ε is a random variable that represents the white-noise vector of each instance. The link function g is a logit function paired with a category C_M, i.e., $\ln\left(P(Y_n = C_m)/P(Y_n = C_M)\right) = X_n\beta_m + \varepsilon_{nm}$. Since the probabilities of the categories will sum up to one, we can derive the following modeling equation:

$$P(Y_n = C_m) = \frac{\exp\left\{X_n\beta_m + \varepsilon_{nm}\right\}}{1 + \sum_{l=1}^{M-1}\exp\left\{X_n\beta_l + \varepsilon_{nl}\right\}} \quad (10)$$

and

$$P(Y_n = C_M) = \frac{1}{1 + \sum_{l=1}^{M-1}\exp\left\{X_n\beta_l + \varepsilon_{nl}\right\}} \quad (11)$$

The coefficient vector β usually represents the decision boundary in a classification problem. In this approach, β is used to capture the normal behavior of the data.

The robust version of categorical regression applies a heavy-tailed distribution, which is a zero-mean Student-t distribution to capture the error effect caused by outliers.

The solution to this regression model is approximated with a variational Expectation-Maximization (EM) algorithm [2]. Once converged, the errors of the instances are expected to be absorbed in the ε variables. Finally, the outliers can be identified by finding the instances with ε that is 3 times greater than the unity standard deviation.

4. EXPERIMENT

4.1 Experiment Setup

To evaluate the state-of-the-art outlier detection methods as described in Sect. 3.2, different experiments are conducted on the well-known GTZAN dataset [29]. This dataset consists of 10 music genres (i.e., blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock), with each genre containing 100 audio tracks; each track is a 30-second long excerpt from a complete mixture of music. For each method, two sets of experiments are conducted.

In the first set of experiments, we use a purified GTZAN dataset, which excludes the conspicuous misclassified and jitter music clips reported in [26]. This setup simulates the best case scenario, where the dataset is clean and all genres are well separated in the feature space. The results can serve as a sanity check of all the methods. Two types of injection experiments are conducted on this purified dataset, namely label injection and noise injection. The label injection process is performed by randomly choosing 5% of instances, and swapping their genre labels to create outliers. In this experiment, two sets of features are used to represent the music data, one is the full feature set as described in Sect. 3.1, and the other is the baseline feature set using only 13 MFCCs as reported in the work of Hansen et al. [12] for comparison. The noise injection process is performed by randomly choosing 5% of instances in the data, and shifting 20% of their feature values by 5 times the standard deviation. This experiment uses the full feature set to test the methods' capability of detecting corrupted data. For each of the experiments above, we generate 10 random realizations and report the averaged evaluation results.

In the second set of experiments, we apply all the methods to the full GTZAN dataset directly, and the identified outliers are compared with the list of conspicuous genre labels and the obviously corrupted clips (*Hip-hop (38), Pop (37), Reggae (86)*) reported in [26]. This experiment provides the real-world scenario, in which case the outlier detection should find the outliers identified by human experts.

All of the experiments use the same metrics for the performance measurements, which include the standard calculation of Precision, Recall, F-measure, and the Area Under ROC Curves (AUC).

4.2 Experiment Results

The results of the first set of experiments, evaluating the performance of the methods on detecting injected misclassification labels with full features, are shown in Table 1. With F-measures in the range from 0.1–0.57, the results do not have high reliability but are usable for some methods. The *Robust Categorical Regression* approach outperforms the other algorithms. Since RCR explicitly models the input-output relationship between the features and the labels, it fits the data better compared to the other methods. Surprisingly, the simple methods such as *Clustering* and *KNN* also perform relatively well in terms of AUC, and they outperform the more sophisticated approaches such as *LOF*

Method	Precision	Recall	F-measure	AUC
CLUS	0.23	0.23	0.23	0.74
KNN	0.26	0.26	0.26	0.77
LOF	0.11	0.11	0.11	0.57
SVM	0.06	0.32	0.10	0.52
RPCA	0.47	0.34	0.39	0.78
RCR	0.59	0.55	0.57	0.91

Table 1. Average Detection Rate comparison of label injection with full features

Method	Precision	Recall	F-measure	AUC
CLUS	0.06	0.06	0.06	0.61
KNN	0.10	0.10	0.10	0.71
LOF	0.09	0.09	0.09	0.62
SVM	0.05	0.38	0.09	0.50
RPCA	0.30	0.20	0.24	0.65
RCR	0.52	0.40	0.45	0.87

Table 2. Average Detection Rate comparison of label injection with MFCCs only

and *One-Class SVM*. One possible explanation is that the label injection datasets contain outliers generated by swapping dissimilar genres, e.g., swapping the label from Jazz to Metal. As a result, the decision boundaries of *LOF* and *One-Class SVM* might be biased towards the extreme values and perform poorly. On the other hand, the simple methods such as *Clustering* and *KNN*, which are based on Euclidean distance, were able to identify these instances without being biased. Generally speaking, the *statistics-based* approaches, such as the robust statistics based methods *RPCA* and *Robust Categorical Regression*, perform better on the label injection datasets.

The results of the same experiments with only MFCCs as features are shown in Table 2. In general, the performance drops drastically. This result implies that MFCCs might not be representative enough for the outlier detection task.

Table 3 shows the results for the noise injection experiment. It can be observed that density and distance methods, such as *CLUS*, *KNN*, and *LOF*, have better results on detecting corrected data. The main distinction of this kind of outlier is that the abnormal behavior is explicitly shown in the feature space instead of implicitly embedded in the relationship between genre labels and the features. Therefore, the methods that directly detect outliers in the feature space tend to outperform the other methods such as *SVM*, *RCR* and *RPCA*.

In the second set of experiments, we perform the anomaly detection on the complete GTZAN dataset with full features, and aim to detect the misclassified music clips reported by Sturm [26]. The experiment result is shown in Table 4. Based on these metrics, none of these methods are able to detect the anomalies with high accuracy. Compared

Method	Precision	Recall	F-measure	AUC
CLUS	0.92	0.90	0.91	0.99
KNN	0.99	0.98	0.99	1.00
LOF	1.00	0.98	0.99	1.00
SVM	0.05	0.41	0.09	0.50
RPCA	0.32	0.23	0.27	0.72
RCR	0.61	0.50	0.55	0.75

Table 3. Average Detection Rate comparison of noise injection with full features

Method	Precision	Recall	F-measure	AUC
CLUS	0.15	0.13	0.14	0.54
KNN	0.18	0.15	0.16	0.56
LOF	0.18	0.15	0.16	0.59
SVM	0.09	0.63	0.15	0.66
RPCA	0.08	0.09	0.08	0.51
RCR	0.17	0.22	0.19	0.60

Table 4. Detection Rate comparison on GTZAN detecting Sturm's anomalies with full features

to the other methods, *SVM* and *RCR* present AUCs that are relatively higher, however, the Precision, Recall and F-measures are still too low to be applicable in real-world scenarios. The *One-Class SVM* method performs better in this experiment than it does in the previous experiment. We speculated that in the case of injection, the model is biased by the extreme values introduced by the injected outliers. In the real-world scenario, however, the differences between the outliers and the non-outliers are relatively subtle. When *One-Class SVM* expands its in-class region moderately, it learns a better decision boundary. Therefore, it has a better capability of detecting the outliers.

It can be observed that both *statistics-based* approaches, *RPCA* and *RCR*, do not perform well compared to the results of the previous experiment. Since these methods are good at capturing extreme values and prevent the model from being biased by the outliers, they are relatively weak in differentiating subtle differences in the feature space. Therefore, the resulting performances are not ideal.

4.3 Discussion

To further reveal the relationship between different methods and outliers from different genres, we list the distribution of top 20 true and false outliers ranked by the anomalous scores of different methods as well as the true distribution reported by Sturm [26]. The results are shown in Table 5. Interestingly, majority of the approaches have most of the true outliers in *Disco* and *Reggae* except the *One-Class SVM*. For the *One-Class SVM*, its top 20 includes 14 metal outliers, which are barely detected by the other methods. More specifically, the *One-Class SVM* had a high precision of 14/26 in the *Metal* genre. Since most of the true outliers in the *Metal* genre can be categorized to punk rock according to the definition on the online music library, [1] they could exhibit similar features with subtle differences in the feature space, and they are still detected by the *One-Class SVM*. In *Reggae*, there is a jitter music clip which presents extreme values in the feature space, along with the other outliers. For the *One-Class SVM* in the context of *Reggae*, however, only the jitter instance is captured while the other outliers are missing. These two observations confirm that *One-Class SVM* is especially good at distinguishing the outliers that have subtle differences, and could be easily biased by the outliers with extreme values.

Three of methods have about 10 of the top 20 false outliers in *Pop*. This may due to the variety of *Pop* music in the dataset. For example, although *Pop (12) - Aretha Franklin, Celine Dion, Mariah Carey, Shania Twain, and Gloria Es-*

[1] AllMusic: http://www.allmusic.com/

	True	CLUS	KNN	LOF	SVM	RPCA	RCR
Blues	0	0/2	0/0	0/0	0/0	0/4	0/0
Classical	0	0/0	0/3	0/0	0/0	0/1	0/0
Country	4	2/0	2/0	3/0	0/0	2/0	1/2
Disco	7	5/0	5/0	2/0	0/0	4/0	5/2
Hip-hop	3	1/0	3/3	2/2	3/5	1/1	2/3
Jazz	2	0/7	1/6	1/5	0/0	0/5	2/0
Metal	17	1/0	2/0	5/1	14/0	2/2	2/1
Pop	4	4/10	2/2	2/11	2/8	3/3	2/0
Reggae	7	7/1	5/3	5/1	1/5	7/4	5/2
Rock	2	0/0	0/3	0/0	0/2	1/0	1/10

Table 5. Distribution of the Top 20 Ranked True Outliers/False Outliers among Methods.

tefan "You Make Me Feel Like A Natural Woman" is not identified by the expert as an outlier, it can be argued to be *Soul* music. By the nature of the *One-Class SVM*, this clip is also ranked at the top by its anomalous score. Another interesting observation is that four methods have about 5 *jazz* instances in the top 20 false outliers. Although *jazz* music has strong characteristics and can be easily distinguished by humans, it shows the most significant average variance on its features comparing with other genres. Thus, the methods that calculate the Euclidean distances such as *Clustering*, *KNN*, and *LOF*, and the approaches that absorb variances as errors such as *RPCA* could report more false outliers in this circumstance. We also noticed that *RCR* includes 10 *Rock* false outliers in its top 20. This may because it models the input-output relationship among all genres, and this global-view property thus causes the model mixing up *Rock* with other partially overlapping genres such as *Metal* and *Blues*.

To summarize, outlier detection in music data faces the following challenges compared to other types of data: first, due to the ambiguity in the genre definitions, some of the tracks can be considered as both outliers and non-outliers. This inconsistency may impact the training and testing results for both supervised and unsupervised approaches. In Sturm's [27] work, a risk model is proposed to model the loss of misclassification by the similarity of genres. Second, the music data has temporal dependencies. In the current framework, we aggregate the block-wise feature matrix into a single feature vector as it allows for the immediate use in the context of the state-of-the-art methods. This approach, however, does not keep the temporal changes of the music signals and potentially discards important information for identifying outliers with subtle differences. Third, the extracted low-level features might not be able to capture the high-level concept of music genre, therefore, it is difficult for the outlier detection algorithms to find the outliers agreed on by the human experts. Finally, the outliers are unevenly distributed among genres (e.g., *Metal* has 16 while *Blues* and *Classical* have none), and the data points are also distributed differently in the feature space in each genre. An approach or a specific parameter setting may perform well on some of the genres and fail on others.

5. CONCLUSION

In this paper, we have presented the application of outlier detection methods on a music dataset. Six state-of-the-art approaches have been investigated in the context of music genre recognition, and their performance is evaluated based on their capability of finding the outliers identified by human experts [26]. The results show that all of the methods fail to identify the outliers with reasonably high accuracy. This leaves room for future improvement in the automatic detection of outliers in music data. The experiment results also reveal the main challenges for outlier detection in music genre recognition: first, genre definitions are usually subjective and ambiguous. Second, the temporal dependencies of music need to be modeled. Third, the low-level audio features might not be able to capture the high-level concepts. These challenges may also generalize to other music datasets, and they should be further addressed in future work.

We identify possible directions for future work as: First, as shown in the experiments, a better feature representation should lead to a better performance for the majority of the methods. Therefore, to robustly isolate outliers, a better feature representation for outlier detection algorithms seems to be necessary. Second, since music data has temporal dependencies, the static approach in the current framework might not be feasible. An outlier detection method that can handle the temporal dependencies could potentially show improved performance Third, in the top 20 list for different methods, it is shown that different methods could be sensitive to different types of outliers. An ensemble approach that takes advantage of multiple methods might be considered in future studies.

With our results, we have shown that outlier detection in music datasets is still at a very early stage. To fully characterize a music signal, many challenges and questions still need to be answered. With current advances in feature design and feature learning, however, we expect significant progress to be made in the near future.

6. REFERENCES

[1] Mikhail Atallah, Wojciech Szpankowski, and Robert Gwadera. Detection of significant sets of episodes in event sequences. In *Data Mining, 2004. ICDM '04. Fourth IEEE International Conference on*, pages 3–10, Nov 2004.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: identifying density-based local outliers. *SIGMOD Record*, 29(2):93–104, May 2000.

[4] Patrick L. Brockett, Xiaohua Xia, and Richard A. Derrig. Using kohonen's self-organizing feature map to uncover automobile bodily injury claims fraud. *The Journal of Risk and Insurance*, 65(2):245–274, 1998.

[5] Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):11:1–11:37, June 2011.

[6] Andrea Cerioli. Multivariate outlier detection with high-breakdown estimators. *Journal of the American Statistical Association*, 105(489):147–156, 2009.

[7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computer Survey*, 41(3):15:1–15:58, July 2009.

[8] Santanu Das, Bryan L. Matthews, Ashok N. Srivastava, and Nikunj C. Oza. Multiple kernel learning for heterogeneous anomaly detection: algorithm and aviation safety case study. In *KDD 10': 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 47–56, 2010.

[9] Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy, and Sal Stolfo. A geometric framework for unsupervised anomaly detection. In Daniel Barbará and Sushil Jajodia, editors, *Applications of Data Mining in Computer Security*, pages 77–101. Springer US, Boston, MA, 2002.

[10] Arthur Flexer, Elias Pampalk Gerhard, and Gerhard Widmer. Novelty detection based on spectral similarity of songs arthur flexer. In *in Proc. of the 6 th Int. Symposium on Music Information Retrieval*. Ms, 2005.

[11] Zhouyu Fu, Guojun Lu, Kai Ming Ting, and Dengsheng Zhang. A Survey of Audio-Based Music Classification and Annotation. *IEEE Transactions on Multimedia*, 13(2):303–319, April 2011.

[12] Lars Kai Hansen, Tue Lehn-Schiler, Kaare Brandt Petersen, Jeronimo Arenas-Garcia, Jan Larsen, and Sren Holdt Jensen. Learning and clean-up in a large scale music database. In *European Signal Processing Conference (EUSIPCO)*, pages 946–950, 2007.

[13] Shohei Hido, Yuta Tsuboi, Hisashi Kashima, Masashi Sugiyama, and Takafumi Kanamori. Statistical outlier detection using direct density ratio estimation. *Knowledge Information Systems*, 2011.

[14] Edwin M. Knorr, Raymond T. Ng, and Vladimir Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3–4):237–253, 2000.

[15] Alexander Lerch. *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*. John Wiley and Sons, 2012.

[16] Spiros Papadimitriou, Hiroyuki Kitagawa, Christos Faloutsos, and Phillip B. Gibbons. Loci: fast outlier detection using the local correlation integral. In *Data Engineering, 2003. Proceedings. 19th International Conference on*, pages 315–326, March 2003.

[17] Eun Park, Shawn Turner, and Clifford Spiegelman. Empirical approaches to outlier detection in intelligent transportation systems data. *Transportation Research Record: Journal of the Transportation Research Board*, 1840:21–30, 2003.

[18] Cludia Pascoal, M. Rosrio Oliveira, Antnio Pacheco, and Rui Valadas. Detection of outliers using robust principal component analysis: A simulation study. In Christian Borgelt, Gil González-Rodríguez, Wolfgang Trutschnig, María Asunción Lubiano, María Ángeles Gil, Przemysław Grzegorzewski, and Olgierd Hryniewicz, editors, *Combining Soft Computing and Statistical Methods in Data Analysis*, pages 499–507. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[19] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. *SIGMOD Record*, 29(2):427–438, May 2000.

[20] Marco Riani, Anthony C. Atkinson, and Andrea Cerioli. Finding an unknown number of multivariate outliers. *Journal of the Royal Stats Society Series B*, 71(2):447–466, 2009.

[21] Volker Roth. Outlier detection with one-class kernel fisher discriminants. In *Advances in Neural Information Processing Systems 17*, pages 1169–1176, 2005.

[22] Markus Schedl, Emilia Gómez, and Julián Urbano. Music Information Retrieval: Recent Developments and Applications. *Foundations and Trends in Information Retrieval*, 8:127–261, 2014.

[23] Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, July 2001.

[24] Michael R. Smith and Tony Martinez. Improving classification accuracy by identifying and removing instances that should be misclassified. In *Proceedings of International Joint Conference on Neural Networks*, pages 2690–2697, 2011.

[25] Raheda Smith, Alan Bivens, Mark Embrechts, Chandrika Palagiri, and Boleslaw Szymanski. Clustering approaches for anomaly based intrusion detection. In *Proceedings of intelligent engineering systems through artificial neural networks*, 2002.

[26] Bob L. Sturm. An analysis of the GTZAN music genre dataset. In *Proceedings of the second international ACM workshop on Music Information Retrieval with user-centered and multimodal strategies (MIRUM)*, 2012.

[27] Bob L. Sturm. Music genre recognition with risk and rejection. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2013.

[28] David E. Tyler. Robust statistics: Theory and methods. *Journal of the American Statistical Association*, 103:888–889, 2008.

[29] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[30] Dantong Yu, Gholam Sheikholeslami, and Aidong Zhang. Findout: Finding outliers in very large datasets. Technical report, Dept. of CSE SUNY Buffalo, 1999.

AVA: AN INTERACTIVE SYSTEM FOR VISUAL AND QUANTITATIVE ANALYSES OF VIBRATO AND PORTAMENTO PERFORMANCE STYLES

Luwei Yang[1] **Khalid Z. Rajab**[2] **Elaine Chew**[1]
[1] Centre for Digital Music, Queen Mary University of London
[2] Antennas & Electromagnetics Group, Queen Mary University of London
{l.yang, k.rajab, elaine.chew}@qmul.ac.uk

ABSTRACT

Vibratos and portamenti are important expressive features for characterizing performance style on instruments capable of continuous pitch variation such as strings and voice. Accurate study of these features is impeded by time consuming manual annotations. We present AVA, an interactive tool for automated detection, analysis, and visualization of vibratos and portamenti. The system implements a Filter Diagonalization Method (FDM)-based and a Hidden Markov Model-based method for vibrato and portamento detection. Vibrato parameters are reported directly from the FDM, and portamento parameters are given by the best fit Logistic Model. The graphical user interface (GUI) allows the user to edit the detection results, to view each vibrato or portamento, and to read the output parameters. The entire set of results can also be written to a text file for further statistical analysis. Applications of AVA include music summarization, similarity assessment, music learning, and musicological analysis. We demonstrate AVA's utility by using it to analyze vibratos and portamenti in solo performances of two Beijing opera roles and two string instruments, erhu and violin.

1. INTRODUCTION

Vibrato and portamento use are important determinants of performance style across genres and instruments [4, 6, 7, 14, 15]. Vibrato is the systematic, regular, and controlled modulation of frequency, amplitude, or the spectrum [12]. Portamento is the smooth and monotonic increase or decrease in pitch from one note to the next [15]. Both constitute important expressive devices that are manipulated in performances on instruments that allow for continuous variation in pitch, such as string and wind instruments, and voice. The labor intensive task of annotating vibrato and portamento boundaries for further analysis is a major bottleneck in the systematic study of the practice of vibrato and portamento use.

© Luwei Yang, Khalid Z. Rajab and Elaine Chew. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Luwei Yang, Khalid Z. Rajab and Elaine Chew. "AVA: An Interactive System for Visual and Quantitative Analyses of Vibrato and Portamento Performance Styles", 17th International Society for Music Information Retrieval Conference, 2016.

While vibrato analysis and detection methods have been in existence for several decades [2, 10, 11, 13], there is currently no widely available software tool for interactive analysis of vibrato features to assist in performance and musicological research. Portamenti have received far less attention than vibratos due to the inherent ambiguity in what constitutes a portamento—beyond a note transition, a portamento is a perceptual feature that can only exist if it is recognizable by the human ear—some recent work on the modeling of portamenti can be found in [15].

The primary goal of this paper is to introduce the AVA system [1] for interactive vibrato and portamento detection and analysis. AVA seeks to fill the gap in knowledge discovery tools for expressive feature analysis for continuous pitch instruments. The AVA system is built on recent advances in automatic vibrato and portamento detection and analysis. As even the best algorithm sometimes produces erroneous vibrato or portamento detections, the AVA interface allows the user to interactively edit the detection solutions so as to achieve the best possible analysis results.

A second goal of the paper is to demonstrate the utility of the AVA system across instruments and genres using two datasets, one for voice and the other for string instruments. The vocal dataset comprises of monophonic samples of phrases from two Beijing opera roles, one female one male; the string instruments dataset consists of recordings of a well known Chinese piece on erhu and on violin.

Applications of AVA include music pedagogy and musicological analysis. AVA can be used to provide visual and quantitative feedback in instrumental learning, allowing students to inspect their expressive features and adapt accordingly. AVA can also be used to quantify musicians' vibrato and portamento playing styles for analyses on the ways in which they use these expressive features. It can be used to conduct large-scale comparative studies, for example, of instrumental playing across cultures. AVA's analysis results can also serve as input to expression synthesis engines, or to transform expressive features in recorded music.

The remainder of the paper is organized as follows: Section 2 presents the AVA system and begins with a description of the vibrato and portamento feature detection and analysis modules; Section 3 follows with details of AVA's user interface. Section 4 presents two case studies

[1] The beta version of AVA is available at `luweiyang.com/research/ava-project`.

using AVA to detect and analyse vibratos and portamenti and their properties in two Beijing opera roles, and in violin and erhu recordings. Section 5 closes with discussions and conclusions.

2. FEATURE DETECTION AND ANALYSIS

Figure 1 shows AVA's system architecture. The system takes monophonic audio as input. The pitch curve, which is given by the fundamental frequency, is extracted from the input using the pYIN method [5]. The first part of the system focuses on vibrato detection and analysis. The pitch curve derived from the audio input is sent to the vibrato detection module, which detects vibrato existence using a Filter Diagonalization Method (FDM). The vibratos extracted are then forwarded to the module for vibrato analysis, which outputs the vibrato statistics.

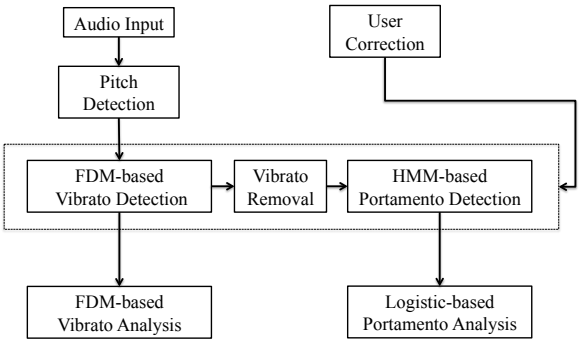


Figure 1. The AVA system architecture.

The next part of the system deals with portamento detection and analysis. The oscillating shapes of the vibratos degrade portamento detection. To ensure the best possible performance for portamento detection, the detected vibratos are flatten using the built-in MATLAB function 'smooth'. The portamento detection module, which is based on the Hidden Markov Model (HMM), uses this vibrato-free pitch curve to identify potential portamenti. A Logistic Model is fitted to each detected portamento for quantitative analysis.

For both the vibrato and portamento modules, if there are errors in detection, the interface allows the user to mark up missing vibratos or portamenti and delete spurious results. Further details on the AVA interface will be given in Section 3.

2.1 Vibrato Detection and Analysis

We use an FDM-based method described in [16] to analyze the pitch curve and extract the vibrato parameters. The advantage of the FDM is its ability to extract sinusoid frequency and amplitude properties for a short time signal, thus making it possible to determine vibrato presence over the span of a short time frame.

Vibrato detection methods can be classified into note-wise and frame-wise methods. Note-wise methods have a pre-requisite note segmentation step before they can determine if the note contains a vibrato [8, 10]. Frame-wise methods divide the audio stream, or the extracted f_0 information, into a number of uniform frames. Vibrato existence is then decided based on information in each frame [2, 11, 13, 16]. The FDM approach constitutes one of the newest frame-wise methods.

Fundamentally, the FDM assumes that the time signal (the pitch curve) in each frame is the sum of exponentially decaying sinusoids,

$$f(t) = \sum_{k=1}^{K} d_k e^{-in\tau\omega_k}, \text{ for } n = 0, 1, \ldots, N, \quad (1)$$

where K is the number of sinusoids required to represent the signal to within some tolerance threshold, and the fitting parameters ω_k and d_k are the complex frequency and complex weight, respectively, of the k-th sinusoid. The aim of the FDM is to find the $2K$ unknowns, representing all ω_k and d_k. A brief summary of the steps is described in Algorithm 1. Further details of the algorithm and implementation are given in [16].

Algorithm 1: The Filter Diagonalization Method

Input: Pitch curve (fundamental frequency)
Output: The frequency and amplitude of the sinusoid with the largest amplitude
Set the vibrato frequency range;
Filter out sinusoids having frequency outside the allowable range;
Diagonalize the matrix given by the pitch curve;
for *each iteration* **do**
  Create a matrix by applying a 2D FFT on the pitch curve;
  Diagonalize this matrix;
  Get the eigenvalues;
  Check that the eigenvalues are within the acceptance range;
end
Compute the frequencies from the eigenvalues;
Calculate the amplitudes from the corresponding eigenvectors;
Return the frequency and amplitude of the sinusoid with the largest amplitude;

Information on vibrato rate and extent fall naturally out of the FDM analysis results. Here, we consider only the frequency and amplitude of the sinusoid having the largest amplitude. The window size is set to 0.125 seconds and step size is one quarter of the window. Given the frequency and amplitude, a Decision Tree determines the likely state of vibrato presence. Any vibrato lasting less than 0.25 seconds is pruned.

A third parameter is reported by the vibrato analysis module, that of sinusoid similarity, which is used to characterize the sinusoid regularity of the shape of the detected

vibrato. The sinusoid similarity is a parameter between 0 and 1 that quantifies the similarity of a vibrato shape to a reference sinusoid using cross correlation (see [14]).

2.2 Portamento Detection and Analysis

Portamenti are continuous variations in pitch connecting two notes. Not all note transitions are portamenti; only pitch slides that are perceptible to the ear are considered portamenti. They are far less well defined in the literature than vibratos, and there is little in the way of formal methods for detecting portamenti automatically.

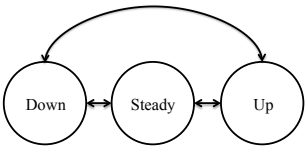


Figure 2. The portamento detection HMM transition network.

To detect portamentos, we create a fully connected three-state HMM using the delta pitch curve as input as shown in Figure 2. The three states are down, steady, and up, which correspond to slide down, steady pitch, and slide up gestures. Empirically, we choose as transitions pobabilities the numbers shown in Table 1, which have worked well in practice. Each down state and up state observation

	Down	Steady	Up
Down	0.4	0.4	0.2
Steady	1/3	1/3	1/3
Up	0.2	0.4	0.4

Table 1. Transition probabilities for portamento detection HMM.

is modeled using a Gamma distribution model. The steady pitch observation is modeled as a sharp needle around 0 using a Gaussian function. The best (most likely) path is decoded using the Viterbi algorithm. All state changes are considered to be boundaries, and the minimum note or transition (portamento) duration is set as 0.09 seconds.

To quantitatively describe each portamento, we fit a Logistic Model to the pitch curve in the fashion described in [15]. The choice of model is motivated by the observation that portamenti largely assume S or reverse S shapes. An ascending S shape is characterized by a smooth acceleration in the first half followed by a deceleration in the second half, with an inflection point between the two processes.

The Logistic Model can be described mathematically as

$$P(t) = L + \frac{(U-L)}{\left(1 + Ae^{-G(t-M)}\right)^{1/B}} , \qquad (2)$$

where L and U are the lower and upper horizontal asymptotes, respectively. Musically speaking, L and U are the antecedent and subsequent pitches of the transition. A, B, G, and M are constants. Furthermore, G can be interpreted as the growth rate, indicating the steepness of the transition slope.

The time of the point of inflection is given by

$$t_R = -\frac{1}{G} \ln\left(\frac{B}{A}\right) + M . \qquad (3)$$

The pitch of the inflection point can then be calculated by substituting t_R into Eqn (2).

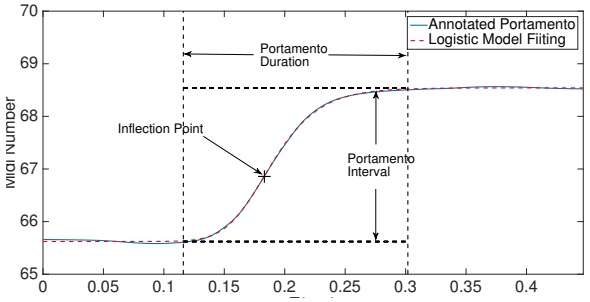


Figure 3. Description of the portamento duration, interval and inflection for a real sample.

Referring to Figure 3, the following portamento parameters are reported by the portamento analysis module and are caculated as follows:

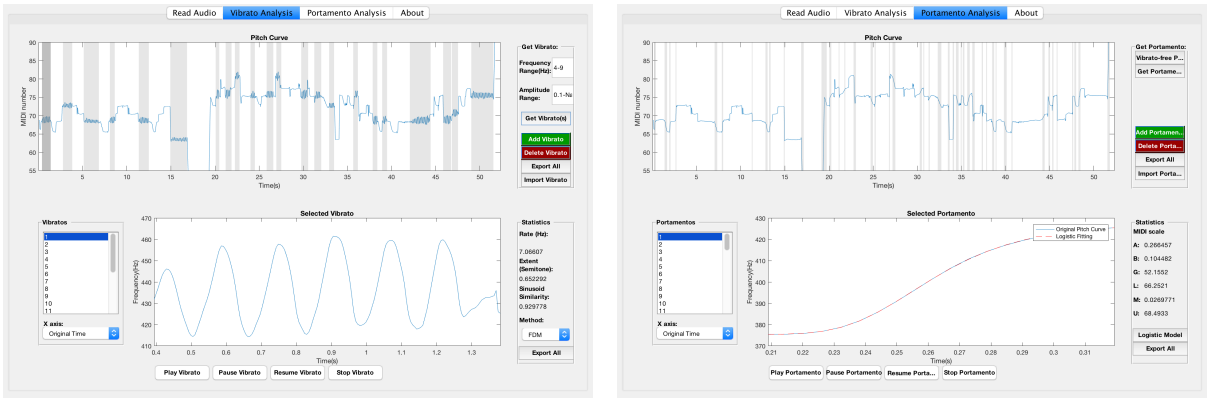
1. Portamento slope: the coefficient G in Eqn (2).
2. Portamento duration (in seconds): the time interval during which the first derivative (slope) of the logistic curve is greater than 0.861 semitones per second (i.e. 0.005 semitones per sample).
3. Portamento interval (in semitones): the absolute difference between the lower (L) and upper (U) asymptotes.
4. Normalized inflection time: time between start of portamento and inflection point time, t_R in Eqn (3), as a fraction of the portamento duration.
5. Normalized inflection pitch: distance between the lower (L) asymptote and the inflection point pitch as a fraction of the portamento interval.

3. THE AVA INTERFACE

The vibrato and portamento detection and analysis methods described above were implemented in AVA using MATLAB.

AVA's Graphical User Interface (GUI) consists of three panels accessed through the tabs: Read Audio, Vibrato Analysis, and Portamento Analysis. The Read Audio panel allows a user to input or record an audio excerpt and obtain the corresponding (fundamental frequency) pitch curve. The Vibrato Analysis and Portamento Analysis panels provide visualizations of vibrato and portamento detection and analysis results, respectively.

Figure 4 shows screenshots of the AVA interface. Figure 4(a) shows the Vibrato Analysis panel analyzing an



(a) Vibrato Analysis

(b) Portamento Analysis

Figure 4. AVA screenshots: the vibrato analysis (left) and portamento analysis (right) panels.

erhu excerpt. Figure 4(b) shows the Portamento Analysis panel analyzing the same excerpt.

Our design principle was to have each panel provide one core functionality while minimizing unnecessary functions having little added value. As vibratos and portamenti relate directly to the pitch curve, each tab shows the entire pitch curve of the excerpt and a selected vibrato or portamento in that pitch curve.

To allow for user input, the Vibrato Analysis and Portamento Analysis panels each have "Add" and "Delete" buttons for creating or deleting highlight windows against the pitch curve. Playback functions allow the user to hear each detected feature so as to inspect and improve detection results. To enable further statistical analysis, AVA can export to a text file all vibrato and portamento annotations and their corresponding parameters.

3.1 Vibrato Analysis Panel

We first describe the Vibrato Analysis Panel, shown in Figure 4(a). The pitch curve of the entire excerpt is presented in the upper part, with the shaded areas marking the detected vibratos. Vibrato existence is determined using the method described in Section 2.1. The computations are triggered using the "Get Vibrato(s)" button in the top right, and the detected vibratos are highlighted by grey boxes on the pitch curve. Users can correct vibrato detection errors using the "Add Vibrato" and "Delete Vibrato" buttons.

The interface allows the user to change the default settings for the vibrato frequency and amplitude ranges; these adaptable limits serve as parameters for the Decision Tree vibrato existence detection process. In this case, with the vibrato frequency range threshold $[4, 9]$ Hz and amplitude range threshold $[0.1, \infty]$ semitones.

On the lower left is a box listing the indices of the detected vibratos. The user can click on each highlighted vibrato on the pitch curve, use the left- or right-arrow keys to navigate from the selected vibrato, or click on one of the indices to select a vibrato. The pitch curve of the vibrato thus selected is presented in the lower plot with corresponding parameters shown to the right of that plot.

In Figure 4(a), the selected vibrato has frequency 7.07 Hz, extent 0.65 semitones, and sinusoid similarity value 0.93. These parameters are obtained using the FDM-based vibrato analysis technique. Alternatively, using the drop down menu currently marked "FDM", the user can toggle between the FDM-based technique and a more basic Max-min method that computes the vibrato parameters from the peaks and troughs of the vibrato pitch contour.

Another drop down menu, labeled "X axis" under the vibrato indices at the bottom left, lets the user to choose between the original time axis and a normalized time axis for visualizing each detected vibrato. A playback function assists the user in vibrato selection and inspection. All detected vibrato annotations and parameters can be exported to a text file at the click of a button to facilitate further statistical analysis.

3.2 Portamento Analysis Panel

Next, we present the functions available on the Portamento Analysis Panel, shown in Figure 4(b).

In the whole-sample pitch curve of Figure 4(b), the detected vibratos of Figure 4(a) have been flattened to improve portamento detection. Clicking on the "Get Portamentos" button initiates the process of detecting portamenti. The "Logistic Model" button triggers the process of fitting Logistic Models to all the detected portamenti.

Like the Vibrato Analysis panel, the Portamento Analysis panel also provides "Add Portamento" and "Delete Portamento" buttons for the user to correct detection errors. The process for selecting and navigating between detected portamenti is like that for the Vibrato Analysis panel.

When a detected portamento is selected, the best-fit Logistic model is shown as a red line against the original portamento pitch curve. The panel to the right shows the corresponding Logistic Model parameters. In the case of the portamento highlighted in Figure 4(b), the growth rate is 52.15 and the lower and upper asymptotes are 66.25 and 68.49 (in MIDI number), respectively, which could be interpreted as the antecedent and subsequent pitches. From this, we infer that the transition interval is 2.24 semitones.

As with the Vibrato Analysis panel, a playback function assists the user in portamento selection and inspection. Again, all detected portamento annotations and parameters can be exported to a text file at the click of a button to facilitate further statistical analysis.

4. CASE STUDIES

This section demonstrates the application of AVA to two sets of data: one for two major roles in Beijing opera, and one for violin and erhu recordings.

4.1 Case 1: Beijing Opera (Vocal)

Vibrato and portamento are widely and extensively employed in opera; the focus of the study here is on singing in Beijing opera. Investigations into Beijing opera singing include [9]. For the current study, we use selected recordings from the Beijing opera dataset created by Black and Tian [1]; the statistics on the amount of vibrato and portamenti, based on manual annotations, in the sung samples are shown in Table 2. The dataset consists of 16 monophonic recordings by 6 different Chinese opera singers performing well-known phrases from the Beijing opera roles Laosheng(老生) and Zhengdan(正旦).

No.	Role	Duration(s)	# Vibratos	# Portamenti
1	Laosheng	102	49	94
2		184	62	224
3		91	56	159
4		95	49	166
5		127	26	115
6		147	51	106
7		168	47	144
8		61	19	89
9	Zhengdan	159	47	173
10		80	24	49
11		119	42	176
12		50	24	71
13		155	57	212
14		41	12	48
15		180	59	219
16		70	34	87

Table 2. Beijing opera dataset.

Each recording was uploaded to the AVA system for vibrato and portamento detection and analysis. Detection errors were readily corrected using the editing capabilities of AVA. Figure 5 presents the resulting histogram envelops of the vibrato and portamento parameter values, each normalized to sum to 1, for the Zhengdan (red) and Laosheng (blue) roles. Translucent lines show the parameter's distributions for individual recordings, and bold lines show the aggregate histogram for each role.

The histograms show the similarities and differences in the underlying probability density functions. Visual inspection shows that the singing of the Zhengdan and Laosheng roles to be most contrastive in the vibrato extents, with peaks at around 0.5 and 0.8 semitones, respectively. A Kolmogorov-Smirnov (KS) test [2] shows that the histogram envelopes of vibrato extent from Laosheng and Zheng to be significant different ($p = 2.86 \times 10^{-4}$) at 1% significant level. The same test shows that the distributions for vibrato rate ($p = 0.0536$) and vibrato sinusoid similarity ($p = 0.0205$) are not significant different. Significant differences are found between the singing of the Laosheng and Zhengdan roles for the portamento slope ($p = 1.80 \times 10^{-3}$) and interval ($p = 2.30 \times 10^{-34}$) after testing using the KS test; differences in duration ($p = 0.345$), normalized inflection time ($p = 0.114$) and normalized inflection pitch ($p = 1.00$) are not significant.

4.2 Case 2: Violin vs. Erhu (String)

Here, we demonstrate the usability of the AVA system on the analysis of vibrato and portamento performance styles on erhu and violin. The study centers on a well known Chinese piece *The Moon Reflected on the Second Spring* (二泉映月) [3]. The study uses four recordings, two for erhu and two more for violin. Table 3 lists the details of the test set, which comprises of a total of 23.6 minutes of music; with the help of AVA, 556 vibratos and 527 portamenti were found, verified, and analysed.

No.	Instrument	Duration(s)	# Vibratos	# Portamenti
1	Erhu	446	164	186
2		388	157	169
3	Violin	255	131	91
4		326	104	81

Table 3. Erhu and violin dataset.

The histograms of the vibrato and portamento parameters are summarized in Figure 6. Again, we use the KS test to assess the difference in the histograms between violin and erhu. As with the case for the Beijing opera roles, the most significant difference between the instruments is found in the vibrato extent ($p = 2.70 \times 10^{-3}$), with the vibrato extent for the erhu about twice that for violin (half semitone vs. quarter semitone). There is no significant difference found between erhu and violin for vibrato rate ($p = 0.352$) and sinusoid similarity ($p = 0.261$), although the plots show that the violin recordings have slightly faster vibrato rates and lower sinusoid similarity.

Regarding portamento, the portamento interval histogram has a distinct peak at around three semitones for both violin and erhu, showing that notes separated by this gap is more frequently joined by portamenti. The difference between the histograms is highly insignificant ($p = 0.363$). The most significant difference between violin and erhu portamenti histograms is observed for the slope ($p = 1.51 \times 10^{-4}$). Inspecting the histograms, violinists tend to place the normalized inflection time after the midpoint and erhu players before the midpoint of the portamento duration. However, it is not supported by the KS

[2] http://uk.mathworks.com/help/stats/kstest2.html

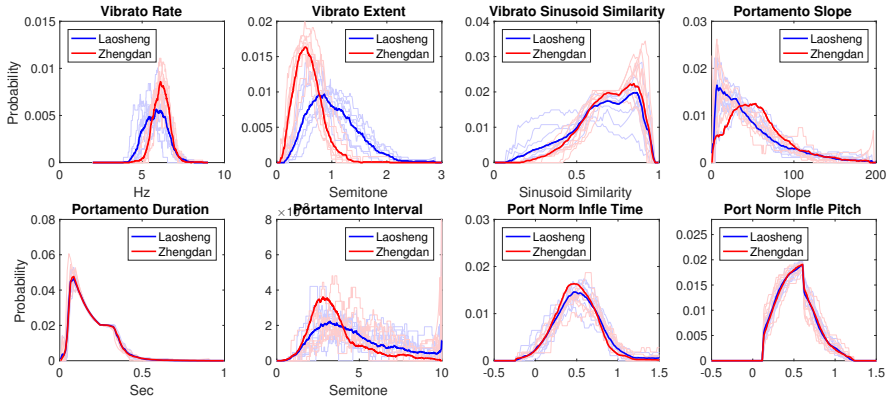


Figure 5. Histogram envelopes of vibrato and portamento parameters for Beijing opera roles: Laosheng (blue) and Zhengdan (red). Translucent lines show histograms for individual singers; bold line shows aggregated histograms for each role.

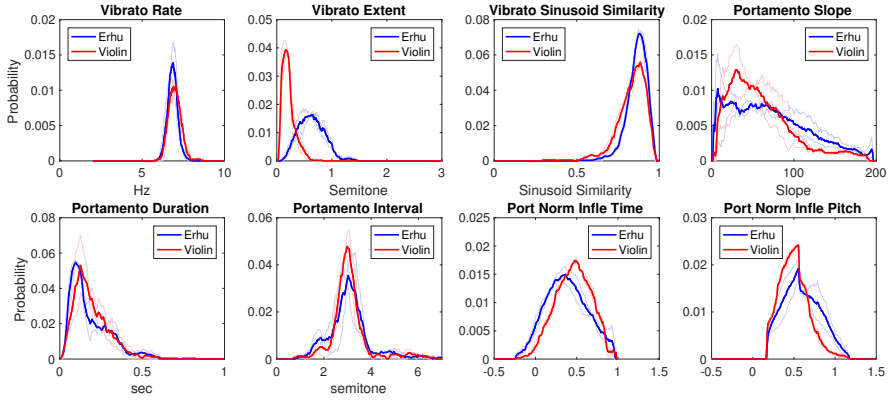


Figure 6. Histogram envelopes of vibrato and portamento parameters for two instruments: erhu (blue) and violin (red). Translucent lines show histograms for individual players; bold line shows aggregated histograms for each instrument.

test ($p = 0.256$). The duration ($p = 0.344$) and normalized inflection pitch ($p = 0.382$) doesn't show significant results.

5. CONCLUSIONS AND DISCUSSIONS

We have presented an interactive vibrato and portamento detection and analysis system, AVA. The system was implemented in MATLAB, and the GUI provides interactive and intuitive visualizations of detected vibratos and portamenti and their properties. We have also demonstrated its use in analyses of Beijing opera and string recordings.

For vibrato detection and analysis, the system implements a Decision Tree for vibrato detection based on FDM output and an FDM-based vibrato analysis method. The system currently uses a Decision Tree method for determining vibrato existence; a more sophisticated Bayesian approach taking advantage of learned vibrato rate and extent distributions is described in [16]. While the Bayesian approach has been shown to give better results, it requires training data; the prior distributions based on training data can be adapted to specific instruments and genres.

For portamento detection and analysis, the system uses an HMM-based portamento detection method with Logistic Models for portamento analysis. Even though a threshold has been set to guarantee a minimum note transition duration, the portamento detection method sometimes misclassifies normal note transitions as portamenti, often for notes having low intensity (dynamic) values. While there were significant time savings over manual annotation, especially for vibrato boundaries, corrections of the automatically detected portamento boundaries proved to be the most time consuming part of the exercise. Future improvements to the portamento detection method could take into account more features in addition to the delta pitch curve.

For the Beijing opera study, the two roles differed significantly in vibrato extent, and in portamento slope and interval. The violin and erhu study showed the most significant differences in vibrato extent and portamento slope. Finally, the annotations and analyses produced with the help of AVA will be made available for further study.

6. ACKNOWLEDGEMENTS

This project is supported in part by the China Scholarship Council.

7. REFERENCES

[1] Dawn A. A. Black, Li Ma, and Mi Tian. Automatic identification of emotional cues in Chinese opera singing. In *Proc. of the 13th International Conference on Music Perception and Cognition and the 5th Conference for the Asian-Pacific Society for Cognitive Sciences of Music (ICMPC 13-APSC0M 5)*, 2014.

[2] Perfecto Herrera and Jordi Bonada. Vibrato extraction and parameterization in the spectral modeling synthesis framework. In *Proc. of the Digital Audio Effects Workshop*, volume 99, 1998.

[3] Yanjun Hua. *Erquanyingyue*. Zhiruo Ding and Zhanhao He, violin edition, 1958. Musical Score.

[4] Heejung Lee. Violin portamento: An analysis of its use by master violinists in selected nineteenth-century concerti. In *Proc. of the 9th International Conference on Music Perception and Cognition*, 2006.

[5] Matthias Mauch and Simon Dixon. pyin: A fundamental frequency estimator using probabilistic threshold distributions. In *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2014.

[6] Tin Lay Nwe and Haizhou Li. Exploring vibrato-motivated acoustic features for singer identification. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(2):519–530, 2007.

[7] Tan Hakan Özaslan, Xavier Serra, and Josep Lluis Arcos. Characterization of embellishments in ney performances of makam music in turkey. In *Proc. of the International Society for Music Information Retrieval Conference*, 2012.

[8] Hee-Suk Pang and Doe-Hyun Yoon. Automatic detection of vibrato in monophonic music. *Pattern Recognition*, 38(7):1135–1138, 2005.

[9] Rafael Caro Repetto, Rong Gong, Nadine Kroher, and Xavier Serra. Comparison of the singing style of two jingju schools. In *Proc. of the International Society for Music Information Retrieval Conference*, 2015.

[10] Stéphane Rossignol, Philippe Depalle, Joel Soumagne, Xavier Rodet, and Jean-Luc Collette. Vibrato: detection, estimation, extraction, modification. In *Proc. of the Digital Audio Effects Workshop*, 1999.

[11] Jose Ventura, Ricardo Sousa, and Anibal Ferreira. Accurate analysis and visual feedback of vibrato in singing. In *Proc. of the IEEE 5th International Symposium on Communications, Control and Signal Processing*, pages 1–6, 2012.

[12] Vincent Verfaille, Catherine Guastavino, and Philippe Depalle. Perceptual evaluation of vibrato models. In *Proc. of the Conference on Interdisciplinary Musicology*, March 2005.

[13] Henrik von Coler and Axel Roebel. Vibrato detection using cross correlation between temporal energy and fundamental frequency. In *Proc. of the Audio Engineering Society Convention 131*. Audio Engineering Society, 2011.

[14] Luwei Yang, Elaine Chew, and Khalid Z. Rajab. Vibrato performance style: A case study comparing erhu and violin. In *Proc. of the 10th International Conference on Computer Music Multidisciplinary Research*, 2013.

[15] Luwei Yang, Elaine Chew, and Khalid Z. Rajab. Logistic modeling of note transitions. In *Mathematics and Computation in Music*, pages 161–172. Springer, 2015.

[16] Luwei Yang, Khalid Z. Rajab, and Elaine Chew. Filter diagonalisation method for music signal analysis: Frame-wise vibrato detection and estimation. *Journal of Mathematics and Music*, 2016. Under revision.

COMPOSER RECOGNITION BASED ON 2D-FILTERED PIANO-ROLLS

Gissel Velarde[1] **Tillman Weyde**[2] **Carlos Cancino Chacón**[3]
David Meredith[1] **Maarten Grachten**[3]
[1] Department of Architecture Design & Media Technology, Aalborg University, Denmark
[2] Department of Computer Science, City University London, UK
[3] Austrian Research Institute for Artificial Intelligence, Austria
{gv, dave}@create.aau.dk, t.e.weyde@city.ac.uk, {carlos.cancino, maarten.grachten}@ofai.at

ABSTRACT

We propose a method for music classification based on the use of convolutional models on symbolic pitch–time representations (i.e. piano-rolls) which we apply to composer recognition. An excerpt of a piece to be classified is first sampled to a 2D pitch–time representation which is then subjected to various transformations, including convolution with predefined filters (Morlet or Gaussian) and classified by means of support vector machines. We combine classifiers based on different pitch representations (MIDI and morphetic pitch) and different filter types and configurations. The method does not require parsing of the music into separate voices, or extraction of any other predefined features prior to processing; instead it is based on the analysis of texture in a 2D pitch–time representation. We show that filtering significantly improves recognition and that the method proves robust to encoding, transposition and amount of information. On discriminating between Haydn and Mozart string quartet movements, our best classifier reaches state-of-the-art performance in leave-one-out cross validation.

1. INTRODUCTION

Music classification has occupied an important role in the music information retrieval (MIR) community, as it can immediately lead to musicologically interesting findings and methods, whilst also being immediately applicable in, for example, recommendation systems, music database indexing, music generation and as an aid in resolving issues of spurious authorship attribution.

Composer recognition, one of the classification tasks addressing musical style discrimination (among genre, period, origin identification, etc.), has aroused more attention in the audio than in the symbolic domain [13]. Particularly in the symbolic domain, the string quartets by Haydn and Mozart have been repeatedly studied [10, 12, 13, 24], since discriminating between Haydn and Mozart has been found to be a particularly challenging composer recognition task [24].

In this study, we propose a novel method and evaluate it on the classification of the string quartet movements by Haydn and Mozart. The method is based on the use of convolutional models on symbolic pitch–time representations (i.e. piano-rolls). An excerpt of a piece to be classified is first sampled to a 2D pitch–time representation which is then subjected to various transformations, including convolution with predefined filters (Morlet or Gaussian) and classified by means of Support Vector Machines (SVM).

© Gissel Velarde, Carlos Cancino Chacón, Tillman Weyde, David Meredith, Maarten Grachten. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Gissel Velarde, Carlos Cancino Chacón, Tillman Weyde, David Meredith, Maarten Grachten. "Composer Recognition based on 2D-Filtered Piano-Rolls", 17th International Society for Music Information Retrieval Conference, 2016.

2. RELATED WORK

Typically it is seen that computational methods use some kind of preprocessing to extract melody and harmony. Previous computational methods addressing composer discrimination of polyphonic works required defining sets of musical features or style makers, and/or relied on the encoding of separate parts or voices [10,12,13,24]. However, hard-coded musical features require musical expertise and may not perform similarly on different datasets [24], while the performance of methods relying on separate encoding of voice parts could be affected if voices are not encoded separately or even be unusable.

In order to avoid the requirements of previous methods, we aim to develop a more general approach studying the texture of pitch–time representations (i.e. piano-rolls) in the two-dimensional space. Previous studies did not address musical texture as it is proposed here.

Next, we review previous work that employs 2D music representations (2.1), and briefly sketch the background of the use of convolutional methods for machine perception and classification (2.2).

2.1 Representing music with 2D images

Visually motivated features generated from spectrograms have been successfully used for music classification (see [5, 28]). This success may be partly due to the fact that similar principles of perceptual organization operate in both vision and hearing [8]. The Gestalt principles of proximity, similarity and good continuation, originally developed to account for perceptual organization in vision, have also been used to explain the way that listeners organize

sonic events into streams and chunks [3, 7, 16]. Moreover, other studies suggest direct interaction between visual and auditory processing in common neural substrates of the human brain, which effectively integrates these modalities in order to establish robust representations of the world [9, 11, 21].

Graphical notation systems have been used since ancient times to transmit musical information [27]. Moreover, most Western music composed before the age of recording survives today only because of transmission by graphical notation — as staff notation, tablature, neumatic notation, etc. Standard graphical musical notation methods have proved to be extremely efficient and intuitive, possibly in part due to the natural mapping of pitch and time onto two orthogonal spatial dimensions.

2.2 Convolutional models

Convolutional models have been used extensively to model the physiology and neurology of visual perception. For example, in 1980, Daugman [6] and Marčelja [17] modeled receptive field profiles in cortical simple cells with parametrized 2D Gabor filters. In 1987, Jones and Palmer [14] showed that receptive-field profiles of simple cells in the visual cortex of a cat are well described by the real parts of complex 2D Gabor filters. More recently, Kay et al. [15] used a model based on Gabor filters to identify natural images from human brain activity. In our context, the Gabor filter is equivalent to the Morlet wavelet which we have used as a filter in the experiments described below.

Filters perform tasks like contrast enhancement or edge detection. In image classification, filtering is combined with classification algorithms such as SVM or neural networks for object or texture recognition [2, 23].

In the remainder of this paper, we present our proposed method in detail (3). Then, we report the results of our experiments (4) and finally, state our conclusions (5).

3. METHOD

Figure 1 provides an overview of our proposed method. As input, the method is presented with excerpts from pieces of music in symbolic format. Then, in the *sampling* phase, a 2D image is derived from each input file in the form of a piano-roll. After the *sampling* phase, various *tranformations* are applied to the images before carrying out the final *classification* phase, which generates a class label for the input file using an SVM. Details of each phase are given below. We begin by describing the *sampling* phase, in which symbolic music files are transformed into images of piano-rolls.

3.1 Sampling piano-roll images from symbolic representations

3.1.1 MIDI note numbers encoding

Symbolic representations of music (e.g. MIDI files) encode each note's pitch, onset and duration. We encoded pitch as an integer from 1 to 128 using MIDI note numbers (MNN), where C4 or *middle C* is mapped to MNN 60. Onset and duration are temporal attributes measured in quarter notes (qn).

3.1.2 Morphetic pitch encoding

The pitch name of a note is of the form <letter name><alteration><octave number>, e.g. C♯4. By removing the <alteration> and mapping all note names with the same <letter name> and <octave number> to the same number we reduce the space to *morphetic pitch*: an integer corresponding to the vertical position of the note on a musical staff.

We use a pitch-spelling algorithm by Meredith called *PS13s1* [18], to compute the pitch names of notes. The *PS13s1* algorithm has been shown to perform well on classical music of the type considered in this study. The settings of the *PS13s1* algorithm used here are the same as in [18], [1] with the *pre-context* parameter set to 10 notes and the *post-context* set to 42 notes. These parameters define a context window around the note to be spelt, which is used to compute the most likely pitch name for the note, based on the extent to which the context implies each possible key. When transposing a pattern within a major or minor scale (or, indeed, any scale in a diatonic mode), as is common practice in tonal (and modal) music, chromatic pitch intervals within the pattern change although the transposed pattern is still recognized by listeners as an instance of the same musical motif [8]. Morphetic pitch intervals are invariant to within-scale transpositions. We hypothesize that preserving this tonal motif identity might improve the performance of our models.

3.1.3 Piano-rolls (p_{70qn})

Symbolic representations of music are sampled to 2D binary images of size $P \times T$ pixels taking values of 0 or 1, called piano-roll representations. Our piano-roll representations are sampled from the first 70 qn of each piece, using onset in qn, duration in qn and either MNN or morphetic pitch, with a sampling rate of 8 samples per qn. We denote such representations by p_{70qn}. Each note of a piece symbolically encoded is described as an ordered tuple (onset, duration, pitch). The onsets are shifted, so that the first note starts at 0 qn. The piano-roll image is initialized with zeros and filled with ones for each sampled note. Its rows correspond to pitch and columns to samples in time. For each note, its onset and duration are multiplied by the sampling rate and rounded to the nearest integer. Note that since the tempo in terms of quarter notes per minute varies across pieces in our test corpora, the resulting samples vary in physical duration.

3.1.4 Piano-rolls (p_{400n})

As an alternative to the 70 qn piano roll excerpts, p_{70qn}, defined in 3.1.3 above, we also tested the methods on piano-roll excerpts consisting of the first 400 notes of each piece.

[1] We use a Java implementation of the *PS13s1* algorithm by David Meredith that takes MIDI files as input. **kern files are first converted to MIDI. Then we use the function *writemidi_seconds* by Christine Smit: `http://www.ee.columbia.edu/~csmit/midi/matlab/html/example_script1.html#2`

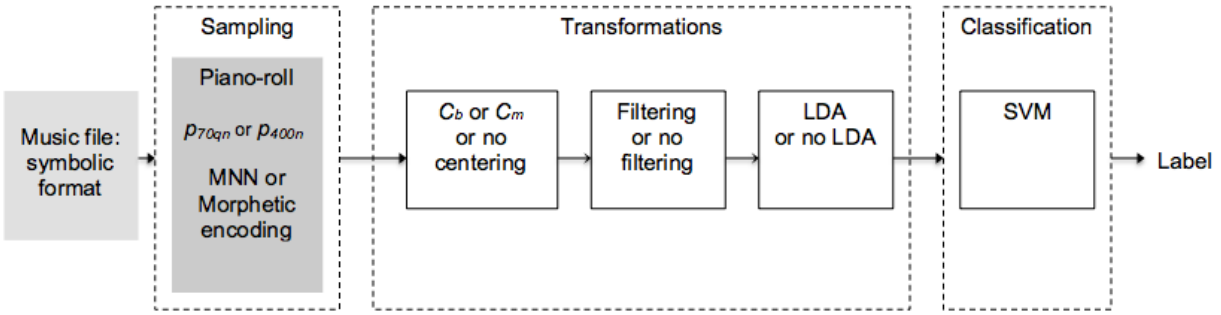


Figure 1. Overview of the method. Music, represented symbolically, is first sampled to 2D images of piano-rolls. Then, various transformations or processing steps are applied to the images, including convolution with predefined filters. The order of applying these transformations is from left to right. Finally, the images are classified with an SVM.

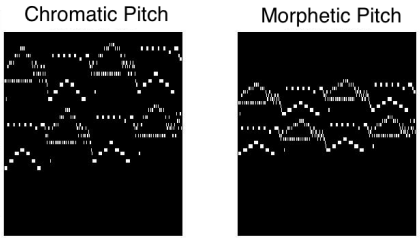


Figure 2. Piano-roll representation using MNN (left) and morphetic pitch (right) of the first 48 qn of Prelude 3 in C♯ major, BWV 848 by Bach. Note that the approximately similar inverted "V" shaped patterns in the left-hand figure are transformed into patterns of exactly the same shape in the right-hand figure.

We denote this type of representation by p_{400n}. These p_{400n} representations were produced by sampling in the way described in section 3.1.3, but using the first 400 notes instead of the first 70 qn of a piece and sampling to a size of $P \times T$ pixels. If a piece has fewer than 400 notes, all notes of the piece are represented. This representation is used to approximately normalize the amount of information per image.

In the next phase of our proposed method with a single classifier, as seen in Figure 1, various transformations or processing steps are applied which will be described as follows.

3.2 Transformations

We explore the effect of applying transformations or processing techniques to the piano-roll images. These transformations are applied in order to find a suitable normalization (i.e., alignment between the images) before classification, and to test the robustness of the method to transformations of the input data that would not be expected to reduce the performance of a human expert (cf. [22]). We now consider each of these transformations in turn.

3.2.1 Pitch range centering (C_b)

Typically, the pitch range of a piece in a piano-roll representation does not extend over the full range of possible MIDI note number values. We hypothesized that we could improve performance by transposing each piano roll so that its pitch range is centered vertically in its image. That is, for a piano-roll image of size $P \times T$ pixels, we translated the image by $y_s = (P - (y_d + y_u))/2$ pixels vertically, where y_d and y_u are the lower and upper co-ordinates, respectively, of the bounding box of the piano roll (i.e., corresponding to the minimum and maximum pitches, respectively, occurring in the piano roll). This transformation is used to test robustness to pitch transposition.

3.2.2 Center of mass centering (C_m)

An image p of size $P \times T$ pixels is translated so that the centroid of the piano roll occurs at the center of the image. We denote the centroid by $(\bar{x}, \bar{y}) = (M_{10}/M_{00}, M_{01}/M_{00})$, where $M_{ij} = \sum_x \sum_y x^i y^j p(x, y)$. The elements of the image are shifted circularly to the central coordinates (x_c, y_c) of the image, where $(x_c = T/2)$ and $(y_c = P/2)$, an amount of $(x_c - \bar{x})$ pixels on the x-axis, and $(y_c - \bar{y})$ pixels on the y-axis. In this case, circular shift is applied to rows and columns of p. In the datasets used for the experiments, in 5% of the pieces with MNN encoding, one low-pitch note was shifted down by this transformation and wrapped around so that it became a high-pitched note (in one piece there were four low-pitch notes shifted to high pitch-notes after circular shift). However, this transformation caused most pieces to be shifted and wrapped around in the time dimension so that, on average, approximately the initial 2 quarter notes of each representation were transferred to the end.

3.2.3 Linear Discriminant Analysis

We apply Linear Discriminant Analysis (LDA) [4] solving the singularity problem by Singular Value Decomposition and Tikhonov regularization to find a linear subspace for

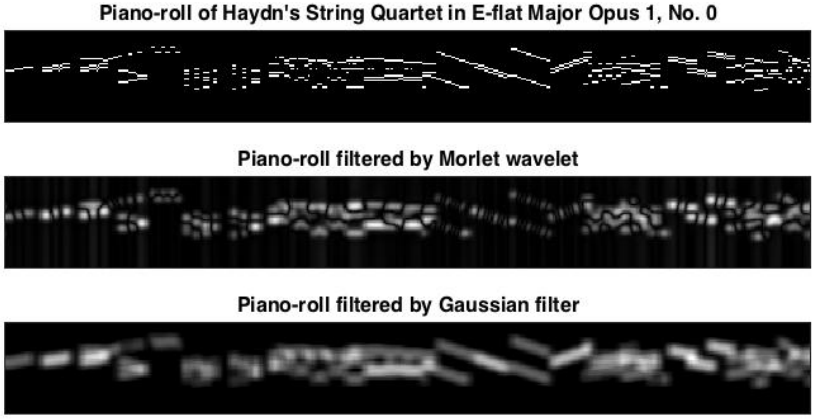


Figure 3. Piano-roll (p_{400n}) morphetic pitch representation (top) of Haydn's String Quartet in E-flat Major Opus 1, No. 0 and its transformations filtered by the Morlet wavelet at a scale of 2 pixels oriented of 90 degrees (second image), and by a Gaussian filter of size 9×9 pixels with $\sigma = 3$ (third image). p_{400n} and its filtered versions are each 56×560 pixels.

discrimination between classes. [2]

3.2.4 Filtering

Images are convolved with pre-defined filters (Morlet wavelet or a Gaussian filter). We apply the continuous wavelet transform (CWT) [1], with the Morlet wavelet ψ at fixed scale a and rotation angle θ

$$\psi_{a,\theta}(x,y) = a^{-1}\psi(a^{-1}r_{-\theta}(x,y)) \quad (1)$$

with rotation r_θ

$$r_\theta(x,y) = (x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta), 0 \leq \theta < 2\pi. \quad (2)$$

where

$$\psi(x,y) = e^{ik_0 y}e^{-\frac{1}{2}(\varepsilon^{-1}x^2+y^2)} \quad (3)$$

with frequency $k_0 = 6$ and $\varepsilon = 1$.

The filtered images are the absolute values of the real part of the wavelet coefficients. We test a defined set of scales and angles (see section 4). The selection of scale and angle of orientation are those that yield the best classification as in [25].

We also filter images with a rotationally symmetric Gaussian low-pass filter g:

$$g(x,y) = e^{\frac{-(x^2+y^2)}{2\sigma^2}} \quad (4)$$

where x and y are the distances from the origin in the horizontal and vertical axis, respectively.

We test a defined set of filter sizes h and σ values (see section 4). The selection of the size h of the filter and the value of σ are those that yield the best classification. As an example of the effect of filtering, Figure 3 shows the piano-roll image, p_{70qn} of Haydn's String Quartet in E-flat Major Opus 1, No. 0 and the filtered images obtained by the convolution with Morlet wavelet and Gaussian filter.

[2] We use Deng Cai's LDA implementation version 2.1: `http://www.cad.zju.edu.cn/home/dengcai/Data/DimensionReduction.html`.

3.3 Classification with support vector machines

For classification, we use SVM with the Sequential Minimal Optimization (SMO) method to build an optimal hyperplane that separates the training samples of each class using a linear kernel [19]. Samples are transformed images of size $P \times T$ if they are not reduced by LDA. If LDA is applied, samples are points in 1D. Each sample is normalized around its mean, and scaled to have unit standard deviation before training. The Karush–Kuhn–Tucker conditions for SMO are set to 0.001.

4. EXPERIMENTS

We used a set of movements from string quartets by Haydn and Mozart, two composers that seemed to have influenced each other on this musical form. Walthew [26] observes that "Mozart always acknowledged that it was from Haydn that he learnt how to write String Quartets" and, in his late string quartets, Haydn was directly influenced by Mozart.

Distinguishing between string quartet movements by Haydn and Mozart is a difficult task. Sapp and Liu [20] have run an online experiment to test human performance on this task and found, based on over 20000 responses, that non-experts perform only just above chance level, while self-declared experts achieve accuracies up to around 66%.

Classification accuracy—that is, the proportion of pieces in the test corpus correctly classified—has been the established evaluation measure for audio genre and composer classification since the MIREX 2005 competition [3] and also for symbolic representations [12, 13, 24].

In our experiments we used the same dataset as in [24] consisting of 54 string quartet movements by Haydn and 53 movements by Mozart, encoded as **kern files, [4] and

[3] See `http://www.music-ir.org/mirex/wiki/2005:Main_Page`.

[4] `http://www.music-cog.ohio-state.edu/Humdrum/representations/kern.html`

	Pitch–time representation	Morlet LDA	Gauss LDA	NF LDA	Morlet	Gauss	NF
Morphetic pitch	p_{70qn}	65.4	58.9	57.9	53.3	68.2	58.9
	$C_b(p_{70qn})$	65.4	60.7	47.7	57.9	63.6	51.4
	$C_m(p_{70qn})$	53.3	60.7	52.3	64.5	59.8	56.1
	p_{400n}	67.3	**80.4**	57.0	63.6	72.9	55.1
	$C_b(p_{400n})$	62.6	72.9	54.2	61.7	66.4	53.3
	$C_m(p_{400n})$	65.4	65.4	55.1	66.4	70.1	53.3
MNN	p_{70qn}	64.5	67.3	66.4	62.6	66.4	64.5
	$C_b(p_{70qn})$	**70.1**	61.7	63.6	67.3	61.7	61.7
	$C_m(p_{70qn})$	63.6	57.9	57.0	66.4	56.1	54.2
	p_{400n}	66.4	69.2	64.5	65.4	63.6	64.5
	$C_b(p_{400n})$	54.2	64.5	52.3	58.9	58.9	49.5
	$C_m(p_{400n})$	53.3	62.6	42.1	56.1	63.6	44.9

Table 1. Haydn and Mozart String Quartet classification accuracies in leave-one-out cross validation for different configurations of classifiers (NF = no filtering).

evaluated our method's classification accuracies in leave-one-out cross-validation as it was done in [24].

Table 1 shows the classification accuracies (mean values) obtained in leave-one-out cross-validation for images of size 56×560 pixels. The standard deviation values are not presented, as they are not informative. The standard deviation can be derived from the accuracy in this case (accuracy of binary classification in leave-one-out cross-validation). The filters of the classifiers were tuned according to their classification accuracy over the different pitch–time representations. The angle of orientation of the Morlet wavelet was set to 90 degrees. This orientation was chosen out of a selection of angles (0, 45, 90 and 135 degrees). The scale was set to 2 pixels, selected varying its value from 1 to 9 pixels. The Gaussian filter was tested with pixel sizes of 1 to 10 pixels, with values of σ ranging from 1 to 4 pixels. Gaussian filters were set to 9 pixels and $\sigma = 3$. The best classifier using MNN encoding corresponds to a classifier operating on pitch–time representation $C_b(p_{70qn})$, filtered by Morlet wavelet oriented 90 degrees at a scale of 2 pixels, and LDA reduction. The best classifier of all reaches state-of-the-art performance with an accuracy of 80.4%. This classifier corresponds to a pitch–time representation p_{400n} in morphetic pitch encoding, filtered by a Gaussian filter of size 9 pixels and $\sigma = 3$, and LDA reduction. It misclassified 12 movements by Haydn and 9 by Mozart. The misclassified movements (mov.) are shown in Table 2. Due to our model section, it could be that the results present some overfitting.

From the results in Table 1 we observe that filtering significantly improves recognition at 5% significance level (Wilcoxon rank sum $= 194.5$, $p = 0.0107$, $n = 12$, with Morlet wavelet), (Wilcoxon rank sum $= 203$, $p = 0.0024$, $n = 12$, with Gaussian filter), and it is not significantly different to filter with Morlet or Gaussian filters (Wilcoxon rank sum $= 133$, $p = 0.3384$, $n = 12$). On the other side, there is not sufficient evidence to conclude that LDA improves recognition (Wilcoxon rank sum $= 154$, $p = 0.8395$, $n = 12$).

Movements by Haydn	**Movements by Mozart**
Op 1, N. 0, mov. 4	K. 137, mov. 3
Op 1, N. 0, mov. 5	K. 159, mov. 3
Op 9, N. 3, mov. 1	K. 168, mov. 2
Op 20, N. 6, mov. 2	K. 168, mov. 3
Op 20, N. 6, mov. 4	K. 428, mov. 3
Op 50, N. 1, mov. 3	K. 465, mov. 2
Op 64, N. 1, mov. 2	K. 465, mov. 4
Op 64, N. 4, mov. 2	K. 499, mov. 1
Op 64, N. 4, mov. 3	K. 499, mov. 4
Op 71, N. 2, mov. 2	
Op 103, mov. 1	
Op 103, mov. 2	

Table 2. Misclassified movements of our best classifier.

We study the effect of encoding (MNN vs. morphetic pitch), transposition (not centering vs. centering with C_b) and the amount of information (p_{70qn} vs. P_{400n}). The center of mass centering C_m was not evaluated, as this transformation may affect human recognition. Considering all results in Table 1 obtained with filtering and excluding the ones obtained with C_m, the difference in encoding between MNN and morphetic pitch is not significant at %5 significance level (Wilcoxon rank sum $= 269.5$, $p = 0.8502$, $n = 16$), nor are the results significantly different with or without centering C_b (Wilcoxon rank sum $= 311.5$, $p = 0.0758$, $n = 16$), neither it is significantly different to use p_{70qn} or P_{400n} (Wilcoxon rank sum $= 242$, $p = 0.4166$, $n = 16$). These findings suggest that the method based on 2D-Filtered piano-rolls is robust to transformations such as encoding, transposition, and amount of information that are considered not to affect human perception.

In Table 3, we list all previous studies where machine-learning methods have been applied to this Haydn/Mozart discrimination task. A direct comparison can be made between the classification accuracy achieved by the method of van Kranenburg and Backer [24] and our proposed method, as we used the same dataset. The datasets used by the other approaches in Table 3 were not available for us to test our method and make direct comparisons. Hontanilla et al. [13] used a subset of the set used in [24]: 49 string quartets movements by Haydn and 46 string quartets movements by Mozart [13]. Hillewaere et al. [12] extended van Kranenburg and Backer's [24] dataset to almost double its size, including several movements from the period 1770–1790. Herlands et al. [10] used a dataset consisting of MIDI encodings of only the first movements of the string quartets.

Table 3 shows that our best classifier reaches state-of-the-art performance and that there is no significant dif-

Method	Accuracy
Proposed best classifier	**80.4**
Van Kranenburg and Backer (2004) [24]	79.4
Herlands et al. (2014) [10]*	80.0
Hillewaere et al. (2010) [12]*	75.4
Hontanilla et al. (2013) [13]*	74.7

Table 3. Classification accuracies achieved by previous computational approaches on the Haydn/Mozart discrimination task. * indicates that a different dataset was used from that used in the experiments reported here.

ference from the results obtained by van Kranenburg and Backer at 5% significance level (Wilcoxon rank sum $= 11449$, $p = 0.8661$, $n = 107$). Compared to previous approaches [10,12,13,24], our method is more general in that it does not need hard-coded musical style markers for each dataset as in [24], nor does it require global musical feature sets as in [12], nor does it depend on the music having been parsed into separate parts or voices as in [10, 12, 13].

5. CONCLUSION

We have shown that string quartets by Haydn and Mozart can be discriminated by representing pieces of music as 2-D images of their pitch–time structure and then using convolutional models to operate on these images for classification. Our approach based on classifying pitch–time representations of music does not require parsing of the music into separate voices, or extraction of any other predefined features prior to processing. It addresses musical texture of 2-D pitch–time representations in a more general form. We have shown that filtering significantly improves recognition and that the method proves robust to encoding, transposition and amount of information. Our best single classifier reaches state-of-the-art performance in leave-one-out cross validation on the task of discriminating between string quartet movements by Haydn and Mozart.

With the proposed method, it is possible to generate a wide variety of classifiers. In preliminary experiments, we have seen that diverse configurations of classifiers (i.e. different filter types, orientations, centering, etc.) seem to provide complementary information which could be potentially used to build ensembles of classifiers improving classification further. Besides, we have observed that the method can be applied to synthetic audio files and audio recordings. In this case, audio files are sampled to spectrograms instead of piano-rolls, and then follow the method's chain of transformations, filtering and classification. We are optimistic that our proposed method can perform similarly on symbolic and audio data, and might be used successfully for other style discrimination tasks such as genre, period, origin, or performer recognition.

6. ACKNOWLEDGMENTS

The work for this paper carried out by G. Velarde, C. Cancino Chacón, D. Meredith, and M. Grachten was done as part of the EC-funded collaborative project, "Learning to Create" (Lrn2Cre8). The project Lrn2Cre8 acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET grant number 610859. G. Velarde is also supported by a PhD fellowship from the Department of Architecture, Design and Media Technology, Aalborg University. The authors would like to thank Peter van Kranenburg for sharing with us the string quartets dataset and results that allowed as statistical tests, William Herlands and Yoel Greenberg for supporting the unsuccessful attempt to reconstruct the dataset used in their research, Jordi Gonzalez for comments and suggestions on an early draft of this paper, and the anonymous reviewers for their detailed insight on this work.

7. REFERENCES

[1] J-P Antoine, Pierre Carrette, R Murenzi, and Bernard Piette. Image analysis with two-dimensional continuous wavelet transform. *Signal Processing*, 31(3):241–272, 1993.

[2] Yoshua Bengio, Aaron Courville, and Pierre Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013.

[3] Albert S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. MIT Press, Cambridge, MA., 1990.

[4] Deng Cai, Xiaofei He, Yuxiao Hu, Jiawei Han, and T. Huang. Learning a spatially smooth subspace for face recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–7, June 2007.

[5] Yandre MG Costa, LS Oliveira, Alessandro L Koerich, Fabien Gouyon, and JG Martins. Music genre classification using lbp textural features. *Signal Processing*, 92(11):2723–2737, 2012.

[6] John G Daugman. Two-dimensional spectral analysis of cortical receptive field profiles. *Vision Research*, 20(10):847–856, 1980.

[7] Diana Deutsch. Grouping mechanisms in music. In Diana Deutsch, editor, *The Psychology of Music*, pages 299–348. Academic Press, San Diego, 2nd edition, 1999.

[8] Diana Deutsch. *Psychology of Music*. Academic Press, San Diego, 3rd edition, 2013.

[9] Marc O Ernst and Heinrich H Bülthoff. Merging the senses into a robust percept. *Trends in Cognitive Sciences*, 8(4):162–169, 2004.

[10] William Herlands, Ricky Der, Yoel Greenberg, and Simon Levin. A machine learning approach to musically meaningful homogeneous style classification. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*, pages 276–282, 2014.

[11] Souta Hidaka, Wataru Teramoto, Yoichi Sugita, Yuko Manaka, Shuichi Sakamoto, Yôiti Suzuki, and Melissa Coleman. Auditory motion information drives visual motion perception. *PLoS One*, 6(3):e17499, 2011.

[12] Ruben Hillewaere, Bernard Manderick, and Darrell Conklin. String quartet classification with monophonic models. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 537–542, Utrecht, The Netherlands, 2010.

[13] María Hontanilla, Carlos Pérez-Sancho, and Jose M Iñesta. Modeling musical style with language models for composer recognition. In *Pattern Recognition and Image Analysis*, pages 740–748. Springer, 2013.

[14] Judson P Jones and Larry A Palmer. An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, 58(6):1233–1258, 1987.

[15] Kendrick N Kay, Thomas Naselaris, Ryan J Prenger, and Jack L Gallant. Identifying natural images from human brain activity. *Nature*, 452(7185):352–355, 2008.

[16] Fred Lerdahl and Ray S. Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, Cambridge, MA., 1983.

[17] S Marčelja. Mathematical description of the responses of simple cortical cells. *Journal of Neuophysiology*, 70(11):1297–1300, 1980.

[18] David Meredith. The *ps13* pitch spelling algorithm. *Journal of New Music Research*, 35(2):121–159, 2006.

[19] John C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods - Support Vector Learning*. MIT Press, January 1998.

[20] Craig Sapp and Yi-Wen Liu. The Haydn/Mozart String Quartet Quiz, 2015. `http://qq.themefinder.org` (Accessed 26 December 2015).

[21] Daniele Schön and Mireille Besson. Visually induced auditory expectancy in music reading: a behavioral and electrophysiological study. *Journal of Cognitive Neuroscience*, 17(4):694–705, 2005.

[22] Bob L Sturm. A simple method to determine if a music information retrieval system is a horse. *IEEE Transactions on Multimedia*, 16(6):1636–1644, 2014.

[23] Devis Tuia, Michele Volpi, Mauro Dalla Mura, Alain Rakotomamonjy, and Remi Flamary. Automatic feature learning for spatio-spectral image classification with sparse svm. *Geoscience and Remote Sensing, IEEE Transactions on*, 52(10):6062–6074, 2014.

[24] Peter Van Kranenburg and Eric Backer. Musical style recognition—a quantitative approach. In *Proceedings of the Conference on Interdisciplinary Musicology (CIM)*, pages 106–107, 2004.

[25] Gissel Velarde, Tillman Weyde, and David Meredith. An approach to melodic segmentation and classification based on filtering with the haar-wavelet. *Journal of New Music Research*, 42(4):325–345, 2013.

[26] Richard H Walthew. String quartets. *Proceedings of the Musical Association*, pages 145–162, 1915.

[27] Martin Litchfield West. The babylonian musical notation and the hurrian melodic texts. *Music & Letters*, pages 161–179, 1994.

[28] Ming-Ju Wu, Zhi-Sheng Chen, Jyh-Shing Roger Jang, Jia-Min Ren, Yi-Hsung Li, and Chun-Hung Lu. Combining visual and acoustic features for music genre classification. In *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*, volume 2, pages 124–129. IEEE, 2011.

CONVERSATIONS WITH EXPERT USERS IN MUSIC RETRIEVAL AND RESEARCH CHALLENGES FOR CREATIVE MIR

Kristina Andersen
Studio for Electro Instrumental Music (STEIM)
Amsterdam, the Netherlands
kristina@steim.nl

Peter Knees
Dept. of Computational Perception
Johannes Kepler University Linz, Austria
peter.knees@jku.at

ABSTRACT

Sample retrieval remains a central problem in the creative process of making electronic dance music. This paper describes the findings from a series of interview sessions involving users working creatively with electronic music. We conducted in-depth interviews with expert users on location at the Red Bull Music Academies in 2014 and 2015. When asked about their wishes and expectations for future technological developments in interfaces, most participants mentioned very practical requirements of storing and retrieving files. A central aspect of the desired systems is the need to provide increased flow and unbroken periods of concentration and creativity.

From the interviews, it becomes clear that for Creative MIR, and in particular, for music interfaces for creative expression, traditional requirements and paradigms for music and audio retrieval differ to those from consumer-centered MIR tasks such as playlist generation and recommendation and that new paradigms need to be considered. Despite all technical aspects being controllable by the experts themselves, searching for sounds to use in composition remains a largely semantic process. From the outcomes of the interviews, we outline a series of possible conclusions and areas and pose two research challenges for future developments of sample retrieval interfaces in the creative domain.

1. MOTIVATION AND CONTEXT

Considerable effort has been put into analysing user behaviour in the context of music retrieval in the past two decades [35]. This includes studies on music information seeking behaviour [14,17], organisation strategies [15], usage of commercial listening services [36], the needs or motivations of particular users, such as kids [28], adolescents [34], or musicologists [29], and behaviour analysis for specific tasks, e.g., playlist and mix generation [13], or in specific settings, e.g., riding together in a car [16] or in music lessons in secondary schools [49].

© Kristina Andersen, Peter Knees. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Kristina Andersen, Peter Knees. "Conversations with Expert Users in Music Retrieval and Research Challenges for Creative MIR", 17th International Society for Music Information Retrieval Conference, 2016.

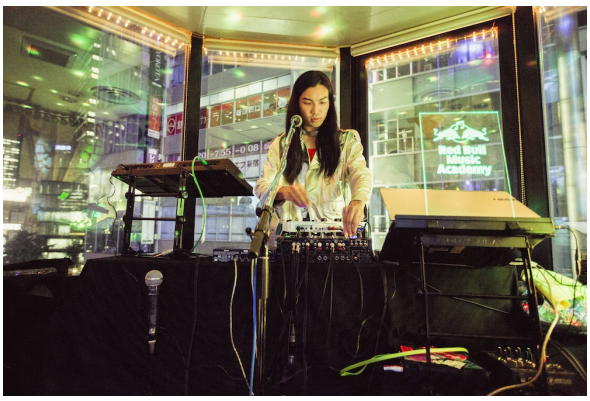


Figure 1. Live electronic music performance at the Red Bull Music Academy 2014

In this paper, we want to address music retrieval from the perspective of music producers, thus investigate the user behaviour of a group that deals with audio retrieval professionally on a daily basis, but has received comparatively less attention in MIR research so far—as have other questions from the area of Creative MIR [27].

The majority of today's electronic music is created from pre-recorded or live-generated sound material. This process often combines sound loops and samples with synthesized and processed elements using a so-called digital audio workstation (DAW), an electronic device or computer application for recording, editing and producing audio files. In these systems, Music Information Retrieval (MIR) methods, e.g., for content analysis, gain importance. In essence, future tools and applications need to be aware of the nature and content of the music material, in order to effectively support the musician in the creative process.

However, user studies on retrieval for musicians and producers are scarce. Cartwright et al. [6] investigate potential alternatives to existing audio production user interfaces in a study with 24 participants. In another example, Bainbridge et al. [4] explore and test a personal digital library environment for musicians, where based on a spatial paradigm musicians should be able to capture, annotate, and retrieve their ideas, e.g., using query-by-humming. In this paper, our approach is not to test an existing system, but to gain an understanding of the processes involved for music producers, who are used to working with existing music software suites.

This work is organized a follows. In section 2, we identify and briefly discuss existing MIR approaches in the context of music production. In section 3, we describe our motivation for engagement with expert users, our approach of conducting semi-structured interviews, and information on the interview background. The main part of the paper is presented in sections 4 and 5, where we highlight interesting outcomes of the interview sessions and distill central topics. Corresponding to this, we conclude this work by posing two research challenges for Creative MIR (section 6).

2. MIR RESEARCH IN MUSIC MAKING

Existing MIR research targeted at composition support and music production deals with browsing interfaces to facilitate access to large collections of potentially homogeneous material, such as drum samples [41]. Exploration of sample libraries, e.g., [8, 50], is often supported or driven by methods to automatically extract music loops from music files. Given the prevalent techniques of sampling and remixing in today's music production practise, such methods are useful to identify reusable materials and can lead to inspirations for new compositions [40, 51] and new interfaces for remixing [23] In terms of retrieval in the creative domain, and in contrast to consumer-based information systems, the query-by-example paradigm, implemented as query-by-humming or through other vocal inputs [5, 25, 31], is still an active research field.

Other MIR systems facilitate musical creation through automatic composition systems [10] or mosaicing systems that "reconstruct" the sound of a target piece by concatenating slices of other recordings [38, 45, 54]. This principle of concatenative synthesis can also be found in interactive systems for automatic accompaniment or improvisation such as the OMax system by Assayag et al. [3], Audio Oracle by Dubnov et al. [19], or Cont's Antescofo [11].

Other MIR systems emphasize the embodiment of creativity expression. For instance, Schnell et al. [44] propose a system that combines real-time audio processing, retrieval, and playback with gestural control for re-embodiment of recorded sound and music. The Wekinator [22] by Fiebrink is a real-time, interactive machine learning toolkit that can be used in the processes of music composition and performance, as well as to build new musical interfaces and has also shown to support the musical expression of people with disabilities [32].

3. WORKING WITH EXPERT USERS

Standards for user involvement in the field of Human Computer Interaction (HCI) have evolved from a traditional approach of metric user-testing of already designed systems, to understanding of users and their context through ethnographic methods and scenarios, towards an emerging focus on developing empathy with the user's experience of life. Wright and McCarthy state that "'knowing the user' in their lived and felt life involves understanding what it feels like to be that person, what their situation is like from their own perspective." [52]

This is especially important in the case of the creative expert users, who are not just looking to complete a series of tasks, but rather are engaging **with** the technology in order to express themselves **through** it. As such they can be seen to be not only using the technology, but rather collaborating with it as described by Tom Jenkinson (aka Squarepusher): "Through his work, a human operator brings as much about the machine to light as he does about himself ... The machine has begun to participate." [30]

This paper describes some of our efforts at building such an understanding. Eventually, our work will aim to create musical tools that provide new interfaces to the selection of sounds and musical data through music analysis algorithms. The underlying concern will be to not just improve existing user interfaces for the creation of electronic music through increases in efficiency, but facilitate increased flow and unbroken periods of concentration and creativity. To do so, we are engaging with expert users throughout the entire project, allowing them a strong peer position in the conceptualisation and evaluation of any ideas.

Our main users are the participants at the Red Bull Music Academy (RBMA), cf. fig. 1, an event held yearly with a carefully selected group of professional electronic dance music makers on the point of breaking though. [1]

Our sustained involvement with this group of expert users is key to our strategy of building detailed understandings of current forms of electronic music making, cf. [1]. We hope that this will allow us to go beyond user testing, and instead aim for a coherent impression of how an interface may benefit the real-life creative process of the users. To this end, we are committed to conducting interviews in a fashion that fits within the work-flow and interpersonal communication style of these music professionals, we ultimately aim to support creatively with the outcomes of the project. What we need to understand is: How do they organise their work, what are their needs, and ultimately what are their mental models of their music?

We conducted 33 in-depth interviews with expert users on location at the Red Bull Music Academy in Tokyo (2014) and Paris (2015). The aim of the 2014 sessions was to establish understandings of existing work practices among users, and the 2015 sessions were set up to investigate a number of emergent themes in more detail. Our interviews were executed in an open conversational structure, engaging the interviewees directly as peers, while aiming to support them to go beyond evaluation of current interfaces and into the imagination of new and unknown interfaces for their own creative practice.

The interviews were audio recorded and fully transcribed. Three independent HCI researchers analysed the

[1] `http://redbullmusicacademy.com`; From the web page: "The Red Bull Music Academy is a world-travelling series of music workshops and festivals *[in which]* selected participants – producers, vocalists, DJs, instrumentalists and all-round musical mavericks from around the world – come together in a different city each year. For two weeks, each group will hear lectures by musical luminaries, work together on tracks and perform in the city's best clubs and music halls."

transcripts for content-rich quotes: short sentences or paragraphs describing a particular idea or concern. The keywords from these quotes were extracted and used for labelling the quote, e.g., "search", "finding", "colour", or "pace". Following this, keywords were manually clustered into bigger concepts or themes. From the material collected, we identified the themes of *Search*, *Categories*, *Visualisation*, *Colour*, *Organisation*, *Assistance*, *Workflow*, *Connections*, *Correction*, *Suggestions*, *Obstructions*, *Deliberate error*, *Tweaks*, *Interfaces*, and *Live*. The material we address in this paper belongs to the themes of *Search*, *Categories*, *Colour*, *Visualisation*, *Organisation*, *Suggestions*, and *Obstructions*.

4. INTERVIEW QUOTES AND FINDINGS

When asked about their wishes and expectations for future technological developments, most participants mentioned very practical requirements for storing and retrieving files. Sounds are usually stored as short audio files (samples) or presets, which can be loaded into playback devices in composition software.

> *"Because we usually have to browse really huge libraries [...] that most of the time are not really well organized."* (TOK003)

> *"If you have like a sample library with 500,000 different chords it can take a while to actually find one because there are so many possibilities."* (TOK015)

> *"Like, two hundred gigabytes of [samples]. I try to keep some kind of organisation."* (TOK006)

> *"I easily get lost... I always have to scroll back and forth and it ruins the flow when you're playing"* (PA011)

> *"...what takes me really long time is organising my music library for DJing. [...] Yes, it could be something like Google image search for example. You input a batch of noise, and you wait for it to return a sound."* (TOK011)

Even from this small selection of statements it becomes clear that organisation of audio libraries, indexing, and efficient retrieval plays a central role in the practice of music creators and producers. However, in the search tools provided by existing DAWs, this aspect seems addressed insufficiently. When asked directly: "How do you find what you are looking for?" answers indicated a number of personal strategies that either worked with, or sometimes in opposition to, the existing software design.

> *"You just click randomly and just scrolling, it takes for ever!"* (TOK009)

> *"Sometimes, when you don't know what you are looking for, and you're just going randomly through your samples, that might be helpful, but most of the time I have something in mind that I am looking for, and I am just going through all these sound files, and I am just waiting for the sound which I had in mind to suddenly appear. Or what comes the closest to what I had in mind. So I think that most of the time, I know what I am looking for, and then it is just a matter of time before I find it."* (TOK002)

> *"Part of making music is about being lost a little bit and accidentally stumbling upon stuff that you didn't think would work."* (TOK007)

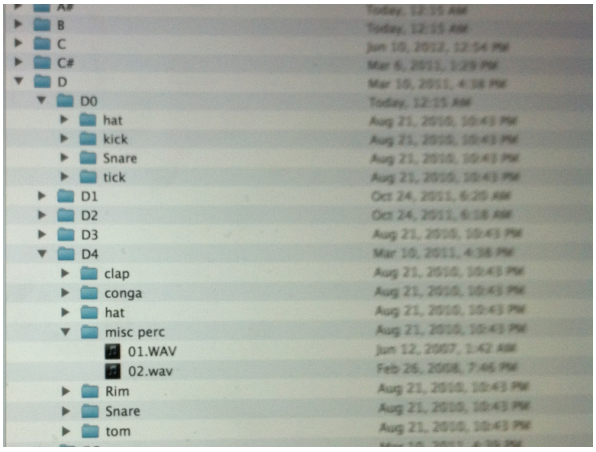


Figure 2. User sample file collection, photographed from laptop screen of expert music producer at RBMA 2014.

This highlights a key element of much creative work, the element of the accidental, sometimes caused by the positive and negative effects of malfunctioning of sound-editing software. Our *finding 1* is that serendipity is highly important to support creative work and that when existing software is not providing this desired functionality, workarounds will be created.

Often, users are constructing their own personal systems for searching, sometimes working with the structures available to them, but often developing idiosyncratic and personal strategies of misuse or even randomness. We also see users painstakingly creating hierarchical categorisation schemes manually in order to stay in control of their own sound collections as seen in Figure 2.

Searching for sounds to use in composition remains a broadly semantic process, where the user has a certain structure of meaning in mind when querying the collection. Current solutions in this direction rely on databases with sets of meta-data (tags) for the available sounds. However, all available solutions that come with pre-tagged sounds use a static semantic structure, which cannot adapt to the users individual understanding of the sounds. This is especially problematic when the user has a specific target sound in mind, but does not know how this would be described in the pre-tagged semantic structure of the database. In short, as our *finding 2* we see that our users have mental images of sound that they translate to verbal expressions.

> *"So it would be really useful to for example have some kind of sorting system for drums, for example, where I could for example choose: 'bass drum', and here it is: 'bass' and 'bright', and I would like it to have maybe bass drum 'round' and 'dry', and you can choose both, the more I choose, of course, the less results I will have [...] So it is filtering it down, that is really helpful, if it works well of course."* (TOK002)

> *"It would be even more useful to be able to search for a particular snare, but I can't really imagine how, I need something short, low in pitch, dark or bright in tone and then it finds it..."* (TOK003)

> *"There are a lot of adjectives for sound, but for me, if you want a 'bright' sound for example it actually means a sound with a lot of treble, if you say you want a 'warm' sound, you put a round bass, well, round is another adjective."* (TOK009)

We also see that the semantic descriptions used in these descriptions are very individually connoted and often stemming from non-auditory domains, such as haptic, or most prominently, the visual domain (e.g., round, dark, bright). Thus, the mental images expressed verbally are often actually rooted in another domain.

In fact, one solution to the problem of indexing suggested by our participants is to organize sounds by sources, by projects in which they have been used, or to colour-code them.

> *"If I have hundreds of tracks, I have to colour code everything, and name properly everything. It's a kind of system, and also kind of I feel the colours with the sounds, or maybe a rose, if kind of more orange, and brownish and maybe... I use that kind of colour coding."* (TOK006)

Finding 3 is that we see a need for semantic representations of sounds, but it's not only a matter of just tags and words, but rather an ability to stay much closer to the vocabulary and mental representations of sound of each user.

Additionally, the question arises whether the interviewees really want a direct representation of their mental map in the data structure, or if indeed they rather expect something more akin to a machine collaborator, that could come up with its own recommendations and suggest a structure based on the personal requirements of the individual user.

> *"I'd like it to do the opposite actually, because the point is to get a possibility, I mean I can already make it sound like me, it's easy."* (TOK001)

> *"What I would probably rather want it would do is make it complex in a way that I appreciate, like I would be more interested in something that made me sound like the opposite of me, but within the boundaries of what I like, because that's useful. Cause I can't do that on my own, it's like having a band mate basically."* (TOK007)

Again we see the computer as a potential collaborator, one that might even be granted some level of autonomy:

> *"Yeah, yeah, well I like to be completely in charge myself, but I like to... I don't like other humans sitting the chair, but I would like the machine to sit in the chair, as long as I get to decide when it gets out."* (TOK014)

Finding 4 is that instead of computer-generated and similarity-based recommendations, desired features are surprise, opposition, individuality, and control over the process (with the possibility to give up control when needed).

5. INTERVIEW CONCLUSIONS

From the interviews outlined in the previous section, we see that central concerns in everyday work in music production are core topics of MIR: indexing, retrieval, browsing, recommendation, and intuitive (visual) interfaces. More than in music retrieval systems built for consumer or entertainment needs, the expert user in a music production environment will actually evaluate a large part of the returned items to find the best—a process that is integral to some as a way to get inspired.

In order to facilitate this process and to enable creative work to make better use of MIR tools, we identified four crucial findings:

1. Surprise and serendipity in recommendation and retrieval are important to support creative work
2. Users have personal mental images of sound
3. There is a need for semantic representations of sounds for retrieval, which are not just tags and words but rather reflect those mental images (which can be visual or haptic)
4. Instead of "more of the same" recommendations, desired features are surprise, opposition, individuality, and control over the recommendation process

The desires for systems that respond to personal vocabularies and individual mental images of sound alongside the desire to have a controllable element of otherness and difference, constitute both a challenge and an opportunity. However, this also goes someway towards illustrating how individualized the creative expert user needs may turn out to be. While we can try to find common concerns, it is clear that no system will fit the requirements of all users. In the creative domain even more than in consumer-oriented

MIR, allowing personalisation of the systems is a central requirement, cf. [4]. This is reflected in the two following areas, which take the findings into a bigger context and outline two conceptual ideas for future Creative MIR research, namely "The Collaborative Machine," building upon findings 1 and 4 to go beyond the idea of a traditional recommender system, and "Synaesthetic Sound Retrieval," based upon findings 2 and 3, as an approach beyond tag-based "semantic retrieval".

5.1 The Collaborative Machine

The collaborative machine can be imagined as a virtual bandmate who assesses, critiques, takes over, and occasionally opposes.

It appears that in creative work as well as in the consumer world, successful imitation is not enough for the machine to be recognized as "intelligent" anymore. While this is a first and necessary step in creative and intelligent behavior, a machine requires more multi-faceted and complex behavior in order to be considered a useful advice-giver or even collaborator. However, no matter how well grounded or wise they can be, artificial knowledge and expert agent-based advice might be completely useless or, even worse, annoying and even odious. Aspects of such human behavior, as well as of surprise, opposition, and obstruction, should contribute to making the interaction with the machine more interesting and engaging.

Can we imagine an intelligent machine providing the user with creative obstructions in the place of helpful suggestions?

A creative obstruction is based on the artistic technique of "defamiliarisation" as defined by Shklovsky [47]—a basic artistic strategy central to both Surrealism and Dada. It is based on the idea that the act of experiencing something occurs inside the moment of perceiving it and that the further you confuse or otherwise prolong the moment of arriving at an understanding, the deeper or more detailed that understanding will be. This technique and the findings from the interviews can be directly translated into new requirements for recommendation engines in music making.

This need for opposition goes far beyond the commonly known and often addressed *needs for diversity, novelty, and serendipity* in recommendation system research, which has identified purely similarity-based recommendation as a shortcoming that leads to decreased user satisfaction and monotony [7, 48, 53]. This phenomenon spans multiple domains: from news articles [37] to photos [42] to movies [18]. One idea proposed to increase diversity is to subvert the basic idea of collaborative filtering systems of recommending what people with similar interests found interesting ("people with similar interests also like...") by recommending the opposite of what the least similar users (the k-furthest neighbors) want [43]. Indeed it could be shown that this technique allows to increase diversity among relevant suggestions.

In the context of experimental music creation, Collins has addressed the question of opposition in the Contrary Motion system [9] using a low-dimensional representation of rhythm. The system opposes a piano player's rhythm in real time by constructing a structure located in the space of actions "where the human is likely not to be" [9]. The hypothesis underlying the system is that being confronted with an oppositional music style can be stimulating for a musician. Experiments where the opposing structure is sonified using a different instrument have indeed shown that musicians start to experiment and play with the opposing agent. For future work, it would be interesting to see whether computer-system-created music (or a system that suggests fragments) will be accepted by experts or declined, cf. [39].

5.2 Synaesthetic Sound Retrieval

Multiple search avenues allow the user to use many different ways to describe the searched-for sound. This includes acoustic sketching, e.g., [5, 25, 31], as well as graphical representations.

In a number of quotes in section 4, sounds are described by shapes (round), brightness (bright, dark) and textures (soft, dry). While these might be regarded as unusual descriptors of sound, there is some evidence that many humans make to some degree use of *synaesthetic* connections between visual perceptions and sound. In the Creative MIR scenario, we make use of a weak definition of synaesthesia as cross-modal associations, cf. [20, 26], and, in the context of computer science, "the more general fact that digital technologies offer, if not a union of the senses, then something akin: the inter-translatability of media, the ability to render sound as image, and vice versa." [12]

Focusing on the visual domain, through the interviews, a number of ideas and notions came up in addition to the importance of brightness, shape, and texture for sound retrieval. More precisely, colour plays a central role: *"I see the music sometimes as more aesthetic and something that I can see more than something that I can hear"* (PA013), *"When I listen to music I see colours. [...] I remember colours."* (PA011), *"Different sounds to me have specific colours. ... [For finding files,] I don't have the actual ability to use images [now], so I just use colour."* (PA009).

Such a colour-coding, for instance, takes the role of "semantic tagging". The fact that a system needs time to learn a user's associations first, i.e., that it might not work perfectly out of the box but learn their associations over time (personalisation), is understood and accepted:

> *"You could imagine that your computer gets used to you, it learns what you mean by grainy, because it could be different from what that guy means by grainy."* (PA008)

The models learned from personal categories and (visual) tagging could then be applied on new collections for personal indexing.

For browsing sound, the idea of tapping into the visual domain is well-established. Most proposed sound browsing systems are based on 2D arrangements of sounds [8, 21, 41, 46]—even including personalised adaptation of the arrangement [24]. In these systems, the visual aspect is the

spatial arrangement of sounds, however, this does not reflect the mental models but rather requires the user to learn the mapping provided by the system. Grill and Flexer [26] get closer to a synaesthetic representation by visualizing perceptual qualities of sound textures through symbols on a grid. [2] To this end, they map bipolar qualities of sound that describe spectral and temporal aspects of sound to visual properties. The spectral qualities of pitch (high vs. low) and tonality (tonal vs. noisy) are mapped to brightness and hue, and saturation, respectively. The temporal (or structural) qualities of smoothness vs. coarseness, order vs. chaos, and homogeneity vs. heterogeneity are associated with the jaggedness of an element's outline, the regularity of elements on the grid, and a variation in colour parameters, respectively.

To the best of our knowledge, there is no system that matches a visual query representing a mental image of sound to a sound collection for retrieval. Developing such a system would, however, pose an interesting challenge.

6. POSED CHALLENGES

Results that enable the two conceptual ideas discussed above can not be trivially achieved. Therefore, to conclude this paper, we want to pose two topics as challenges for future work in Creative MIR to the wider community. Both of these topics should allow for alternative retrieval paradigms particularly relevant in creative work. As discussed before, they require high levels of personalisation in order to facilitate "semantic" retrieval.

Challenge 1: developing models for exploring dissimilarity in search

To arrive at an artificial collaborator capable of inspiring by opposing, the concept of opposition needs to be explored first, cf. [2]. Music similarity is a multi-dimensional concept and while proximity can be easily, "semantically" defined through minimizing distance measures, the concept of dissimilarity is by far more difficult to capture as it "spreads out" to different directions and dimensions of sound. Finding dissimilar sounding audio from a given query is therefore more challenging and requires individual user models of music perception as well as a solid understanding of usage context in order to derive an understanding of sounding "different".

Challenge 2: developing retrieval methods for visual queries

This challenge is to develop a software interface for sound search based on queries consisting of sketches of mental images, cf. [33]. A central requirement for such an interface is that it needs to be able to deal with different sound properties and different types of sounds, such as effects, samples, ambient, tonal, or textured recordings, and therefore comprise different simultaneous representational models for indexing. For instance, while tonal aspects might be best represented using symbolic music notation, noise sounds should be modeled primarily via their textural properties. It is expected that modeling and indexing will heavily draw from audio content processing and analysis methods—again—in order to cover a wide range of sound property dimensions.

We hope that these challenges will drive the discussion on Creative MIR and its applications in music production and help reflecting upon and advancing the field of music retrieval also beyond the specific area of study of this work.

[2] `http://grrrr.org/test/texvis/map.html`

7. ACKNOWLEDGEMENTS

This work is supported by the European Union's seventh Framework Programme FP7 / 2007–2013 for research, technological development and demonstration under grant agreement no. 610591 (GiantSteps). Figure 1 courtesy of Red Bull Music Academy.

8. REFERENCES

[1] K. Andersen and F. Grote. "GiantSteps: Semi-structured conversations with musicians". In *Proc CHI EA*, 2015.

[2] K. Andersen and P. Knees. "The dial: Exploring computational strangeness". In *Proc CHI EA*, 2016.

[3] G. Assayag, G. Bloch, M. Chemillier, A. Cont, and S. Dubnov. "OMAX Brothers: A dynamic topology of agents for improvisation learning". In *Proc ACM MM Workshop on Audio and Music Computing for Multimedia*, 2006.

[4] D. Bainbridge, B.J. Novak, and S.J. Cunningham. "A user-centered design of a personal digital library for music exploration". In *Proc JCDL*, 2010.

[5] M. Cartwright and B. Pardo. "Synthassist: Querying an audio synthesizer by vocal imitation". In *Proc NIME*, 2014.

[6] M. Cartwright, B. Pardo, and J. Reiss. "Mixploration: Rethinking the audio mixer interface". In *Proc IUI*, 2014.

[7] Ò. Celma and P. Herrera. "A new approach to evaluating novel recommendations". In *Proc RecSys*, 2008.

[8] G. Coleman. "Mused: Navigating the personal sample library". In *Proc ICMC*, 2007.

[9] N. Collins. "Contrary motion : An oppositional interactive music system". In *Proc NIME*, 2010.

[10] N. Collins. "Automatic composition of electroacoustic art music utilizing machine listening". *CMJ*, 36(3):8–23, 2012.

[11] A. Cont. ""Antescofo: Anticipatory synchronization and control of interactive parameters in computer music". In *Proc ICMC*, 2008.

[12] C. Cox. "Lost in translation: Sound in the discourse of synaesthesia". *Artforum International*, 44(2):236–241, 2005.

[13] S.J. Cunningham, D. Bainbridge, and A. Falconer. "'More of an art than a science': Supporting the creation of playlists and mixes". In *Proc ISMIR*, 2006.

[14] S.J. Cunningham, J.S. Downie, and D. Bainbridge. "'The pain, the pain': Modelling music information behavior and the songs we hate". In *Proc ISMIR*, 2005.

[15] S.J. Cunningham, M. Jones, and S. Jones. "Organizing digital music for use: An examination of personal music collections". In *Proc ISMIR*, 2004.

[16] S.J. Cunningham, D.M. Nichols, D. Bainbridge, and H. Ali. "Social music in cars". In *Proc ISMIR*, 2014.

[17] S.J. Cunningham, N. Reeves, and M. Britland. “An ethnographic study of music information seeking: Implications for the design of a music digital library”. In *Proc JCDL*, 2003.

[18] T. Di Noia, V.C. Ostuni, J. Rosati, P. Tomeo, and E. Di Sciascio. “An analysis of users’ propensity toward diversity in recommendations”. In *Proc RecSys*, 2014.

[19] S. Dubnov, G. Assayag, and A. Cont. “Audio oracle: A new algorithm for fast learning of audio structures”. In *Proc ICMC*, 2007.

[20] K.K. Evans and A. Treisman. “Natural cross-modal mappings between visual and auditory features”. *JOV*, 10(1):6, 2010.

[21] M. Fernström and E. Brazil. “Sonic browsing: An auditory tool for multimedia asset management”. In *Proc ICAD*, 2001.

[22] R. Fiebrink. *Real-time Human Interaction with Supervised Learning Algorithms for Music Composition and Performance*. PhD thesis, Princeton University, NJ, USA, Jan 2011.

[23] J. Forsyth, A. Glennon, and J.P. Bello. “Random access remixing on the iPad”. In *Proc NIME*, 2011.

[24] O. Fried, Z. Jin, R. Oda, and A. Finkelstein. “AudioQuilt: 2D arrangements of audio samples using metric learning and kernelized sorting”. In *Proc NIME*, 2014.

[25] O. Gillet and G. Richard. “Drum loops retrieval from spoken queries”. *JIIS*, 24(2-3):159–177, 2005.

[26] T. Grill and A. Flexer. “Visualization of perceptual qualities in textural sounds”. In *Proc ICMC*, 2012.

[27] E.J. Humphrey, D. Turnbull, and T. Collins. “A brief review of Creative MIR”. In *ISMIR Late-Breaking News and Demos*, 2013.

[28] M. Hutter and S.J. Cunningham. “Towards the design of a kids’ music organizer”. In *Proc CHINZ*, 2008.

[29] C. Inskip and F. Wiering. “In their own words: Using text analysis to identify musicologists’ attitudes towards technology”. In *Proc ISMIR’15*, 2015.

[30] T. Jenkinson. “Collaborating with machines”. *Flux Magazin (www.fluxmagazine.com)*, March 2004.

[31] A. Kapur, M. Benning, and G. Tzanetakis. “Query-by-beatboxing: Music retrieval for the DJ”. In *Proc ISMIR*, 2004.

[32] S. Katan, M. Grierson, and R. Fiebrink. “Using interactive machine learning to support interface development through workshops with disabled people”. In *Proc CHI*, 2015.

[33] P. Knees and K. Andersen. “Searching for audio by sketching mental images of sound – A brave new idea for audio retrieval in creative music production”. In *Proc ICMR*, 2016.

[34] A. Laplante and J.S. Downie. “Everyday life music information-seeking behaviour of young adults”. In *Proc ISMIR*, 2006.

[35] J.H. Lee and S.J. Cunningham. “Toward an understanding of the history and impact of user studies in music information retrieval”. *JIIS*, 41(3):499–521, 2013.

[36] J.H. Lee and R. Price. “Understanding users of commercial music services through personas: Design implications”. In *Proc ISMIR*, 2015.

[37] L. Li, D. Wang, T. Li, D. Knox, and B. Padmanabhan. “Scene: A scalable two-stage personalized news recommendation system”. In *Proc SIGIR*, 2011.

[38] E. Maestre, R. Ramírez, S. Kersten, and X. Serra. “Expressive concatenative synthesis by reusing samples from real performance recordings”. *CMJ*, 33(4):23–42, Dec 2009.

[39] D.C. Moffat and M. Kelly. “An investigation into people’s bias against computational creativity in music composition”. *Assessment*, 13(11), 2006.

[40] B.S. Ong and S. Streich. “Music loop extraction from digital audio signals”. In *Proc ICME*, 2008.

[41] E. Pampalk, P. Hlavac, and P. Herrera. “Hierarchical organization and visualization of drum sample libraries”. In *Proc DAFx*, 2004.

[42] A.-L. Radu, B. Ionescu, M. Menéndez, J. Stöttinger, F .Giunchiglia, and A. De Angeli. “A hybrid machine-crowd approach to photo retrieval result diversification”. *Proc MMM* 2014.

[43] A. Said, B. Fields, B.J. Jain, and S. Albayrak. “User-centric evaluation of a k-furthest neighbor collaborative filtering recommender algorithm”. In *Proc CSCW*, 2013.

[44] N. Schnell, F. Bevilacqua, N. Rasamimanana, J. Blois, F. Guédy, and E. Fléty. “Playing the ‘MO’ – Gestural Control and Re-Embodiment of Recorded Sound and Music”. In *Proc NIME*, 2011.

[45] D. Schwarz. “Current research in concatenative sound synthesis”. In *Proc ICMC*, 2005.

[46] D. Schwarz and N. Schnell. “Sound search by content-based navigation in large databases”. In *Proc SMC*, 2009.

[47] V. Shklovsky. *Art as Technique. In: Russian Formalist Criticism: Four Essays, 1965. Trans. L.T. Lemon and M.J. Reis.* University of Nebraska Press, Lincoln, USA, 1917.

[48] B. Smyth and P. McClave. “Similarity vs. diversity”. In *Proc ICCBR*, 2001.

[49] D. Stowell and S. Dixon. “MIR in school? Lessons from ethnographic observation of secondary school music classes”. In *Proc ISMIR*, 2011.

[50] S. Streich and B.S. Ong. “A music loop explorer system”. In *Proc ICMC*, 2008.

[51] H.L. Tan, Y. Zhu, S. Rahardja, and L. Chaisorn. “Rhythm analysis for personal and social music applications using drum loop patterns”. In *Proc ICME*, 2009.

[52] P. Wright and J. McCarthy. “Empathy and experience in HCI”. In *Proc CHI*, 2008.

[53] Y.C. Zhang, D. Ó Séaghdha, D. Quercia, and T. Jambor. “Auralist: Introducing serendipity into music recommendation”. In *Proc WSDM*, 2012.

[54] A. Zils and F. Pachet. “Musical mosaicing”. In *Proc DAFx*, 2001.

DOWNBEAT TRACKING USING BEAT-SYNCHRONOUS FEATURES AND RECURRENT NEURAL NETWORKS

Florian Krebs, Sebastian Böck, Matthias Dorfer, and Gerhard Widmer
Department of Computational Perception
Johannes Kepler University Linz, Austria
Florian.Krebs@jku.at

ABSTRACT

In this paper, we propose a system that extracts the downbeat times from a beat-synchronous audio feature stream of a music piece. Two recurrent neural networks are used as a front-end: the first one models rhythmic content on multiple frequency bands, while the second one models the harmonic content of the signal. The output activations are then combined and fed into a dynamic Bayesian network which acts as a rhythmical language model. We show on seven commonly used datasets of Western music that the system is able to achieve state-of-the-art results.

1. INTRODUCTION

The automatic analysis of the metrical structure in an audio piece is a long-standing, ongoing endeavour. A good underlying meter analysis system is fundamental for various tasks like automatic music segmentation, transcription, or applications such as automatic slicing in digital audio workstations.

The meter in music is organised in a hierarchy of pulses with integer related frequencies. In this work, we concentrate on one of the higher levels of the metrical hierarchy, the *measure* level. The first beat of a musical measure is called a *downbeat*, and this is typically where harmonic changes occur or specific rhythmic pattern begin [23].

The first system that automatically detected beats and downbeats was proposed by Goto and Muraoka [15]. It modelled three metrical levels, including the measure level by finding chord changes. Their system, built upon hand-designed features and rules, was reported to successfully track downbeats in $4/4$ music with drums. Since then, much has changed in the meter tracking literature. A general trend is to go from hand-crafted features and rules to automatically learned ones. In this line, rhythmic patterns are learned from data and used as observation model in probabilistic state-space models [23, 24, 28]. Support Vector Machines (SVMs) were first applied to downbeat tracking in a semi-automatic setting [22] and later used in a fully automatic system that operated on several beat-synchronous hand-crafted features [12]. The latter system was later refined by using convolutional neural networks (ConvNets) instead of SVMs and a new set of features [10, 11] and is the current state-of-the-art in downbeat tracking on Western music.

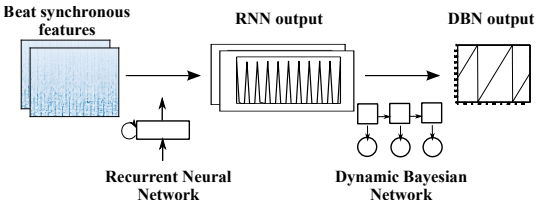


Figure 1: Model overview

Recurrent neural networks (RNNs) are Neural Networks adapted to sequential data and therefore are the natural choice for sequence analysis tasks. In fact, they have shown success in various tasks such as speech recognition [19], handwriting recognition [17] or beat tracking [2]. In this work, we would like to explore the application of RNNs to the downbeat tracking problem. We describe a system that detects downbeats from a beat-synchronous input feature sequence, analyse the performance of two different input features, and discuss shortcomings of the proposed model. We report state-of-the-art performance on seven datasets.

The paper is organised as follows: In Section 2 we describe the proposed RNN-based downbeat tracking system, in Section 3 we explain the experimental set-up of our evaluation and present and discuss the results in Section 4.

2. METHOD

An overview of the system is shown in Fig. 1. Two beat-synchronised feature streams (Section 2.1) are fed into two parallel RNNs (Section 2.2) to obtain a downbeat activation function which indicates the probability whether a beat is a downbeat. Finally, the activation function is decoded into a sequence of downbeat times by a dynamic Bayesian network (DBN) (Section 2.3).

© Florian Krebs, Sebastian Böck, Matthias Dorfer, Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Florian Krebs, Sebastian Böck, Matthias Dorfer, Gerhard Widmer. "DOWNBEAT TRACKING USING BEAT-SYNCHRONOUS FEATURES AND RECURRENT NEURAL NETWORKS", 17th International Society for Music Information Retrieval Conference, 2016.

2.1 Feature extraction

In this work we assume that the beat times of an audio signal are known, by using either hand-annotated or automatically generated labels. We believe that the segmentation into beats makes it much more easy for the subsequent stage to detect downbeats because it does not have to deal with tempo or expressive timing on one hand and it greatly reduces the computational complexity by both reducing the sequence length of an excerpt and the search space. Beat-synchronous features have successfully been used before for downbeat tracking [5, 10, 27]. Here, we use two features: A spectral flux with logarithmic frequency spacing to represent percussive content (*percussive feature*) and a chroma feature to represent the harmonic progressions throughout a song (*harmonic feature*).

2.1.1 Percussive feature

As a percussive feature, we compute a multi-band spectral flux: First, we compute the magnitude spectrogram by applying the Short-time Fourier Transform (STFT) with a Hann window, hopsize of 10ms, and a frame length of 2048 samples, as shown in Fig. 2a. Then, we apply a logarithmic filter bank with 6 bands per octave, covering the frequency range from 30 to 17 000 Hz, resulting in 45 bins in total. We compress the magnitude by applying the logarithm and finally compute for each frame the difference between the current and the previous frame. The feature sequence is then beat-synchronised by only keeping the mean value per frequency bin in a window of length Δ_b/n_p, where Δ_b is the beat period and $n_p = 4$ is the number of beat subdivisions, centred around the beginning of a beat subdivision. An example of the percussive feature is shown in Fig. 2b.

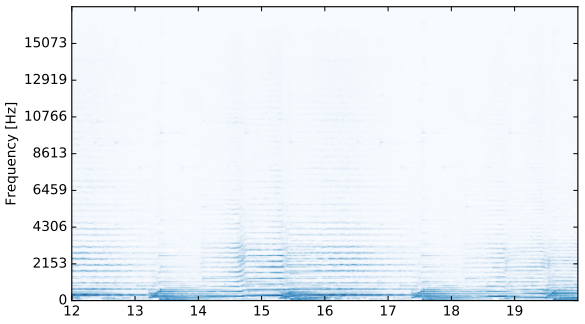
2.1.2 Harmonic feature

As harmonic feature, we use the *CLP* chroma feature [26] with a frame rate of 100 frames per second. We synchronise the features to the beat by computing the mean over a window of length Δ_b/n_h, yielding $n_h = 2$ feature values per beat interval. We found that for the harmonic feature the resolution can be lower than for the percussive feature, as for chord changes the exact timing is less critical. An example of the harmonic feature is shown in Fig. 2d.

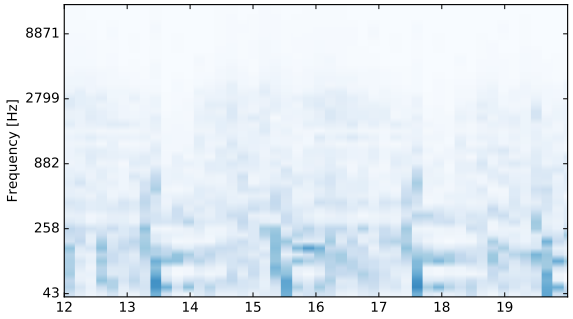
2.2 Recurrent Neural Network

RNNs are the natural choice for sequence modelling tasks but often difficult to train due to the exploding and vanishing gradient problems. In order to overcome these problems when dealing with long sequences, Long-Short-Term memory (LSTM) networks were proposed [20]. Later, [4] proposed a simplified version of the LSTMs named Gated Recurrent Units (GRUs), which were shown to perform comparable to the traditional LSTM in a variety of tasks and have less parameters to train. Therefore, we will use GRUs in this paper.

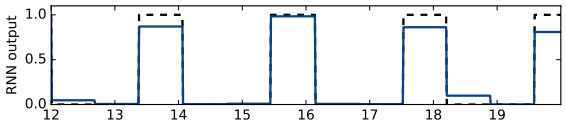
The time unit modelled by the RNNs is the *beat period*, and all feature values that fall into one beat are condensed into one vector. E.g., using the percussive feature with 45



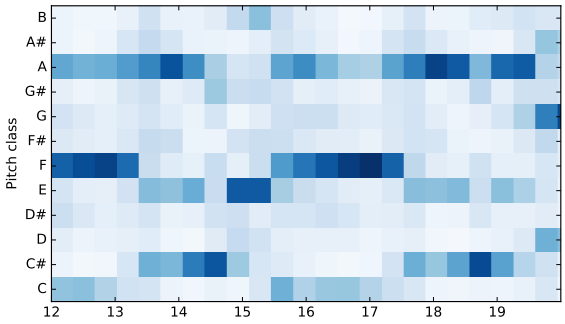
(a) Spectrogram



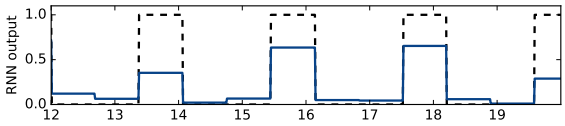
(b) Beat-synchronous percussive feature



(c) Activation and target of the rhythmic network



(d) Beat-synchronous chroma feature



(e) Activation and target of the harmonic network

Figure 2: Visualisation of the two feature streams and their corresponding network output of an 8-second excerpt of the song *Media-105701* (Ballroom dataset). The dashed line in (c) and (e) represents the target (downbeat) sequence, the solid line the networks' activations. The x-axis shows time in seconds. The time resolution is one fourth of the beat period in (b), and half a beat period in (d).

frequency bins and a resolution of $n_p = 4$ beat subdivisions yields an input dimension of $45 \times 4 = 180$ for the rhythmic RNN. In comparison to an RNN that models *subdivisions* of the beat period as underlying time unit, this vectorisation of the temporal context provided an important speed-up of the network training due to the reduced sequence length, while maintaining the same level of performance.

In preliminary tests, we investigated possible architectures for our task and compared their performances on the validation set (see Section 3.3). We made the following discoveries: First, adding bidirectional connections to the models was found to greatly improve the performance. Second, the use of LSTMs/GRUs further improved the performance compared to the standard RNN. Third, using more than two layers did not further improve the performance.

We therefore chose to use a two layer bidirectional network with GRU units and standard tanh non-linearity. Each hidden layer has 25 units. The output layer is a dense layer with one unit and a sigmoid non-linearity. Due to the different number of input units the rhythmic model has approximately 44k, and the harmonic model approximately 19k parameters.

The activations of both the rhythmic and harmonic model are finally averaged to yield the input activation for the subsequent DBN stage.

2.3 Dynamic Bayesian Network

The language model incorporates musical prior knowledge into the system. In our case it implements the following assumptions:

1. Beats are organised into bars, which consist of a constant number of beats.

2. The time signature of a piece determines the number of beats per bar.

3. Time signature changes are rare within a piece.

The DBN stage is similar to the one used in [10], with three differences: First, we model beats as states instead of tatums. Second, as our data mainly contains $3/4$ and $4/4$ time signatures, we only model these two. Third, we force the state sequence to always transverse a whole bar from left to right, i.e., transitions from beat 2 to beat 1 are not allowed. In the following we give a short review of the DBN stage.

A state $s(b, r)$ in the DBN state space is determined by two hidden state variables: the beat counter b and the time signature r. The beat counter counts the beats within a bar $b \in \{1..N_r\}$ where N_r is the number of beats in time signature r. E.g., $r \in \{3, 4\}$ for the case where a 3/4 and a 4/4 time signature are modelled. The state transition probabilities can then be decomposed using

$$P(s_k|s_{k-1}) = P(b_k|b_{k-1}, r_{k-1}) \times P(r_k|r_{k-1}, b_k, b_{k-1}) \tag{1}$$

where

$$P(b_k|b_{k-1}, r_{k-1}) = \begin{cases} 1 & \text{if } b_k = (b_{k-1} \bmod r_{k-1}) + 1 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Eq. 2 ensures that the beat counter can only move steadily from left to right. Time signature changes are only allowed to happen at the beginning of a bar ($(b_k < b_{k-1})$), as implemented by

$$\begin{aligned} &\text{if} \quad (b_k < b_{k-1}) \\ &\qquad P(r_k|r_{k-1}, b_k, b_{k-1}) = \begin{cases} 1 - p_r & \text{if } (r_k = r_{k-1}) \\ p_r / R & \text{if } (r_k \neq r_{k-1}) \end{cases} \\ &\text{else} \\ &\qquad P(r_k|r_{k-1}, b_k, b_{k-1}) = 0 \end{aligned} \tag{3}$$

where p_r is the probability of a time signature change. We learned p_r on the validation set and found $p_r = 10^{-7}$ to be an overall good value, which makes time signature changes improbable but possible. However, the exact choice of this parameter is not critical, but it should be greater than zero as mentioned in Section 4.5.

As the sigmoid of the output layer of the RNN yields a value between 0 and 1, we can interpret its output as the probability that a specific beat is a downbeat and use it as observation likelihood for the DBN. As the RNN outputs a posterior probability $P(s|\text{features})$, we need to scale it by a factor $\lambda(s)$ which is proportional to $1/P(s)$ in order to obtain

$$P(\text{features}|s) \propto P(s|\text{features})/P(s), \tag{4}$$

which is needed by the observation model of the DBN. Experiments have shown that a value of $\lambda(s(b = 1, r)) = 100$ for downbeat states and $\lambda(s(b > 1, r)) = 1$ for the other states performed best on our validation set, and will be used in this paper.

Finally, we use a uniform initial distribution over the states and decode the most probably state sequence with the Viterbi algorithm.

3. EXPERIMENTS

3.1 Data

In this work, we restrict the data to Western music only and leave the evaluation of Non-Western music for future work. The following datasets are used:

Ballroom [16, 24]: This dataset consists of 685 unique 30 second-long excerpts of Ballroom dance music. The total length is 5h 57m.

Beatles [6]: This dataset consists of 180 songs of the Beatles. The total length is 8h 09m.

Hainsworth [18]: This dataset consists of 222 excerpts, covering various genres. The total length is 3h 19m.

RWC Pop [14]: This dataset consists of 100 American and Japanese Pop songs. The total length is 6h 47m.

Robbie Williams [13]: 65 full songs of Robbie Williams. The total length is 4h 31m

Rock [7]: This dataset consists of 200 songs of the Rolling Stone magazine's list of the "500 Greatest Songs of All Time". The total length is 12h 53m.

System	Ballroom	Beatles	Hainsworth	RWC pop	Robbie Williams	Klapuri	Rock	Mean
With annnotated beats:								
Rhythmic	83.9	87.1	75.7	91.9	93.4	-	87.0	84.4
Harmonic	77.2	89.9	80.1	92.9	92.6	-	86.0	82.2
Combined	91.8	89.6	83.6	94.4	96.6	-	89.4	90.4
With detected beats:								
Combined	80.3	79.8	71.3	82.7	83.4	69.3	79.0	77.3
[11]	77.8	81.4	65.7	86.1	83.7	68.9	81.3	76.1
Beat tracking results:								
Beat tracker [1, 25]	89.0	88.4	88.2	88.6	88.2	85.2	90.5	88.3

Table 1: Mean downbeat tracking F-measures across all datasets. The last column shows the mean over all datasets used. The last row shows beat tracking F-measure scores.

Klapuri [23]: This dataset consists of 320 excerpts, covering various genres. The total length is 4h 54m. The beat annotations of this dataset have been made independently of the downbeat annotations and therefore do not always match. Hence, we cannot use the dataset in experiments that rely on annotated beats.

3.2 Evaluation measure

For the evaluation of downbeat tracking we follow [10, 25] and report the F-measure which is computed by $F = 2RP/(R+P)$, where the recall R is the ratio of correctly detected downbeats within a $\pm 70ms$ window and the total number of annotated downbeats, and the precision P is the ratio of correctly detected downbeats within this window and all the reported downbeats.

3.3 Training procedure

All experiments in this section have been carried out using the leave-one-dataset-out approach, to be as comparable as possible with the setting in [11]. After removing the test dataset, we use 75% of the remaining data for training and 25% for validation. To cope with the varying lengths of the audio excerpts, we split the training data into segments of 15 beats and an overlap of 10 beats. For training, we use cross entropy cost, and AdaGrad [9] with a constant learn rate of 0.04 for the rhythmic model and 0.02 for the harmonic model. The hidden units and the biases are initialised with zero, and the weights of the network are randomly sampled from a normal distribution with zero mean and a standard deviation of 0.1. We stop the learning after 100 epochs or when the validation error does not decrease for 15 epochs. For training the GRUs, we used the Lasagne framework [8].

4. RESULTS AND DISCUSSION

4.1 Influence of features

In this section we investigate the influence of the two different input features described in Section 2.1.

The performance of the two different networks is shown in the upper part of Table 1. Looking at the mean scores over all datasets, the rhythmic and harmonic network achieve a comparable performance. The biggest difference between the two was found in the *Ballroom* and the *Hainsworth* dataset, which we believe is mostly due to differing musical content. While the *Ballroom* set consists of music with clear and prominent rhythm which the percussive feature seems to capture well, the *Hainsworth* set also includes chorales with less clear-cut rhythm but more prominent harmonic content which in turn is better represented by the harmonic feature. Interestingly, combining both networks (by averaging the output activations) yields a score that is almost always higher than the score of the single networks. Apparently, the two networks concentrate on different, relevant aspects of the audio signal and combining them enables the system exploiting both. This is in line with the observations in [11] who similarly combined the output of three networks in their system.

4.2 Estimated vs. annotated beat positions

In order to have a fully automatic downbeat tracking system we use the beat tracker proposed in [1] with an enhanced state space [25] as a front-end to our system. [1] We show the beat tracking F-measures per dataset in the bottom row of Table 1. With regard to beat tracking, the datasets seem to be balanced in terms of difficulty.

The detected beats are then used to synchronise the features of the test set. [2] The downbeat scores obtained with the detected beats are shown in the middle part of Table 1. As can be seen, the values are around $10\% - 15\%$ lower than if annotated beats were used. This makes sense, since an error in the beat tracking stage cannot be corrected in a later stage. This might be a drawback of the proposed system compared to [11], where the tatum (instead of the beat) is the basic time unit and the downbeat tracking stage can still decide whether a beat consists of one, two or more tatums.

Although the beat tracking performance is balanced among the datasets, we find clear differences in the downbeat tracking performance. For example, while the beat tracking performance on the *Hainsworth* and the *Robbie Williams* dataset are similar, the downbeat accuracy differs more than 12%. Apparently, the mix of genres, in-

[1] We use the *DBNBeatTracker* included in *madmom* [3] version 0.13.

[2] We took care that there is no overlap between the train and test sets.

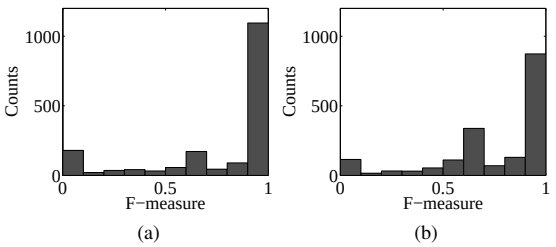


Figure 3: Histogram of the downbeat F-measures of the proposed system (a) and the reference system [11] (b)

cluding time signatures of $2/2$, $3/2$, $3/4$ and $6/8$, in the *Hainsworth* set represents a challenge to downbeat tracking compared to the more simple *Robbie Williams*, which mostly contains $4/4$ time signatures.

4.3 Importance of the DBN stage

System	annotated	detected
RNN	85.0	73.7
RNN+DBN	90.4	77.3

Table 2: Mean downbeat tracking F-measures across all datasets of the proposed, combined system. *annotated* and *detected* means that annotated or detected beats were respectively used to synchronise the features. RNN uses peak-picking to select the downbeats, while RNN+DBN uses the DBN language model.

To assess the importance of the DBN stage (Section 2.3) we implemented a simple baseline, which simply reports downbeats if the resulting (combined) RNN activations exceed a threshold. A threshold of 0.2 was found to yield the best results on the validation set. In Table 2, we show the results of the baseline (RNN) and the results of the combined system (RNN+DBN). As can be seen, the combination of RNN and DBN significantly outperforms the baseline, confirmed by a *Wilcoxon signed-rank* test with $p < 0.01$.

4.4 Comparison to the state-of-the-art

In this section we investigate the performance of our system in relation to the state-of-the-art in downbeat tracking, represented by [11]. Unfortunately, a direct comparison is hindered by various reasons: The datasets used for training the ConvNets [11] are not freely available and the beat tracker at their input stage is different to the one that we use in this work. Therefore, we can only check whether the whole end-to-end system is competitive and leave a modular comparison of the approaches to future work.

In the middle of Table 1 we show the results of the system described in [11], as provided by the authors. The last column shows the mean accuracy over all 1771 excerpts in our dataset. A *paired-sample t-test* did not show any statistically significant differences in the *mean* performance between the two approaches considering all data points. However, a *Wilcoxon signed-rank* test revealed that there is a significant ($p < 0.01$) difference in the *median* F-measure over all data points, which is 89.7% for [11] and 96.2% for the proposed system. Looking at histograms of the obtained scores (see Fig. 3), we found a clear peak at around 66% F-measure, which is typically caused by the beat tracking stage reporting half or double of the correct tempo. The peak is more prominent for the system [11] (Fig. 3b), hence we believe the system might benefit from a more accurate beat tracker.

From this we conclude that overall the proposed system (in combination with the beat tracker [1, 25]) performs comparable to the state-of-the-art when looking at the mean performance and even outperforms the state-of-the-art in terms of the median performance.

4.5 Error analysis

In order to uncover the shortcomings of the proposed model we analysed the errors of a randomly-chosen, small subset of 30 excerpts. We identified two main factors that lead to a low downbeat score. The first one, obviously, are beat tracking errors which get propagated through to the downbeat stage. Most beat tracking errors are octave errors, and among them, the beat tracker mostly tapped twice as fast as the groundtruth tempo. In some cases this is acceptable and therefore would make sense to also allow these metrical levels as, e.g., done in [23]. The second common error is that the downbeat tracker chooses the wrong time signature or has problems following time signature changes or coping with inserted or removed beats. Phase errors are relatively rare. Changing time signatures appear most frequently in the *Beatles* dataset. For this dataset, reducing the transition probability of time signature changes p_r from 10^{-7} to 0 leads to a relative performance drop of 6%, while the results for other datasets remain largely unaffected. Besides, the used datasets mainly contain $3/4$ and $4/4$ time signatures making it impossible for the RNN to learn something meaningful about other time signatures. Here, creating a more balanced training set regarding time signatures would surely help.

5. CONCLUSIONS AND FUTURE WORK

We have proposed a downbeat tracking back-end system that uses recurrent Neural networks (RNNs) to analyse a beat-synchronous feature stream. With estimated beats as input, the system performs comparable to the state-of-the-art, yielding a mean downbeat F-measure of 77.3% on a set of 1771 excerpts of Western music. With manually annotated beats the score goes up to 90.4%.

For future work, a good modular comparison of downbeat tracking approaches needs to be undertaken, possibly with collaboration between several researchers. In particular, standardised dataset train/test splits need to be defined. Second, we would like to train and test the model with non-Western music and 'odd' time signatures, such as done in [21].

The source code will be released as part of the *madmom* library [3], including all trained models and can be found together with additional material under `http://www.cp.jku.at/people/krebs/ismir2016/index.html`.

6. ACKNOWLEDGMENTS

This work is supported by the European Union Seventh Framework Programme FP7 / 2007-2013 through the GiantSteps project (grant agreement no. 610591), the Austrian Science Fund (FWF) project Z159, the Austrian Ministries BMVIT and BMWFW, and the Province of Upper Austria via the COMET Center SCCH. For this research, we have made extensive use of free software, in particular Python, Lasagne, Theano and GNU/Linux. The Tesla K40 used for this research was donated by the NVIDIA corporation. We'd like to thank Simon Durand for giving us access to his downbeat tracking code.

7. REFERENCES

[1] S. Böck, F. Krebs, and G. Widmer. A multi-model approach to beat tracking considering heterogeneous music styles. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, 2014.

[2] S. Böck and M. Schedl. Enhanced beat tracking with context-aware neural networks. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2011.

[3] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer. madmom: a new python audio and music signal processing library. *arXiv:1605.07008*, 2016.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, Dzmitry Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078*, 2014.

[5] M. E. P. Davies and M. D. Plumbley. A spectral difference approach to downbeat extraction in musical audio. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, Florence, 2006.

[6] M.E.P. Davies, N. Degara, and M.D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. *Queen Mary University of London, Tech. Rep. C4DM-09-06*, 2009.

[7] T. De Clercq and D. Temperley. A corpus analysis of rock harmony. *Popular Music*, 30(01):47–70, 2011.

[8] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Eric Battenberg, Aäron van den Oord, et al. Lasagne: First release., August 2015.

[9] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

[10] S. Durand, J. Bello, D. Bertrand, and G. Richard. Downbeat tracking with multiple features and deep neural networks. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, 2015.

[11] S. Durand, J.P. Bello, Bertrand D., and G. Richard. Feature adapted convolutional neural networks for downbeat tracking. In *The 41st IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[12] S. Durand, B. David, and G. Richard. Enhancing downbeat detection when facing different music styles. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3132–3136. IEEE, 2014.

[13] B. D. Giorgi, M. Zanoni, A. Sarti, and S. Tubaro. Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony. In *Proceedings of the 8th International Workshop on Multidimensional Systems*, 2013.

[14] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical and jazz music databases. In *Proceedings of the 3rd International Society for Music Information Retrieval Conference (ISMIR)*, Paris, 2002.

[15] M. Goto and Y. Muraoka. Real-time rhythm tracking for drumless audio signals: Chord change detection for musical decisions. *Speech Communication*, 27(3):311–335, 1999.

[16] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1832–1844, 2006.

[17] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):855–868, 2009.

[18] S. Hainsworth and M. Macleod. Particle filtering applied to musical tempo tracking. *EURASIP Journal on Applied Signal Processing*, 2004:2385–2395, 2004.

[19] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Ng. Deep speech: Scaling up end-to-end speech recognition. *arXiv:1412.5567v2*, 2014.

[20] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[21] A. Holzapfel, F. Krebs, and A. Srinivasamurthy. Tracking the "odd": Meter inference in a culturally diverse music corpus. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, 2014.

[22] T. Jehan. Downbeat prediction by listening and learning. In *Applications of Signal Processing to Audio and Acoustics, 2005. IEEE Workshop on*, pages 267–270. IEEE, 2005.

[23] A. Klapuri, A. Eronen, and J. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–355, 2006.

[24] F. Krebs, S. Böck, and G. Widmer. Rhythmic pattern modeling for beat and downbeat tracking in musical audio. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, 2013.

[25] F. Krebs, S. Böck, and G. Widmer. An efficient state space model for joint tempo and meter tracking. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, Malaga, 2015.

[26] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, Miami, 2011.

[27] H. Papadopoulos and G. Peeters. Joint estimation of chords and downbeats from an audio signal. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):138–152, 2011.

[28] G. Peeters and H. Papadopoulos. Simultaneous beat and downbeat-tracking using a probabilistic framework: theory and large-scale evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, (99):1–1, 2011.

ENHANCING COVER SONG IDENTIFICATION WITH HIERARCHICAL RANK AGGREGATION

Julien Osmalskyj, Marc Van Droogenbroeck, Jean-Jaques Embrechts
INTELSIG Laboratory - University of Liège - Belgium
josmalsky@ulg.ac.be, jjembrechts@ulg.ac.be

ABSTRACT

Cover song identification involves calculating pairwise similarities between a query audio track and a database of reference tracks. While most authors make exclusively use of chroma features, recent work tends to demonstrate that combining similarity estimators based on multiple audio features increases the performance. We improve this approach by using a hierarchical rank aggregation method for combining estimators based on different features. More precisely, we first aggregate estimators based on global features such as the tempo, the duration, the overall loudness, the number of beats, and the average chroma vector. Then, we aggregate the resulting composite estimator with four popular state-of-the-art methods based on chromas as well as timbre sequences. We further introduce a refinement step for the rank aggregation called "local Kemenization" and quantify its benefit for cover song identification. The performance of our method is evaluated on the Second Hand Song dataset. Our experiments show a significant improvement of the performance, up to an increase of more than $200\,\%$ of the number of queries identified in the Top-1, compared to previous results.

1. INTRODUCTION

Given an audio query track, the goal of a cover song identification system is to retrieve at least one different version of the query in a reference database, in order to identify it. In that context, a version can be described as a new performance or recording of a previously recorded track [22]. Retrieving covers is a challenging task, as the different renditions of a song can differ from the original track in terms of tempo, pitch, structure, instrumentation, etc. The usual way of retrieving cover songs in a database involves extracting meaningful features from an audio query first in order to compare them to the corresponding features computed for the other tracks of the database using a pairwise similarity function. The function returns a score or a probability of similarity. Many researchers have been using exclusively chroma features [10, 13, 14, 22] to characterize the songs in the database. Chroma vectors describe the harmony of the songs and are robust to changes in instrumentation and timbre, which makes them quite popular for the task. While chromas are the most used features in the literature, other works investigate the use of different features, such as timbral features [23] or cognition based features [4].

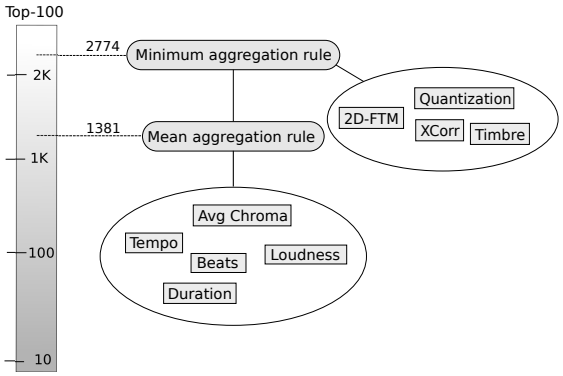


Figure 1. Hierarchical rank aggregation of estimators based on audio features. Global features are first aggregated using the *mean rule*, identifying 1,381 tracks in the top-100, out of 5,464 tracks sampled from the Second Hand Song dataset. The resulting composite estimator is then aggregated with four remaining features using the *minimum rule*, identifying 2,774 tracks in the top-100.

In recent work [16], we established that combining multiple audio features improves the performance of cover song identification systems: designing several classifiers based on different features and combining them through probabilistic rules or rank aggregation techniques improves the performance. In light of this, it seems important to study how state-of-the-art features perform when they are combined for cover song identification. In this paper, we improve upon previous work by considering a total of nine features, including four state-of-the-art ones. These features cover a wide range of audio characteristics, from low-dimensional ones such as the tempo or the duration of the songs, to higher level characteristics such as chromas and timbral sequences. We build similarity estimators for each feature, using supervised machine learning for some of them, and combine them in a hierarchical way to design a new combination method. In this method, we first aggregate five estimators that are based on global features:

© Julien Osmalskyj, Marc Van Droogenbroeck, Jean-Jaques Embrechts. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Julien Osmalskyj, Marc Van Droogenbroeck, Jean-Jaques Embrechts. "Enhancing cover song identification with hierarchical rank aggregation", 17th International Society for Music Information Retrieval Conference, 2016.

tempo, duration, loudness, beats, and averaged chroma. These global features are computed for the entire song, rather than for individual chunks of audio. We show that combining such estimators using rank aggregation methods improves the performance, compared to probabilistic fusion [16]. We then take the resulting aggregated estimator and combine it with four state-of-the-art methods using a different aggregation rule, as shown in Figure 1. We further achieve a higher performance by applying a refinement step called *"local Kemenization"* [7, 8]. We found that this refinement step significantly increases the number of queries identified immediately (Top-1).

2. METHOD OVERVIEW

Identifying cover songs with respect to an audio query involves comparing the query to a set of tracks in a reference database using a similarity function. Considering two input tracks, the function should return a score indicating whether the tracks are considered as being similar or not. Our approach follows the combining method that we proposed in [16]. As there exist several effective features in the literature, the idea is to take advantage of all of them by combining them. We therefore design several pairwise comparison functions, called similarity estimators, based on different audio features. We first consider the same set of estimators as the one used in [16]. We make use of global low-dimensional features such as the tempo, the duration, the number of beats, the overall loudness, and the average chroma vector of a song, learning a probabilistic model to predict the similarity. We also include three estimators based on chromas features. The first and second ones were used in previous works and are respectively based on the quantization of chroma features [11, 16] and the cross-correlation of entire chroma sequences [9]. We add a third chroma estimator based on an efficient large-scale method proposed by Bertin-Mahieux *et al.* [2]. Finally, to take into account timbral information, we include an estimator based on MFCC features. This method, introduced by Tralie *et al.* [23], showed that some covers could be identified based on timbre only.

2.1 Weak estimators

In previous work [16], we demonstrated that global low-dimensional features (tempo, duration, etc.) bring information that helps in the identification process. However, using only such features for identifying cover songs is not enough to achieve good performance. Indeed, such features are considered as *weak* because they only slightly improve a classifier with respect to a purely random classifier. While we combined these features using probabilistic combination rules, we innovate by combining them using rank aggregation techniques. For each feature, we build a probabilistic estimator, using supervised machine learning. To determine the similarity of candidates with respect to the query, we perform pairwise comparisons using the learned probabilistic models to predict probabilities of similarities. Each query is compared to the database using each estimator. We then aggregate the rankings produced by each estimator to build an improved list of results.

2.2 Chroma estimators

2.2.1 Cross-correlation

We design the same cross-correlation estimator as the one used in [16]. This estimator is useful to take into account temporal information. It computes two dimensional cross-correlations between high-pass filtered chroma sequences of the tracks to be compared. The similarity score between two songs is computed as the reciprocal of the peak value of the cross-correlated signal. We refer the reader to the original work [9] for details.

2.2.2 Quantization

To take into account the harmonic distribution of the songs, we make use of an estimator based on the quantization of chroma features [11, 17]. For each track, chroma vectors are mapped to specific codewords. Codewords are determined using a K-Means clustering of 200,000 chromas vectors. We retain 100 clusters for the feature, resulting in a 100-dimensional feature vector. The similarity score is computed as the cosine similarity between two 100-dimensional vectors. To account for key transposition, we make use of the optimal transposition index [21] (OTI) technique, as it has been used in other works [1, 20].

2.2.3 2D Fourier transform magnitude coefficients

This method was first introduced by Bertin-Mahieux *et al.* [2] and was designed as a fast and accurate feature for cover song identification. The idea is to encode harmonic information in a compact representation, to make it invariant to local tempo changes and pitch shifts. First we extract patches of 75 consecutive chromas with an overlap of 1. We then compute the 2D FFT magnitude coefficients for each patch. Next, we aggregate all the patches pointwise using a median rule. Finally, we project the resulting 900-dimensional representation on a 50 dimensional PCA sub-space. Each track is therefore represented by a 50-dimensional vector. The final score between two tracks is computed as the cosine similarity between two projections.

2.3 Timbre estimator

In our base set of estimators, we also include a method proposed by Tralie *et al.* [23], that takes into account the relative evolution of timbre over time. Using careful centering and normalization, the authors were able to design features that are approximately invariant to cover. The features are based on self-similarity matrices of MFCC coefficients and can be used to identify cover songs. Being based on timbre rather than harmony, this feature demonstrates that if the pitch is blurred and obscured, cover song identification should still be possible (see the original paper [23] for a detailed explanation). We designed an estimator based on features that were kindly computed for us by the authors of the method. The similarity score is computed using the Smith-Waterman alignment algorithm.

3. HIERARCHICAL RANK AGGREGATION

3.1 Rank aggregation techniques

To take advantage of all the features that we use, we need a way to combine them. One way of doing that is through probabilistic combination rules. Under the hypothesis that all estimators return probabilities, we can experiment several rules such as the probabilistic product, sum or median rules [5, 6]. The problem is that not all of our estimators return a probability. Some estimators return probabilities, while others return different kinds of scores, for example a cosine similarity or a cross-correlation peak value. One solution for using such rules would be to map scores to probabilities, but there is no straightforward way of doing that. Furthermore, an independent dataset is often mandatory for such a mapping.

Another solution is to combine estimators through rank aggregation techniques, as we proposed in [16]. As a single query q is compared to the entire database using N estimators, we obtain N different orderings of the database. Each track of the database can be found at different positions in the resulting orderings. Based on the positions of the tracks, rank aggregation techniques compute a new position by applying simple rules such as computing the new rank as the mean of the ranks of each track in the initial orderings. Other rules include the minimum, maximum or median rules. Rank aggregation techniques are popular in the web literature [7]. Such techniques are interesting compared to score-based combination because they are intrinsically calibrated and scale-insensitive [19].

In this paper, we aggregate features at different levels. We first aggregate weak features, experimenting with multiple rules. We next use the resulting ranking as a new input for another aggregation rule, by considering our four remaining estimators. We therefore build a hierarchy of two aggregated classifiers (Figure 1) and achieve improved performance compared to previous results [16].

3.2 Optimizing rank aggregation

After several input rankings r_1, r_2, ..., r_k have been aggregated into one final ranking μ using one of the rules proposed before, we can apply a refinement step called *local Kemenization* [8] to further improve the ranking μ. An aggregated ranking is *locally Kemeny optimal* if there are no pairwise swaps of items in the list that will reduce the sum of Kendall τ [8] measures between each input ranking r_i and μ, where $i = 1, \ldots, k$. The sum of the Kendall τ measures with respect to each initial ranking is called "*aggregated Kendall measure*". The Kendall τ measure determines the correlation between two rankings of equal size. It measures the degree to which one list agrees with another [15]. In practice, one way of computing it is to count the number of swaps needed by a bubble sort algorithm to permute one list to the other. Formally, the Kendall τ distance is defined by

$$\tau = \frac{n_c - n_d}{n(n-1)/2}, \qquad (1)$$

where n_c is the number of concordant pairs and n_d is the number of discordant pairs. The denominator corresponds to the total number of pairs of n items in the lists. A pair of tracks (i, j) is *concordant* if i is ranked above j in both lists, and *discordant* otherwise.

Based on this distance measure between rankings, the local Kemenization procedure considers each pair of adjacent tracks in μ and verifies whether a swap will improve the aggregated Kendall measure. In practice, for two adjacent tracks (i, j) in μ, with i ranked above j, the procedure checks whether track j is ranked above i in the majority of the input rankings. If yes, it swaps the two items as it refines the aggregated list with a reduced Kendall distance. The procedure starts from the beginning of the list, and is repeated iteratively for all pairs of tracks, requiring $n - 1$ checks for an aggregated list of length n. Note that the consecutive swaps of the Kemenization process take into account the inclusion of earlier swaps. For implementation details, we refer the reader to our own implementation of several rank aggregation rules with local Kemenization in a C++ library [1]. We used that code to produce results for this paper.

For our task of cover song identification, we apply the local Kemenization step to our final aggregations to improve the overall performance. Detailed results are given in Section 4.

4. EXPERIMENTS AND RESULTS

4.1 Experimental setup

4.1.1 Evaluation database

We evaluate our method on the Second Hand Song dataset [2] (SHS), a subset of the Million Song Dataset (MSD) [3]. The SHS is structured in 5,854 *cliques*, which are groups of 3 cover songs on average, for a total of 18,196 tracks. The dataset does not provide any audio data. Rather than that, it proposes a set of pre-computed features. Since we need independent learning and test sets for learning probabilistic models, we split all tracks in a learning set (LS) containing 70 % of the tracks, and a test set (TS) containing the 30 % remaining tracks. We learn our models on the LS, and evaluate our final system on the TS, containing 5,464 tracks. Following the procedure explained in [16], we get rid of duplicate tracks in the SHS, thus reducing the number of cliques to 5,828.

4.1.2 Estimators settings

For each estimator, we use the pre-computed features in the SHS. As the chroma features provided in the dataset are aligned on onsets rather than the beats, we re-align them on the beats to account for tempo variations within covers, as done in other works [2, 13].

For our weak estimators, we learn probabilistic models using the ExtraTrees algorithm [12], to estimate probabilities of similarity. The models are learned with 1,000 trees

[1] https://github.com/osmju/librag

[2] http://labrosa.ee.columbia.edu/millionsong/secondhand

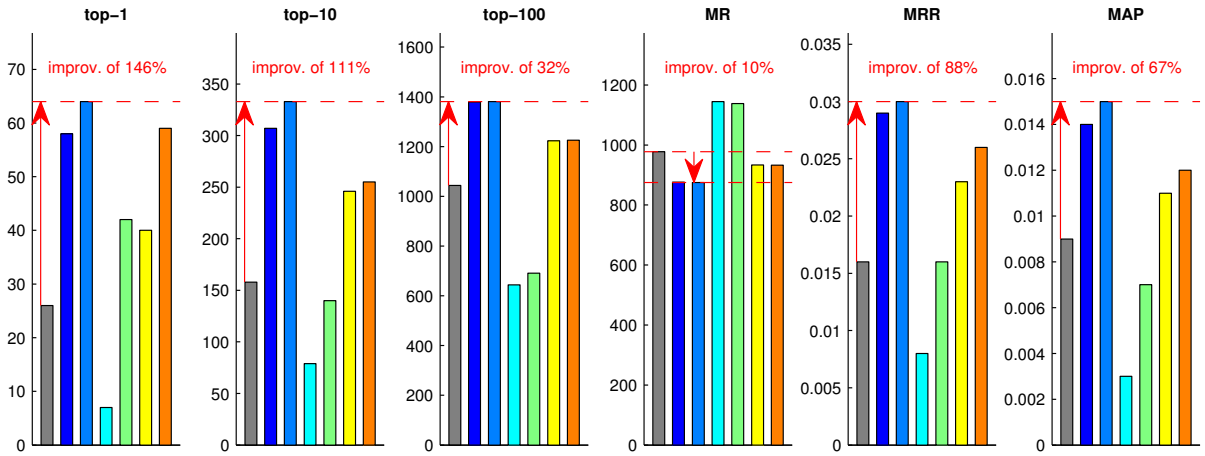


Figure 2. Comparison of the performance of rank aggregation methods for combining estimators based on weak features. The baseline is the probabilistic fusion of weak features proposed in [16]. The mean rank aggregation rule significantly outperforms the baseline, especially in the top-1 with an improvement of 146 %. The red arrows quantify the improvement compared to the baseline. gray baseline, blue mean rank rule, dark blue mean rank rule with local Kemenization, cyan minimum rank rule, green minimim rank rule with local Kemenization, yellow median rank rule, orange median rank rule with local Kemenization.

to reduce the variance, and a maximum depth of 20. The trees are not completely developped to avoid over-fitting. The implementation we use is the Python Scikit-Learn library [18].

To account for key transpositions in our estimator based on the average chroma and in the quantization estimator, we use the optimal transposition index (OTI) technique [21].

For the 2D-FTM estimator, we closely follow the original implementation. We wrote our own C++ implementation, based on the Python code [3] provided by Humphrey *et al.* [13]. We use the FFTW library for computing the 2D-FFT of chroma patches.

Finally, our timbre estimator is close to the original implementation [23], as the features were computed by the author itself for us. The only difference in the implementation comes from the fact that the MFCC sequence of a song in the SHS does not have the same resolution than in the original implementation. We implemented our own version of the sequence alignment Smith-Waterman to compute the final score.

4.2 Aggregation of weak features

Our first experiment consists in aggregating weak features, experimenting with several fusion rules. We take estimators based on the tempo, the duration, the number of beats, the average chroma vectors, the loudness, and aggregate them. We compare the performance to the baseline results obtained in our previous work [16]. We evaluate the performance of the system using standard information retrieval statistics, such as the Mean Rank of the first identified track (MR), the Mean Reciprocal Rank (MRR), and the Mean Average Precision (MAP). Note that the lower the MR is, the better it is, while the goal is to maximize

[3] https://github.com/urinieto/LargeScaleCoverSongId

	Baseline	Mean	Kemeny	Increase
Top-1	26	58	**64**	**+ 146 %**
Top-10	158	307	**333**	**+ 111 %**
Top-100	1,044	1,379	**1,381**	**+ 32 %**
Top-1000	3,729	3,911	**3,911**	**+ 5 %**
MR	977.6	876.2	**875**	**+10 %**
MRR	0.016	0.029	**0.03**	**+ 88 %**
MAP	0.009	0.014	**0.015**	**+ 67 %**

Table 1. Performance achieved with weak estimators when applying the mean rank aggregation rule ("mean" column), and applying the refinement step ("Kemeny" column) to improve the performance.

the MAP and the MRR. We also evaluate the number of queries for which a match is identified in the Top-K, K being a parameter. We present results for the number of tracks identified in the top-1, 10 and 100.

Figure 2 displays the performance of three aggregation rules for the weak estimators with respect to all the metrics. We can notice immediately that the mean aggregation rule outperforms all other combinations. Without local Kemenization, the mean rule provides an improvement of 123 % compared to the baseline, with 58 tracks identified in the top-1 (against 26 in the baseline). That result shows that rank aggregation of these features outperforms the probabilistic rules proposed in previous work. Improvements in terms of the other metrics are also significant and demonstrate the strength of the method. Applying the refinement local Kemenization step, we further improve the performance to 64 tracks identified in the top-1, which corresponds to an increase of 146 % compared to the baseline. Note that the refinement provides surprisingly good improvement, especially for the minimum aggregation rule. Without optimization, we identify 7 tracks in the

	Baseline	Mean	Min	Median
Top-1	328	832	**1010**	839
Top-10	1015	1479	**1785**	1499
Top-100	2015	2669	**2774**	2681
Top-1000	4158	**4456**	4416	4385
MR	726.4	**563**	582	595
MRR	0.107	0.194	**0.234**	0.198
MAP	0.055	0.105	**0.132**	0.106

Table 2. Hierarchical aggregation of all features, with local Kemenization. Best performance is achieved with the minimum rule.

	Mean	Min	Median
Top-1	503	**784**	519
Top-10	1187	**1577**	1106
Top-100	2435	**2535**	2299
Top-1000	**4423**	4290	4309
MR	**593**	651	650
MRR	0.13	**0.19**	0.13
MAP	0.07	**0.1**	0.07

Table 3. Performance of single aggregation rules without hierarchization, with local Kemenization. Performance is not as good as with hierarchization.

top-1. This number jumps to 42 tracks, without changing anything to the base estimators, simply by applying the algorithm presented in Section 3.2. Table 1 quantifies the performance and improvement compared to the baseline for the mean aggregation rule, as it provides the best performance. Note that it is interesting to realize that using such weak features, we still can identify cover songs much better than random guessing.

4.3 Hierarchical aggregation

As the mean rule produces the best experimental results with weak estimators, we consider that resulting aggregated ranking as a single estimator by itself, and combine it with the four remaining estimators based on chroma and timbral features. We experiment the hierarchical combination with the mean, minimum and median rules, as for the weak estimators. Figure 3 displays the performance of each rule, with the local Kemenization step applied. It is straightforward to notice that the minimum rule with Kemenization significantly outperforms the other rules, especially for the top-1, with 1,010 tracks identified in the top-1. For the top-1 metric, we achieve the best performance so far on the subset we use for evaluation, with a MAP set to 0.132 and a MRR set to 0.234. Table 2 quantifies the metrics for all rules. The minimum rule achieves an impressive performance, especially for the top-1 metric, with an improvement of $208\,\%$, and for the MRR and the MAP, with an improvement of respectively $119\,\%$ and $140\,\%$. Note that as we used a significant subset of the SHS (70 %) as a learning set for our probabilistic models, it is difficult to compare our results with other works. Therefore, the baseline here corresponds to the best combination results proposed in our previous work [16]. The baselines in Figures 2 and 3 are different because they respectively correspond to the combination of weak features, as done in [16], and the combination of weak features and chroma based estimators, also as proposed in [16]. Figure 4 shows the performance curves of the aggregations corresponding to the bars in Figure 3. The horizontal axis corresponds to the top-k cutoff, that is the proportion of tracks that are rejected from the final set (the tracks ranked below k). The vertical axis corresponds to the loss, that is the proportion of queries for which no matches at all have been found in the top-k. If at least one corresponding track matches the query, then the loss is set to zero for that query. The second and third charts correspond to zooms in the lower left corner and in the upper right corner. From the performance curves, we clearly observe the improvement compared to the baseline. We also observe that the final curve (minimum rule, green) fits very closely the upper right part of the chart, corresponding to a very high cutoff value. We can reasonably tell that approximately half of the input queries are identified in the top-1% of the returned ranking. Note however how the mean curve (blue) takes the best position at low cutoff values.

4.4 Single aggregation of all features

To quantify the benefit of using *hierarchical* rank aggregation rather than running a single combination, we combined all the features with the three aggregation rules, and applied the refinement step. Aggregating all features in a single run corresponds to setting equal weights to all features. On the other hand, aggregating the results in a hierarchical way corresponds to set different weights to the features. Table 3 gives the performance of all aggregation rules with local Kemenization without any hierarchization. The best performing rule in terms of Top-{1, 10, 100} is again the minimum rule. Similar conclusions yield for the MRR and MAP metrics. For the Top-1000 and the MR, the best rule is the mean aggregation. Overall, the results are worse than using hierarchical aggregation. For the top-1 metric, the number of identified tracks drops by $22\,\%$ for the minimum rule, which is quite significant. For the MRR and the MAP respectively, the performance is decreased by $19\,\%$ and $24\,\%$ for the minimum rule. This demonstrates that attributing different weights to the estimators allows to achieve better performance. Avoiding the hierarchization would lead to a decreased performance, as indicated by the results in Table 3, compared to Table 2.

5. CONCLUSION

In this paper, we improve cover song identification by evaluating multiple rank aggregation rules. Based on previous work, we first construct probabilistic estimators based on multiple weak features such as tempo, duration, loudness, number of beats, and average chroma vectors. We use supervised machine learning to learn models predicting probabilities of similarity. Then, rather than combining the estimators through probabilistic rules, we have evaluated

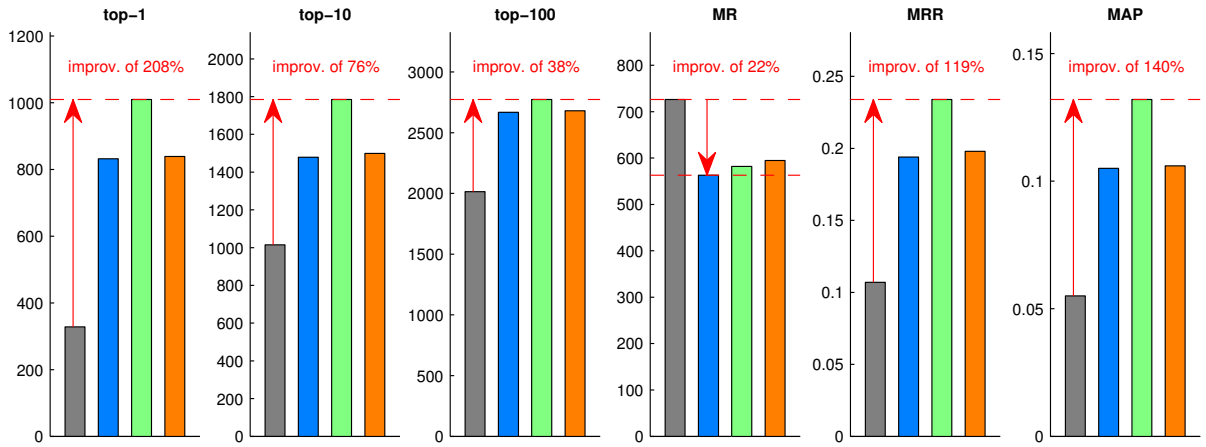


Figure 3. Hierarchical aggregation of all estimators with local Kemenization. The weak estimators are first aggregated using the mean rank aggregation rule. The figure shows how the performance vary when considering different top-level aggregation rules. The red arrows quantify the improvement compared to the baseline. ■ baseline [16], ■ mean rank rule with local Kemenization, ■ minimum rank rule with local Kemenization, ■ median rank rule with local Kemenization.

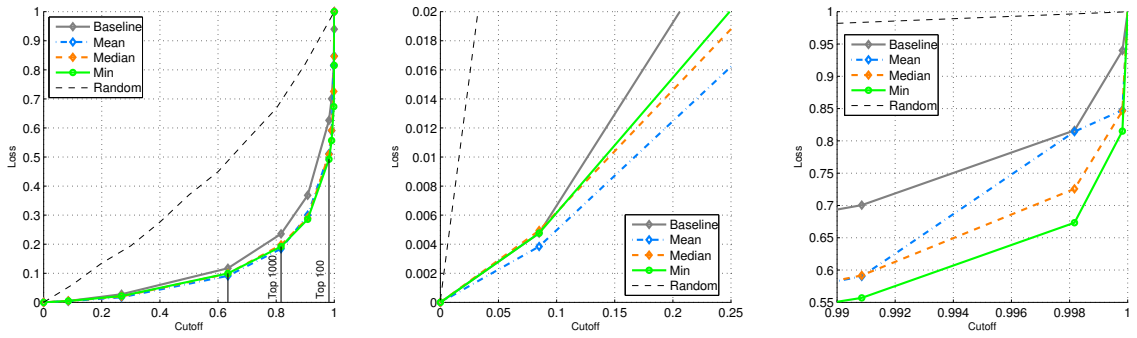


Figure 4. Performance curves of the refined hierarchical aggregation using all features. The x-axis corresponds to the proportion of tracks considered dissimilar by the method and the y-axis corresponds to the proportion of lost queries, that is the proportion of queries for which no matches at all have been found. The second and third charts correspond respectively to a zoom in the lower left part of the first chart, and to a zoom in the upper right corner of the first chart.

several rank aggregation rules, and prove that the mean aggregation rule provides improved results, compared to the baseline. Considering the resulting combined estimator, we further aggregate it with four estimators based on four state-of-the-art features. The selected features take into account harmonic information through chroma features, and timbral information through self-similarity matrices of MFCC coefficients. We further introduce an optimization step, called local Kemenization, that builds an improved aggregated ranking by swapping tracks to the top of the list, with respect to the agreement with each base estimator. To combine the estimators, we aggregate them all in a hierarchical way, evaluating several hierarchical rank aggregation rules. To highlight the gain of using hierarchicak rank aggregation, we also aggregate all nine features through a single aggregation rule, thus allocating an identical weight to all features. We show that such a combination degrades the performance. Our method is evaluated on the Second Hand Song dataset, displaying the performance in terms of standard statistics such as the mean rank of the first identified query, the mean reciprocal rank, the mean average precision and the number of tracks identified at the top-k cutoff. Best results are achieved with the minimum aggregation rule with local Kemenization. Indeed, we are able to identify 1,010 tracks at the first position, which corresponds to $18\,\%$ of the database. In the first 10 tracks returned, we identify 1,785 tracks (32 % of the database), which is a significant improvement over previous work. Compared to previous work on combination, we improve the results by $208\,\%$ in terms of the number of tracks identified in the top-1. In terms of mean reciprocal rank and mean average precision, we achieve improved performance with a value of 0.234 for the MRR and 0.132 for the MAP. The results show that aggregating multiple features, and therefore taking into account multiple sources of musical information, leads to significant improvements in the field of cover song identification. Our method takes advantage of all the best from the literature in that field, and suggests that following that direction of research might eventually lead to an even better performance.

6. REFERENCES

[1] T. Ahonen. Compression-Based Clustering of Chromagram Data : New Method and Representations. In *International Symposium on Computer Music Multidisciplinary Research*, pages 474–481, 2012.

[2] T. Bertin-Mahieux and D. Ellis. Large-scale cover song recognition using the 2d fourier transform magnitude. In *Proceedings of the 13th International Society for Music Information Retrieval (ISMIR)*, pages 241–246, 2012.

[3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Society for Music Information Retrieval (ISMIR)*, pages 591–596, 2011.

[4] S. Downie, H. Xiao, Y. Zhu, J. Zhu, and N. Chen. Modified perceptual linear prediction liftered cepstrum (mplplc) model for pop cover song recognition. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 598–604, Malaga, 2015.

[5] R. Duin. The combining classifier: to train or not to train? *16th International Conference on Pattern Recognition*, 2:765–770, 2002.

[6] R. Duin and D. Tax. Experiments with classifier combining rules. *Lecture Notes in Computer Science*, 31(15):16–29, 2000.

[7] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622, 2001.

[8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank Aggregation Revisited. *Systems Research*, 13(2):86–93, 2001.

[9] D. Ellis, C. Cotton, and M. Mandel. Cross-correlation of beat-synchronous representations for music similarity. In *International conference on acoustics, speech and signal processing (ICASSP)*, pages 57–60, Las Vegas, 2008. IEEE.

[10] D. Ellis and G. Poliner. Identifying Cover Songs withchroma featires and dynamic beat tracking. In IEEE, editor, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–16, New York, 2007. IEEE.

[11] P. Foster, S. Dixon, and A. Klapuri. Identifying Cover Songs Using Information-Theoretic Measures of Similarity. *IEEE Transactions on Audio, Speech and Language Processing*, 23(6):993–1005, 2015.

[12] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[13] E. Humphrey, O. Nieto, and J. Bello. Data Driven and Discriminative Projections for Large-scale Cover Song Identification. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, pages 4–9, 2013.

[14] M. Khadkevich and M. Omologo. Large-Scale Cover Song Identification Using Chord Profiles. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, pages 5–10, 2013.

[15] A. Langville and C. Meyer. *The Science of Rating and Ranking - Who's #1?* Princeton University Press, 2012.

[16] J. Osmalskyj, P. Foster, S. Dixon, and J.J. Embrechts. Combining features for cover song identification. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 462–468, Malaga, 2015.

[17] J. Osmalskyj, S. Pierard, M. Van Droogenbroeck, and J.J. Embrechts. Efficient database pruning for large-scale cover song recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 714–718, Vancouver, BC, 2013.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[19] R. Prati. Combining feature ranking algorithms through rank aggregation. In *Proceedings of the International Joint Conference on Neural Networks*, pages 10–15, 2012.

[20] J. Serrà. *Identification of Versions of the Same Musical Composition by Processing Audio Descriptions*. PhD thesis, 2011.

[21] J. Serrà, E. Gómez, and P. Herrera. Transposing Chroma Representations to a Common Key. In *IEEE Conference on The Use of Symbols to Represent Music and Multimedia Objects*, pages 45–48, 2008.

[22] J. Serrà, E. Gomez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech and Language Processing*, 16(6):1138–1151, 2008.

[23] C. Tralie and P. Bendich. Cover song identification with timbral shape sequences. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 38–44, Malaga, 2015.

ENSEMBLE: A HYBRID HUMAN-MACHINE SYSTEM FOR GENERATING MELODY SCORES FROM AUDIO

Tim Tse[1], Justin Salamon[2], Alex Williams[1], Helga Jiang[1] and Edith Law[1]
[1]University of Waterloo, [2]New York University
{trttse, alex.williams, hhjiang, edith.law}@uwaterloo.ca, justin.salamon@nyu.edu

ABSTRACT

Music transcription is a highly complex task that is difficult for automated algorithms, and equally challenging to people, even those with many years of musical training. Furthermore, there is a shortage of high-quality datasets for training automated transcription algorithms. In this research, we explore a *semi-automated, crowdsourced* approach to generate music transcriptions, by first running an automatic melody transcription algorithm on a (polyphonic) song to produce a series of discrete notes representing the melody, and then soliciting the crowd to correct this melody. We present a novel web-based interface that enables the crowd to correct transcriptions, report results from an experiment to understand the capabilities of non-experts to perform this challenging task, and characterize the characteristics and actions of workers and how they correlate with transcription performance.

1. INTRODUCTION

Music transcription is the process of transforming the acoustic representation of a music piece to a notational representation (e.g., music score). Despite active research on automating this process, music transcription remains a difficult problem [1, 13] for automated algorithms, and equally challenging for human annotators, even those with formal music training. As a result, there is a lack of scalable methods for generating ground truth datasets for training and evaluating music transcription algorithms.

Crowdsourcing has demonstrated great promise as an avenue for generating large datasets. Recent work suggests that the crowd may be capable of performing tasks that require expert knowledge [24, 30]. In this paper, we investigate whether it is feasible to elicit the help of the non-expert crowd to streamline the generation of music transcription ground truth data. Specifically, we introduce a *semi-automated* system, which first extracts a note representation of the melody from an audio file, and then solicits the crowd to correct the melody by making small adjustments to the pitch, onset and offset of each note. Our goal is to characterize the extent to which the crowd can successfully perform this complex task which typically requires musical expertise, describe the relationship between the actions that users take to correct transcriptions and their relationship to performance, and based on these findings, highlight specific challenges associated with crowdsourcing music transcription.

© Tim Tse[1], Justin Salamon[2], Alex Williams[1], Helga Jiang[1] and Edith Law[1]. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Tim Tse[1], Justin Salamon[2], Alex Williams[1], Helga Jiang[1] and Edith Law[1]. "Ensemble: A Hybrid Human-Machine System for Generating Melody Scores from Audio", 17th International Society for Music Information Retrieval Conference, 2016.

2. RELATED WORK

There are three threads of prior work relevant to our research: automatic music transcription (AMT), semi-automatic music transcription, and crowdsourcing the generation of ground truth data in music information retrieval.

In this work, we tackle a specific subtask of automatic music transcription, automatic *melody* transcription. Given a polyphonic music recording that has a clear melodic line, our goal is to transcribe the melody into a piano-roll like representation consisting of a sequence of non-overlapping notes, each with an onset time, offset time, and a pitch value. While several approaches have been proposed to date (see [27] and references therein), the task remains highly challenging and is considered an open problem.

Given that fully automated transcription techniques are at the moment still limited, it is worth investigating how well machines aided by humans, or *semi-automatic* systems, perform at this task. In [15] the authors studied two types of user input for semi-automatic music transcription based on matrix deconvolution. They showed that by asking the users to transcribe a small number of notes from the test data, performance could be significantly improved compared to initializing the model from independent training data. [8] introduced a human-in-the-loop model which solicits users to highlight notes to be extracted from the rest of the audio. In [29], users are asked to hum the melody to be extracted in a sound mixture. A source separation framework that incorporates prior knowledge has been proposed by [20], and the authors have shown that the informed settings outperform the blind settings. Finally, recent work [16] elicits help from 30 users to provide note onsets and pitches as seeds to a semi-automated melody extraction algorithm, and found that experts and novices alike were able to contribute useful information. Likewise, Songle [12], a web service for music listening, provides users with the ability to modify a draft transcription generated by an automated algorithm.

There have been a number of crowdsourcing experiments in music information retrieval that have investigated how to generate ground truth datasets for music tagging [17, 32] and music similarity judgments [33], and for more complicated music information retrieval tasks, such as the creation of emotionally relevant musical scores for audio stories [25]. These prior works found that the non-expert crowd can be incentivized to contribute accurate annotations for music.

3. ENSEMBLE

We propose Ensemble, a two-part architecture to the task of music transcription (Figure 1). The first part involves the task of automatically transcribing the notes of the melody from the audio signal of a music piece. The second part asks the crowd to fix and improve upon the automatically generated score.

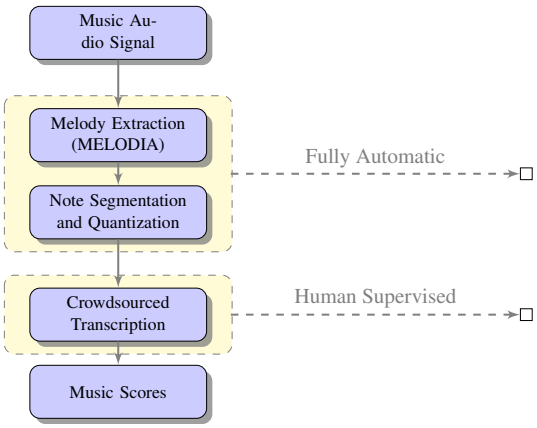


Figure 1. A semi-automatic architecture. Melody extraction and quantization is performed automatically, and this automatically generated transcription is then corrected by the crowd.

3.1 Automatic Melody Note Transcription

We employ a two stage process to produce an automatic transcription of the melody notes from the audio signal. First, we extract the continuous fundamental frequency (f_0) contour of the melody using the MELODIA melody extraction plugin [28]. Next, we quantize the contour in pitch and segment it in time to produce a set of discrete notes. This is performed by means of the following set of simple heuristics: first, every f_0 value is mapped to its nearest semitone pitch assuming an equally tempered scale tuned to 440 Hz. The sequence is then smoothed using a median filter of 250 ms to avoid very short pitch jumps that can occur due to e.g. the presence of vibrato in the unquantized pitch contour. Finally, the sequence must be segmented in time into notes. Since MELODIA already estimates the start and end times of each voiced section (i.e. sections where the melody is present), we only need to identify note transitions within each section, accomplished by simply starting a new note whenever the quantized pitch value changes from one semitone to another. We impose a minimum note duration of 100 ms to avoid very short notes generated by continuous pitch transitions such as glissando. It should be noted that more advanced note segmentation algorithms have been proposed [10, 19], but since our goal is to evaluate the capability of the non-expert crowd to correct the automatic transcriptions (not solve automatic note segmentation), we do not require a state-of-the-art method for this study. Our complete melody note transcription script is available online [1].

3.1.1 Evaluation Metrics and Tools

To evaluate the agreement between two note transcriptions (i.e., automated/crowdsourced transcription against ground truth), we use the evaluation metrics from the MIREX [5] Note Tracking subtask of the Multiple Fundamental Frequency Estimation & Tracking challenge [2]:

$$\text{Precision} = \frac{|\text{correct estimated notes}|}{|\text{estimated notes}|} \quad (1)$$

$$\text{Recall} = \frac{|\text{correct estimated notes}|}{|\text{reference notes}|} \quad (2)$$

$$\text{F-measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

An estimated note is considered to match a reference (ground truth) note if its pitch is within half a semitone (± 50 cents) of the reference pitch, its onset is within ± 50 ms of the reference note's onset, and its offset is within ± 50 ms or 20% of the reference note's duration from the reference note's offset, whichever is greater. MIREX also computes a second version of each metric where note offsets are ignored for note matching, since offsets are both considerably harder to detect automatically and more subjective: our ability to perceive offsets can be strongly affected by the duration of note decay, reverberation and masking [3, 18]. In light of this, and following our own difficulty in annotating offsets for our dataset, for this study we use the metrics that ignore note offsets, as we do not consider it reasonable to expect our participants to be able to accurately match the offsets when our expert was not certain about them in the first place. The metrics are computed using the JAMS [14] evaluation wrapper for the `mir_eval` library [22].

3.2 Crowdsourcing Interface

Figure 2 depicts the interface that we designed for crowd workers to correct the music transcription. The interface consists of two panels: the reference panel (top) and the transcription panel (bottom). The reference panel displays the waveform of the original music clip. The transcription panel displays the current note transcription; initially, this is the transcription automatically generated by the

[1] https://github.com/justinsalamon/audio_to_midi_melodia

[2] http://www.music-ir.org/mirex/wiki/2015:Multiple_Fundamental_Frequency_Estimation_\%26_Tracking_Results_-_MIREX_Dataset#Task_2:Note_Tracking_.28NT.29

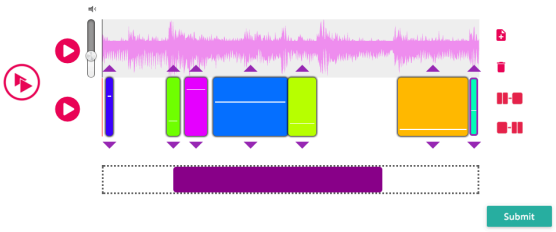


Figure 2. Screenshot of the note editing interface.

MELODIA-based transcription algorithm. Similar to many music editing software, the transcription panel uses rectangles to denote individual notes, height of the white bar within the rectangle (and in our case, also rectangle color) to denote pitch height, and width to denote duration. Workers are given controls to play the reference music clip and transcription separately, as well as simultaneously.

To edit the current transcription, workers can add, delete, split and merge notes. Notes can be added either by clicking on the "add note" button on the right panel or by double-clicking the area where the user wishes to add a note. Notes can be deleted by first clicking on a note, then either clicking the "delete note" button or tapping "backspace" on the keyboard. Adjacent notes can be merged together by clicking on the first note (in time) and then clicking on the "merge notes" button. Similarly, a note can be split into two notes by first clicking on it and then clicking the "split note" button. Pitch can be adjusted (by one semitone at a time) either by using the up and down arrows attached to each note, or by clicking on the note and then pressing the up and down arrow keys on the keyboard. Every time the pitch is adjusted the note is played back to indicate the new pitch. The note can be moved in time by dragging it left or right, and the onset and offset can be adjusted by dragging the note edges left or right. The purple bar at the bottom determines which segment of the audio/transcription is to be played. The bar is adjustable hence allowing the user to selectively choose the portion of the audio to focus on and compare the corrected transcription against.

4. STUDY DESIGN

In this work, our goal is to understand the extent to which crowdworkers can perform transcription correction tasks, which typically requires expertise. To do this, we conducted an experiment via Amazon Mechanical Turk to engage crowdworkers to correct the automatically transcribed melody from a 30 second excerpt of a song. Each 30 s excerpt is sliced into ten 3 s music clips, and each Turker performed corrections on all ten clips, one clip at a time. The reason we use short 3 s clip is that the results of a previous in-lab pilot test suggested that longer clips (10 s) resulted in too much cognitive load on the worker, rendering the task too overwhelming for them. Furthermore, Mechanical Turk workers are more used to performing a series of short, *micro*-tasks.

4.1 Data

For our experiments, we require a dataset of polyphonic music that has clear melodies (such as popular music) with ground truth annotations on a note level. Surprisingly, we found it hard to find suitable data. Most datasets with note annotations are comprised of music that does not contain an unambiguous melody (e.g. chamber music [7, 31] or solo piano [6]), or contain artificially synthesized audio data [9]. The two most relevant datasets, RockCorpus [4] and RWC-Pop [11] also turned out to be problematic: the former only contains pitch-class annotations for notes while the latter was problematic in terms of aligning the audio content to the provided MIDI annotations, which we found to be insufficiently accurate for the purpose of this study. Ultimately, we decided to create our own dataset.

To create the dataset, we selected 20 songs from the Million Song Dataset [2] which had a clear melody, in all cases sung by the human voice. For each song we obtained a MIDI file that was automatically aligned to a 30 s excerpt of the audio from [23]. We then manually identified the MIDI track containing the melody notes and separated it from the rest of the tracks using `pretty_midi` [21]. The separated melody was then loaded into Logic Pro and manually corrected by one of the authors with formal music training to match the melody as accurately as possible, and finally converted to JAMS format [14]. Since transcribing a sung melody into a series of quantized notes is a task that contains a certain degree of subjectivity, the following guidelines were followed:

- The onset and offset of each annotated note should match the audio recording as accurately as possible.
- Pitch is annotated on an equally-tempered semitone scale tuned to 440 Hz.
- Every sung syllable is annotated as a separate note.
- Whenever the pitch of the melody is perceived to change by a semitone or more it should be annotated as a separate note, even if the syllable does not change, including embellishments and glissandos.

The same guidelines were communicated to crowdworkers in the tutorial preceding the experiment.

4.2 Participants

There were a total of 105 Turkers who participated in the study. Each Turker was assigned a randomly chosen 30 s excerpt (sliced into 10 clips 3 s each). The majority of the workers were from the United States. The entire task takes roughly 20-30 minutes, and workers were paid $5 in total. Since we wish to evaluate how well a layperson randomly drawn from the crowd can perform this task, we did not restrict the pool of participants by the level of their formal music training. Finally, our system ensures that each worker takes our study only once.

4.3 Procedure

To begin, workers watch a tutorial video that shows how an expert corrects the transcription for a 3 s music clip. The tutorial guides new users on how to approach this task, by highlighting all of the functionalities of the interface and specifying the rules to follow when considering the correctness of their transcription. The tutorial is followed by a pre-study questionnaire, which asks workers about their music background, including the number of years of formal music training, a set of listening tests (i.e., listening to two consecutive notes and determining which one has a higher pitch), and a set of knowledge questions related to music reading (e.g., whether they are familiar with musical notation such as key and time signature). Workers performed a series of 10 tasks using the note editor interface to correct the transcription of the melody of consecutive 3 s music clips. In order to better understand worker behavior and intent, all interactions with the interface were recorded and saved in a database alongside the workers' corrected melody transcription.

After finishing the transcription, workers are asked to complete a post-study questionnaire, which asks them about their experience using our interface. In particular, we capture motivational factors using the Intrinsic Motivation Inventory (IMI) [26], a scale that measures factors related to *enjoyment* (how much workers enjoy the task), *competence* (how competent workers think they are at the task) and *effort* (how much effort workers put into the task). Workers are asked to rate how much they agree with a set of statements related to these three dimensions on a 7-point Likert scale. For each dimension, we then average the workers' responses (inverting the scores for the negative statements) and use the mean value as the summary statistic for that dimension. Finally, we ask workers to rate the difficulty of the task, as well as comment on ways in which the interface could be improved and the aspects of the task they found most challenging.

5. RESULTS

5.1 Worker Performance

To understand whether the crowd can improve the automatic melody transcription, for each 3 s music clip we compute the F-measure (ignoring offsets) of the automatic (AMT) and the crowd-generated transcriptions against the ground truth. In Figure 3(a) we present the scores of the crowd-annotated transcriptions (green dots) against those of the AMT (blue line) for each 3 s clip, ordered by the score obtained by the AMT. In Figure 3(b) we present the same results, but average the scores for each clip over all workers who annotated that clip. Note that if a data point is located above the $y = x$ line it means the crowd-annotated transcription outperformed the AMT, and vice versa.

The number of points above the line, on the line, and below the line are 272, 366, 338 respectively for subplot (a) and 85, 13, 94 respectively for subplot (b). In general we see that the worker scores vary a lot, with some workers able to correct the AMT to perfectly match the ground truth and others capable of "ruining" an already good AMT. The large number of points on the line in (a) suggests that oftentimes the workers did not think they can improve the AMT and left the transcription unchanged (or modified it insignificantly). In both subplots we observe that when the AMT score exceeds 0.6, the majority of the points fall below the line, suggesting that the crowd has a hard time improving the transcription for clips for which the AMT already performs relatively well. Conversely, when the AMT performs poorly the crowd is capable of making corrections that improve the transcription (i.e., the majority of the data points are on or above the line).

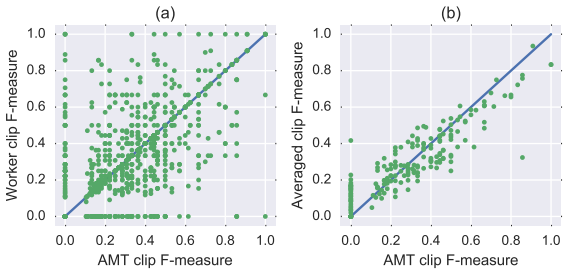


Figure 3. Worker clip F-measure against AMT clip F-measure: (a) individual worker-clip scores, (b) averaged per-clip scores.

5.2 Worker Characteristics

The answers to the pre-study questionnaire are summarized Figure 4: (a) shows the distribution of the workers' musical expertise (T_1), (b) the number of pitch comparison questions answered correctly (T_2), and (c) the number of music notation questions answered correctly (T_3).

We see that the majority of the workers have little to no formal music training with 62% responding "None" or "1 year". 93% of workers answered at least two pitch comparison correctly and 63% answered at least two musical knowledge questions correctly. Given the variability in the scores achieved by the workers, we wanted to see if there was correlation between the workers' answers and their F-measure performance. To determine this, in Figure 4 (d), (e), and (f) we plot the workers' F-measure performance against the three separate topics T_1, T_2 and T_3 respectively. We also compute their Pearson correlation coefficients: 0.12, 0.11 and 0.18 respectively. Results show that the factor most correlated to the workers' performance is their understanding of musical notation. A possible explanation is that the person's proficiency with musical notation is a good indicator of their actual musical expertise. Self-reported expertise is not as good an indicator: this could be (for example) because the worker's musical training happened a long time ago and has since deteriorated through disuse (e.g., an adult took formal music lessons when they were a child but never again). Interestingly, the ability to compare pitches also has a relatively low correlation to the F-measure performance. A possible explanation for this is that the comparison questions (determin-

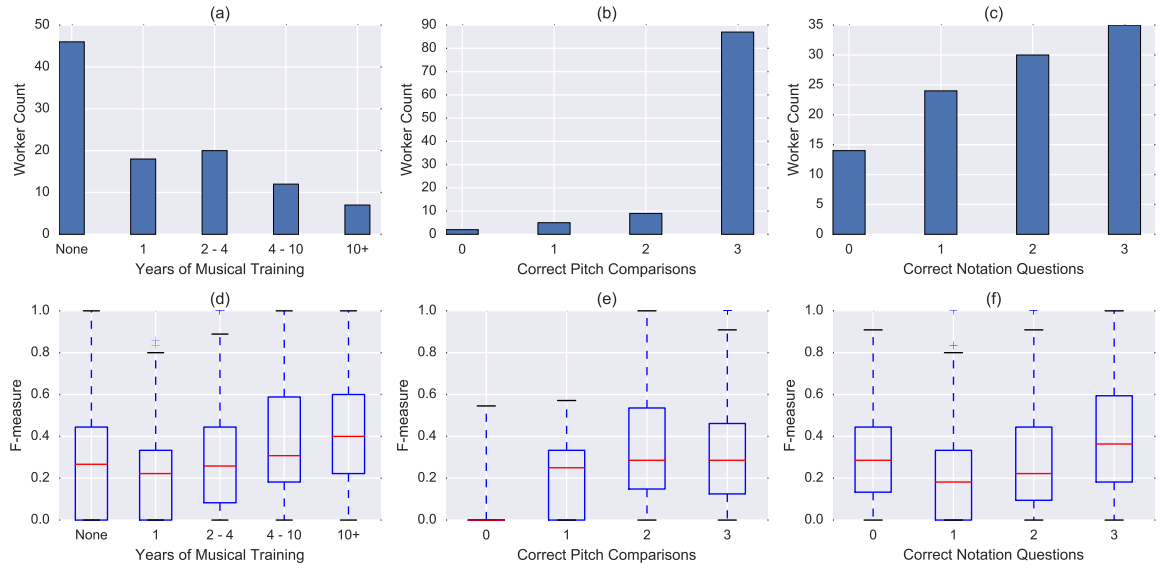


Figure 4. Pre-questionnaire results: (a) years of formal music training, (b) correctly answered pitch comparison questions, (c) correctly answered musical notation questions, (d) years of training vs. performance, (e) correctly answered pitch comparison questions vs. performance, (f) correctly answered musical notation questions vs. performance.

ing which of two pitches is higher) are easier than the task of matching the pitch of a synthesized note to the human voice.

The post-questionnaire shows that there are three main types of challenges. First, many workers ($\sim 20\%$) reported having difficulty matching the pitch of the transcription to the pitch in the audio (e.g., "It's hard to exactly match the notes to the melody."). There were several reasons cited, including *personal inexperience* (e.g., "I'm pretty sure I'm completely tone deaf. It was like trying to learn a foreign language."), *difficulty in separating the vocals from the background* (e.g., "Sometimes it was hard to hear the exact melody over the instrumentation.", "In my specific audio clips, the voice was very ambient and seemed to be layered in octaves at times. So, using only one tone to accurately denote pitch was slightly confusing.") and *difficulty with decorative notes* (e.g., "Vocal inflections/trails can be almost impossible to properly transcribe", "Getting all the nuances of the notes correctly, especially the beginning and ends of notes where the singer sort of "slides" into and out of the note.", "Some changes in tone were particularly difficult to find the right tone to match the voice. Mostly the 'guttural' parts of the singing.", "When someone's voice does that little vibrato tremble thing, it's almost impossible to accurately tab that out.").

Second, some workers reported difficulty with timing and rhythm, knowing *exactly* when things should start and end. For example, one musically trained worker said "Timing was incredibly difficult. I've been a (self taught) musician for 7 years so recognizing pitch wasn't difficult. The hard part was fitting the notes to the parts of the song in the right timing. If there's anything I messed up on during my work, it was that." Finally, a few workers mentioned finding the task difficult due to the lack of feedback indicating whether they were doing the task accurately ("It was hard to tell for sure if I was doing well or not.").

IMI Factor	Mean	Standard Deviation
Enjoyment	5.87	1.46
Competence	4.15	1.79
Effort	6.30	0.93

Table 1. Mean and standard deviation for each IMI factor.

Many workers describe the interface as "intuitive," "easy to work", "well made," and "straightforward". The most requested functionality was the ability to play the audio at a slower speed, e.g., so that one can "catch grace notes or time beats more precisely." In practice this would require us to incorporate a time-stretching algorithm into the interface. Other requests include the ability to duplicate a note, undo previous actions, and get more training and feedback.

Overall, there seems to be an interesting tension: workers find the task extremely challenging, yet enjoyable. For example, workers said "This is one of the best hits on Amazon mTurk.", "Getting it all to sound great is a challenge but fun", "Maybe I'm just not musically inclined, but even after going through several notes, it was still difficult for me to figure out whether or not they matched the singer's voice. Very challenging, but interesting too!"

The quantitative data also reflect this observation. Table 1 summarizes the average intrinsic motivational factors—enjoyment, competence and effort—over all workers. Results show that on average, workers enjoy the task (μ=5.87, σ=1.46), but at the same time, found themselves lacking competence (μ=4.15, σ=1.79) in this task that they find effortful (μ=6.30, σ=0.93). In addition, they reported finding the task difficult (μ=5.57, σ=1.55).

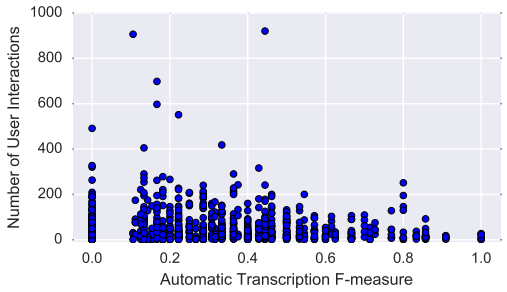


Figure 5. User interaction count vs. AMT performance.

Action	Occur.	Action	Occur.
Play Both	16.1 (26.2)	Resize Play Region	6.3 (8.0)
Change Pitch	12.7 (12.1)	Move Play Region	3.9 (3.8)
Change Offset	9.6 (10.6)	Add Note	2.3 (1.9)
Play Wav	8.3 (13.2)	Merge Note	2.1 (1.8)
Change Onset	7.5 (9.0)	Delete Note	2.0 (1.4)
Play Transcription	7.2 (10.5)	Split Note	1.8 (1.1)

Table 2. Average occurrence (and standard deviation) of each worker action for all 3-second clips.

5.3 Worker Actions

In order to understand more deeply how workers perform the transcriptions, we analyze the action log consisting of a list of coded user actions, including changes in onset, offset, or pitch (up or down), note addition, deletion, merge and split, as well as actions related to re-playing the original song and the transcription (play wav, play transcription, play both, resize-play-region, move-play-region).

In total, there were 53,411 user actions on the interface. The average number of user actions per clip is 60.3, with some workers making as few as 1 edit, and other workers making as many as 920 edits for a single clip. Figure 5 shows that the worse the initial automatic transcription, the more actions the Turkers took to make the corrections, which is quite intuitive.

Table 2 shows the average occurrence of each worker action performed over all transcription tasks. By far, the most frequently taken actions were "change pitch" (changing the pitch of a note up or down) and "play both" (playing both the transcription and the audio clip at the same time). The prevalence of the "change pitch, then replay" interaction potentially reflects workers' general difficulty in determining whether two pitches match. It also may reflect the fact that the interface currently allows only a semitone change at a time. Changing the onset and offset of a note occurred less frequently than changing the pitch, but much more frequently than adding or deleting notes. This behavior could indicate that workers are more inclined to modify a note that exists already in lieu of adding a new note. Together, the results suggest that pitch-matching may be the most challenging aspect of the task, and that workers may have an inherent bias to keep the number of notes in the automated transcription constant, while focusing instead on adjusting the pitch, offset, and onset of existing notes.

6. CONCLUSION

In this paper we introduced Ensemble, a semi-automated system that leverages both algorithms and crowd to perform melody transcription. We reported the characteristics, performance and user-behavior pattern of a non-expert crowd of 105 workers for this complex task. For our experiment, workers were able to improve the initial transcription if it was poor, but found it hard to improve a transcription that was already mostly correct. Despite the crowd workers' sentiment that melody transcription is a difficult task, they also feel that it is a fun and interesting task that can hold their attention. We discover that there is indeed a correlation between the music expertise level of a worker and the F-measure performance of their transcription. Many workers commented on the fact that pitch-matching, while being the most frequent action, is also the most challenging aspect of the task.

In the future we plan to breakdown the results by onset, offset, and pitch-only performance, with the goal of gaining further insight into the strengths and weaknesses of the crowdsourced annotations. Furthermore, currently all transcriptions are evaluated against annotations from a single expert. Since the task is somewhat subjective, we plan to collect additional expert annotations and evaluate expert agreement. This would provide a glass ceiling on the performance we can expect from the untrained crowd. We also plan to develop an aggregation algorithm that collates each worker's contribution to create a single (improved) transcription, investigate whether certain granularities (e.g., shorter/longer clips) and decompositions (e.g., having workers specialize in a particular subtask, such as pitch changes) of the task can produce superior transcriptions, and develop new ways to identify the skillful transcribers in the crowd and incentivize them to perform the task.

7. REFERENCES

[1] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434, 2013.

[2] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *ISMIR*, pages 591–596, Miami, USA, 2011.

[3] A. S. Bregman, P. A. Ahad, and J. Kim. Resetting the pitch-analysis system. 2. role of sudden onsets and offsets in the perception of individual components in a cluster of overlapping tones. *J. Acoust. Soc. Am.*, 96(5):2694–2703, 1994.

[4] T. De Clercq and D. Temperley. A corpus analysis of rock harmony. *Popular Music*, 30(1):47–70, 2011.

[5] J. Stephen Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[6] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE TASLP*, 18(6):1643–1654, 2010.

[7] J. Fritsch. High quality musical audio source separation. Master's thesis, UPMC / IRCAM / Telecom ParisTech, 2012.

[8] B. Fuentes, R. Badeau, and G. Richard. Blind harmonic adaptive decomposition applied to supervised source separation. In *EUSIPCO*, pages 2654–2658, 2012.

[9] J. Ganseman, P. Scheunders, G. J. Mysore, and J. S. Abel. Evaluation of a score-informed source separation system. In *ISMIR*, pages 219–224, 2010.

[10] E. Gómez, F. Cañadas, J. Salamon, J. Bonada, P. Vera, and P. Cabañas. Predominant fundamental frequency estimation vs singing voice separation for the automatic transcription of accompanied flamenco singing. In *ISMIR*, pages 601–606, 2012.

[11] M. Goto. Development of the RWC music database. In *ICA*, pages I–553–556, 2004.

[12] M. Goto, K. Yoshii, H. Fujihara, M. Mauch, and T. Nakano. Songle: A Web Service For Active Music Listening Improved by User Contributions. In *ISMIR*, pages 311–316, 2011.

[13] S. W. Hainsworth and M. D. Macleod. The automated music transcription problem, 2003.

[14] E. J. Humphrey, J. Salamon, O. Nieto, J. Forsyth, R. Bittner, and J. P. Bello. JAMS: A JSON annotated music specification for reproducible MIR research. In *ISMIR*, pages 591–596, 2014.

[15] H. Kirchhoff, S. Dixon, and A. Klapuri. Shift-variant non-negative matrix deconvolution for music transcription. In *IEEE ICASSP*, pages 125–128, 2012.

[16] A. Laaksonen. Semi-Automatic Melody Extraction Using Note Onset Time and Pitch Information From Users. In *SMC*, pages 689–694, 2013.

[17] E. Law and L. von Ahn. Input-agreement: A new mechanism for collecting data using human computation games. In *CHI*, pages 1197–1206, 2009.

[18] R. Mason and S. Harrington. Perception and detection of auditory offsets with single simple musical stimuli in a reverberant environment. In *AES*, pages 331–342, 2007.

[19] E. Molina, L. J. Tardón, A. M. Barbancho, and I. Barbancho. SiPTH: Singing transcription based on hysteresis defined on the pitch-time curve. *IEEE TASLP*, 23(2):252–263, 2015.

[20] A. Ozerov, E. Vincent, and F. Bimbot. A general flexible framework for the handling of prior information in audio source separation. *IEEE TASLP*, 20(4):1118–1133, 2012.

[21] C. Raffel and D. P. W. Ellis. Intuitive analysis, creation and manipulation of midi data with pretty_midi. In *ISMIR*, 2014.

[22] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis. mir_eval: A transparent implementation of common MIR metrics. In *ISMIR*, pages 367–372, 2014.

[23] Colin Raffel. *Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching*. PhD thesis, Columbia University, 2016.

[24] D. Retelny, S. Robaszkiewicz, A. To, W. Lasecki, J. Patel, N. Rahmati, R. Doshi, M. Valentine Melissa, and M. Bernstein. Expert crowdsourcing with flash teams. In *UIST*, pages 75–85, 2014.

[25] S. Rubin and M. Agrawala. Generating emotionally relevant musical scores for audio stories. In *UIST*, pages 439–448, 2014.

[26] R. Ryan. Control and information in the interpersonal sphere: An extension of cognitive evaluation theory. *Journal of Personality and Social Psychology*, 42:450–461, 1982.

[27] M. Ryynänen and A. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, 2008.

[28] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE TASLP*, 20(6):1759–1770, 2012.

[29] P. Smaragdis and G.J. Mysore. Separation by "humming": User-guided sound extraction from monophonic mixtures. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 69–72, 2009.

[30] M. Staffelbach, P. Sempolinski, D. Hachen, A. Kareem, T. Kijewski-Correa, D. Thain, D. Wei, and G. Madey. Lessons learned from an experiment in crowdsourcing complex citizen engineering tasks with amazon mechanical turk. *CoRR*, abs/1406.7588, 2014.

[31] L. Su and Y.-H. Yang. Escaping from the abyss of manual annotation: New methodology of building polyphonic datasets for automatic music transcription. In *ISMIR*, pages 221–233, 2015.

[32] D. Turnbull, R. Liu, L. Barringon, and G. Lanckriet. A game-based approach for collecting semantic annotations of music. In *ISMIR*, 2007.

[33] J. Urbano, J. Morato, M. Marrero, and D. Mart. Crowdsourcing Preference Judgments for Evaluation of Music Similarity Tasks. In *ACM SIGIR Workshop on Crowdsourcing for Search Evaluation*, pages 9–16, 2010.

EXPLORING CUSTOMER REVIEWS FOR MUSIC GENRE CLASSIFICATION AND EVOLUTIONARY STUDIES

Sergio Oramas[1], Luis Espinosa-Anke[2], Aonghus Lawlor[3], Xavier Serra[1], Horacio Saggion[2]
[1]Music Technology Group, Universitat Pompeu Fabra
[2]TALN Group, Universitat Pompeu Fabra
[3]Insight Centre for Data Analytics, University College of Dublin
{sergio.oramas, luis.espinosa, xavier.serra, horacio.saggion}@upf.edu, aonghus.lawlor@insight-centre.org

ABSTRACT

In this paper, we explore a large multimodal dataset of about 65k albums constructed from a combination of Amazon customer reviews, MusicBrainz metadata and AcousticBrainz audio descriptors. Review texts are further enriched with named entity disambiguation along with polarity information derived from an aspect-based sentiment analysis framework. This dataset constitutes the cornerstone of two main contributions: First, we perform experiments on music genre classification, exploring a variety of feature types, including semantic, sentimental and acoustic features. These experiments show that modeling semantic information contributes to outperforming strong bag-of-words baselines. Second, we provide a diachronic study of the criticism of music genres via a quantitative analysis of the polarity associated to musical aspects over time. Our analysis hints at a potential correlation between key cultural and geopolitical events and the language and evolving sentiments found in music reviews.

1. INTRODUCTION

With the democratisation of Internet access, vast amounts of information are generated and stored in online sources, and thus there is great interest in developing techniques for processing this information effectively [27]. The Music Information Retrieval (MIR) community is sensible to this reality, as music consumption has undergone significant changes recently, especially since users are today just one click away from millions of songs [4]. This context results in the existence of large repositories of unstructured knowledge, which have great potential for musicological studies or tasks within MIR such as music recommendation.

In this paper, we put forward an integration procedure for enriching with music-related information a large dataset of Amazon customer reviews [18, 19], with semantic and acoustic metadata obtained from MusicBrainz [1] and AcousticBrainz [2], respectively. MusicBrainz (MB) is a large open music encyclopedia of music metadata, whist AcousticBrainz (AB) is a database of music and audio descriptors, computed from audio recordings via state-of-the-art Music Information Retrieval algorithms [26]. In addition, we further extend the *semantics* of the textual content from two standpoints. First, we apply an aspect-based sentiment analysis framework [7] which provides specific sentiment scores for different aspects present in the text, e.g. album cover, guitar, voice or lyrics. Second, we perform Entity Linking (EL), so that mentions to named entities such as Artist Names or Record Labels are linked to their corresponding Wikipedia entry [24].

This enriched dataset, henceforth referred to as Multimodal Album Reviews Dataset (MARD), includes affective, semantic, acoustic and metadata features. We benefit from this multidimensional information to carry out two experiments. First, we explore the contribution of such features to the Music Genre classification task, consisting in, given a song or album review, predict the genre it belongs to. Second, we use the substantial amount of information at our disposal for performing a diachronic analysis of music criticism. Specifically, we combine the metadata retrieved for each review with their associated sentiment information, and generate visualizations to help us investigate any potential trends in diachronic music appreciation and criticism. Based on this evidence, and since music evokes emotions through mechanisms that are not unique to music [16], we may go as far as using musical information as means for a better understanding of global affairs. Previous studies argue that national confidence may be expressed in any form of art, including music [20], and in fact, there is strong evidence suggesting that our emotional reactions to music have important and far-reaching implications for our beliefs, goals and actions, as members of social and cultural groups [1]. Our analysis hints at a potential correlation between the language used in music reviews and major geopolitical events or economic fluctuations. Finally, we argue that applying sentiment analysis to music corpora may be useful for diachronic musicological studies.

[1] http://musicbrainz.org/
[2] http://acousticbrainz.org

© Sergio Oramas[1], Luis Espinosa-Anke[2], Aonghus Lawlor[3], Xavier Serra[1], Horacio Saggion[2]. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Sergio Oramas[1], Luis Espinosa-Anke[2], Aonghus Lawlor[3], Xavier Serra[1], Horacio Saggion[2]. "Exploring Customer Reviews for Music Genre Classification and Evolutionary Studies", 17th International Society for Music Information Retrieval Conference, 2016.

2. RELATED WORK

One of the earliest attempts on review genre classification is described in [15], where experiments on multiclass genre classification and star rating prediction are described. Similarly, [14] extend these experiments with a novel approach for predicting usages of music via agglomerative clustering, and conclude that bigram features are more informative than unigram features. Moroever, part-of-speech (POS) tags along pattern mining techniques are applied in [8] to extract descriptive patterns for distinguishing negative from positive reviews. Additional textual evidence is leveraged in [5], who consider lyrics as well as texts referring to the meaning of the song, and used for training a kNN classifier for predicting song subjects (e.g. war, sex or drugs).

In [23], a dataset of music reviews is used for album rating prediction by exploiting features derived from sentiment analysis. First, music-related topics are extracted (e.g. artist or music work), and this topic information is further used as features for classification. One of the most thorough works on music reviews is described in [28]. It applies Natural Language Processing (NLP) techniques such as named entity recognition, text segmentation and sentiment analysis to music reviews for generating texts explaining good aspects of songs in recommender systems. In the line of review generation, [9] combine text analysis with acoustic descriptors in order to generate new reviews from the audio signal. Finally, semantic music information is used in [29] to improve topic-wise classification (album, artist, melody, lyrics, etc.) of music reviews using Support Vector Machines. This last approach differs from ours in that it enriches feature vectors by taking advantage of *ad-hoc* music dictionaries, while in our case we take advantage of Semantic Web resources.

As for sentiment classification of text, there is abundant literature on the matter [21], including opinions, reviews and blog posts classification as positive or negative. However, the impact of emotions has received considerably less attention in genre-wise text classification. We aim at bridging this gap by exploring aspect-level sentiment analysis features.

Finally, concerning studies on the evolution of music genres, these have traditionally focused on variation in audio descriptors, e.g. [17], where acoustic descriptors of 17,000 recordings between 1960 and 2010 are analyzed. Descriptors are discretized and redefined as descriptive words derived from several lexicons, which are subsequently used for topic modeling. In addition, [12] analyze expressions located near the keyword *jazz* in newswire collections from the 20th century in order to study the advent and reception of jazz in American popular culture. This work has resemblances to ours in that we also explore how textual evidence can be leveraged, with a particular focus on sentiment analysis, for performing descriptive analyses of music criticism.

3. MULTIMODAL ALBUM REVIEWS DATASET

MARD contains texts and accompanying metadata originally obtained from a much larger dataset of Amazon customer reviews [18, 19]. The original dataset provides millions of review texts together with additional information such as overall rating (between 0 to 5), date of publication, or creator id. Each review is associated to a product and, for each product, additional metadata is also provided, namely Amazon product id, list of similar products, price, sell rank and genre categories. From this initial dataset, we selected the subset of products categorized as *CDs & Vinyls*, which also fulfill the following criteria. First, considering that the Amazon taxonomy of music genres contains 27 labels in the first hierarchy level, and about 500 in total, we obtain a music-relevant subset and select 16 of the 27 which really define a music style and discard for instance region categories (e.g. World Music) and other categories non specifically related to a music style (e.g. Soundtrack, Miscellaneous, Special Interest), function-oriented categories (Karaoke, Holiday & Wedding) or categories whose albums might also be found under other categories (e.g. Opera & Classical Vocal, Broadway & Vocalists). We compiled albums belonging only to one of the 16 selected categories, i.e. no multiclass. Note that the original dataset contains not only reviews about CDs and Vinyls, but also about music DVDs and VHSs. Since these are not strictly speaking music audio products, we filter out those products also classified as "Movies & TV". Finally, since products classified as Classical and Pop are substantially more frequent in the original dataset, we compensate this unbalance by limiting the number of albums of any genre to 10,000. After this preprocessing, MARD amounts to a total of 65,566 albums and 263,525 customer reviews. A breakdown of the number of albums per genre is provided in Table 1.

Genre	Amazon	MusicBrainz	AcousticBrainz
Alternative Rock	2,674	1,696	564
Reggae	509	260	79
Classical	10,000	2,197	587
R&B	2,114	2,950	982
Country	2,771	1,032	424
Jazz	6,890	2,990	863
Metal	1,785	1,294	500
Pop	10,000	4,422	1701
New Age	2,656	638	155
Dance & Electronic	5,106	899	367
Rap & Hip-Hop	1,679	768	207
Latin Music	7,924	3,237	425
Rock	7,315	4,100	1482
Gospel	900	274	33
Blues	1,158	448	135
Folk	2,085	848	179
Total	66,566	28,053	8,683

Table 1: Number of albums by genre with information from the different sources in MARD

Having performed genre filtering, we enrich MARD by extracting artist names and record labels from the Amazon product page. We pivot over this information to query the MB search API to gather additional metadata such as release id, first release date, song titles and song ids. Mapping with MB is performed using the same methodology described in [25], following a pair-wise entity resolution

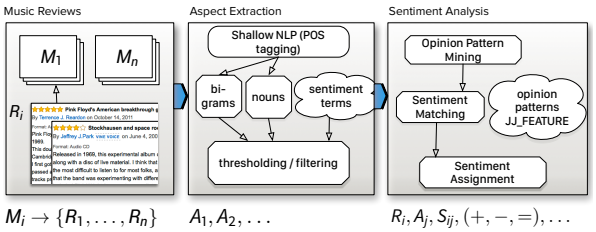


Figure 1: Overview of the opinion mining and sentiment analysis framework.

approach based on string similarity with a threshold value of $\theta = 0.85$. We successfully mapped 28,053 albums to MB. Then, we retrieved songs' audio descriptors from AB. From the 28,053 albums mapped to MB, a total of 8,683 albums are further linked to their corresponding AB entry, which encompasses 65,786 songs. The final dataset is freely available for download [3] .

4. TEXT PROCESSING

In this section we describe how we extract linguistic, sentimental and semantic information from textual reviews. This information will serve both as input features for our genre classification experiments, and also constitute the basis for the diachronic study described in Section 6.

4.1 Sentiment Analysis

Following the work of [6, 7] we use a combination of shallow NLP, opinion mining, and sentiment analysis to extract opinionated features from reviews. For reviews R_i of each album, we mine bi-grams and single-noun aspects (or review features), see [13]; e.g. bi-grams which conform to a noun followed by a noun (e.g. *chorus arrangement*) or an adjective followed by a noun (e.g. *original sound*) are considered, excluding bi-grams whose adjective is a sentiment word (e.g. *excellent*, *terrible*). Separately, single-noun aspects are validated by eliminating nouns that are rarely associated with sentiment words in reviews, since such nouns are unlikely to refer to item aspects. We refer to each of these extracted aspects A_j as review aspects.

For a review aspect A_j we determine if there are any sentiment words in the sentence containing A_j. If not, A_j is marked neutral, otherwise we identify the sentiment word w_{min} with the minimum word-distance to A_j. Next we determine the POS tags for w_{min}, A_i and any words that occur between w_{min} and A_i. We assign a sentiment score between -1 and 1 to A_j based on the sentiment of w_{min}, subject to whether the corresponding sentence contains any negation terms within 4 words of w_{min}. If there are no negation terms, then the sentiment assigned to A_j is that of the sentiment word in the sentiment lexicon; otherwise this sentiment is reversed. Our sentiment lexicon is derived from SentiWordNet [10] and is not specifically tuned for music reviews. An overview of the process is shown in Figure 1. The end result of sentiment analysis is that we determine a sentiment label S_{ij} for each aspect A_j in review R_i. A sample annotated review is shown in Figure 2

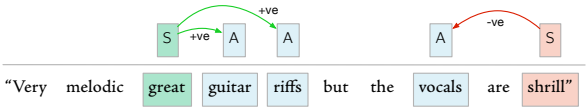


Figure 2: A sentence from a sample review annotated with opinion and aspect pairs.

4.2 Entity Linking

Entity Linking (EL) is the task to provide, given a mention to a named entity (e.g. person, location or organization), its most suitable entry in a reference Knowledge Base (KB) [22]. In our case, EL was performed taking advantage of Tagme [4] [11], an EL system that matches entity candidates with Wikipedia links, and then performs disambiguation exploiting both the in-link graph and the Wikipedia page dataset. TagMe provides for each detected entity, its Wikipedia page id and Wikipedia categories.

5. MUSIC GENRE CLASSIFICATION

5.1 Dataset Description

Starting from MARD, our purpose is to create a subset suitable for genre classification, including 100 albums per genre class. We enforce these albums to be authored by different artists, and that review texts and audio descriptors of their songs are available in MARD. Then, for every album, we selected audio descriptors of the first song of each album as representative sample of the album. From the original 16 genres, 3 of them did not have enough instances complying with these prerequisites (Reggae, Blues and Gospel). This results in a classification dataset composed of 1,300 albums, divided in 13 different genres, with around 1,000 characters of review per album.

5.2 Features

5.2.1 Textual Surface Features

We used a standard Vector Space Model representation of documents, where documents are represented as bag-of-words (BoW) after tokenizing and stopword removal. All words and bigrams (sequences of two words) are weighted according to *tf-idf* measure.

5.2.2 Semantic Features

We enriched the initial BoW vectors with semantic information thanks to the EL step. Specifically, for each named entity disambiguated with Tagme, its Wikipedia ID and its associated categories are added to the feature vector, also with *tf-idf* weighting. Wikipedia categories are organized in a taxonomy, so we enriched the vectors by adding one level more of broader categories to the ones provided by

[3] http://mtg.upf.edu/download/datasets/mard

[4] http://tagme.di.unipi.it/

	Alt. Rock	Classical	Country	Electronic	Folk	Jazz	Latin	Metal	New Age	Pop	R&B	Hip-Hop	Rock
Alt. Rock	28 / 42	1 / 3	3 / 1	10 / 10	7 / 1	1 / 2	2 / 0	18 / 12	10 / 2	4 / 10	3 / 6	3 / 2	10 / 9
Classical	0 / 0	87 / 95	1 / 0	0 / 0	1 / 1	1 / 1	2 / 2	1 / 0	5 / 1	1 / 0	0 / 0	0 / 0	1 / 0
Country	2 / 1	0 / 0	51 / 84	3 / 0	9 / 1	9 / 0	3 / 0	0 / 1	3 / 0	8 / 8	6 / 4	1 / 0	5 / 1
Electronic	7 / 3	3 / 1	1 / 2	40 / 61	4 / 1	1 / 2	2 / 2	6 / 0	7 / 5	6 / 5	6 / 7	13 / 5	4 / 7
Folk	4 / 6	11 / 0	13 / 10	7 / 0	27 / 55	6 / 1	7 / 3	4 / 2	6 / 9	5 / 9	6 / 4	1 / 0	3 / 1
Jazz	7 / 0	10 / 1	6 / 2	2 / 2	5 / 0	45 / 82	6 / 3	3 / 0	8 / 2	3 / 5	4 / 1	1 / 1	0 / 1
Latin	4 / 3	6 / 4	9 / 2	1 / 2	5 / 1	10 / 2	28 / 78	3 / 0	6 / 2	11 / 4	7 / 2	5 / 0	5 / 0
Metal	13 / 5	1 / 0	1 / 1	2 / 2	1 / 0	0 / 1	1 / 0	63 / 87	1 / 0	1 / 0	3 / 1	1 / 0	12 / 3
New Age	9 / 2	7 / 6	9 / 0	7 / 4	10 / 10	9 / 2	7 / 6	3 / 3	15 / 53	10 / 7	6 / 1	2 / 1	6 / 5
Pop	6 / 2	9 / 1	10 / 2	9 / 2	5 / 3	9 / 2	5 / 2	2 / 0	7 / 1	19 / 73	7 / 6	2 / 2	10 / 5
R&B	8 / 2	0 / 1	16 / 3	8 / 4	2 / 0	5 / 3	5 / 0	1 / 0	3 / 0	7 / 10	24 / 71	17 / 5	4 / 1
Hip-Hop	8 / 2	0 / 0	2 / 1	8 / 2	0 / 1	0 / 1	1 / 0	4 / 3	2 / 0	4 / 1	7 / 2	61 / 86	3 / 1
Rock	17 / 15	1 / 2	6 / 8	4 / 7	10 / 5	2 / 4	7 / 1	12 / 13	4 / 1	9 / 7	7 / 4	6 / 2	15 / 31

Table 2: Confusion matrix showing results derived from AB acoustic-based classifier/BoW+SEM text-based approach.

Tagme. Broader categories were obtained by querying DBpedia [5] .

5.2.3 *Sentiment Features*

Based on those aspects and associated polarity extracted with the opinion mining framework, with an average number of aspects per review around 37, we follow [21] and implement a set of sentiment features, namely:

- Positive to All Emotion Ratio: fraction of all sentimental features which are identified as positive (sentiment score greater than 0).
- Document Emotion Ratio: fraction of total words with sentiments attached. This feature captures the degree of affectivity of a document regardless of its polarity.
- Emotion Strength: This document-level feature is computed by averaging sentiment scores over all aspects in the document.
- F-Score [6] : This feature has proven useful for describing the contextuality/formality of language. It takes into consideration the presence of *a priori* "descriptive" POS tags (nouns and adjectives), as opposed to "action" ones such as verbs or adverbs.

5.2.4 *Acoustic Features*

Acoustic features are obtained from AB. They are computed using Essentia [7] . These encompass loudness, dynamics, spectral shape of the signal, as well as additional descriptors such as time-domain, rhythm, and tone [26].

5.3 Baseline approaches

Two baseline systems are implemented. First, we implement the text-based approach described in [15] for music review genre classification. In this work, a Naïve Bayes classifier is trained on a collection of 1,000 review texts, and after preprocessing (tokenisation and stemming), BoW features based on document frequencies are generated. The second baseline is computed using the AB framework for song classification [26]. Here, genre classification is computed using multi-class support vector machines (SVMs) with a one-vs.-one voting strategy. The classifier is trained with the set of low-level features present in AB.

[5] http://dbpedia.org
[6] Not to be confused with the evaluation metric.
[7] http://essentia.upf.edu/

	BoW	BoW+SEM	BoW+SENT
Linear SVM	**0.629**	**0.691**	**0.634**
Ridge Classifier	0.627	0.689	0.61
Random Forest	0.537	0.6	0.521

Table 3: Accuracy of the different classifiers

5.4 Experiments

We tested several classifiers typically used for text classification, namely Linear SVM, Ridge Classifier and Nearest Centroid, using the implementations provided by the scikit-learn library [8] . Among them, Linear SVM has shown better performance when combining different feature sets (see Table 3). Therefore, we trained a Linear SVM classifier with L2 penalty over different subsets of the features described in Section 5.2, which are combined via linear aggregation. Specifically, we combine the different feature sets into five systems, namely **BoW** (BoW), **BoW+Semantic** without broader categories (BoW+SEM), **BoW+Semantic Broader** with broader categories (BoW+SEMb), **BoW+Sentiment** (BoW+SENT) and **BoW+Semantic+Sentiment** (BoW+SEM+SENT). In this way, we aim at understanding the extent to which sentiment and semantic features (and their interaction) may contribute to the review genre classification task. Note that this paper is focused on the influence of textual features in genre classification, and classification based on acoustic features is simply used as a baseline for comparison. A proper combination of acoustic and textual features in text classification is a challenging problem and would require a deeper study that is out of the scope of this paper. The dataset is split 80-20% for training and testing, and accuracy values are obtained after 5-fold cross validation.

5.5 Results and Discussion

Accuracy results of the two baseline approaches introduced in Section 5.3 along with our approach variants are shown in Figure 3. At first sight, we may conclude that sentiment features contribute to slightly outperforming purely text-based approaches. This result implies that

[8] http://scikit-learn.org/

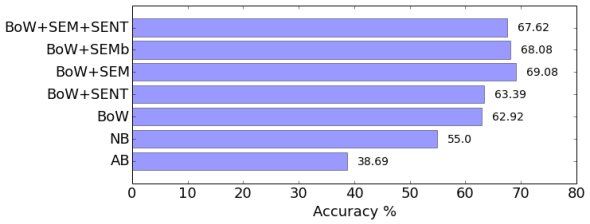


Figure 3: Percentage of accuracy of the different approaches. AB refers to the AcousticBrainz framework. NB refers to the method based on Naïve Bayes from [15].

affective language present in a music review is not a salient feature for genre classification (at least with the technology we applied), although it certainly helps. On the contrary, semantic features clearly boost pure text-based features, achieving 69.08% of accuracy. The inclusion of broader categories does not improve the results in the semantic approach. The combination of semantic and sentiment features improves the BoW approach, but the achieved accuracy is slightly lower than using semantic features only.

Let us review the results obtained with baseline systems. The Naïve Bayes approach from [15] is reported to achieve an accuracy of 78%, while in our results it is below 55%. The difference in accuracy may be due to the substantial difference in length of the review texts. In [15], review texts were at least 3,000 characters long, much larger that ours. Moreover, the addition of a distinction between Classic Rock and Alternative Rock is penalizing our results. As for the acoustic-based approach, although the obtained accuracy may seem low, it is in fact a good result for purely audio-based genre classification, given the high number of classes and the absence of artist bias in the dataset [3]. Finally, we refer to Table 2 to highlight the fact that the text-based approach clearly outperforms the acoustic-based classifier, although in general both show a similar behaviour across genres. Also, note the low accuracy for both Classic Rock and Alternative Rock, which suggests that their difference is subtle enough for making it a hard problem for automatic classification.

6. DIACHRONIC STUDY OF MUSIC CRITICISM

We carried out a study of the evolution of music criticism from two different temporal standpoints. Specifically, we consider when the review was written and, in addition, when the album was first published. Since we have sentiment information available for each review, we first computed an average sentiment score for each year of review publication (between 2000 and 2014). In this way, we may detect any significant fluctuation in the evolution of affective language during the 21st century. Then, we also calculated the average sentiment for each review by year of album publication. This information is obtained from MB and complemented with the average of the Amazon rating scores.

In what follows, we show visualizations for sentiment scores and correlation with ratings given by Amazon users, according to these two different temporal dimensions. Although arriving to musicological conclusions is out of the scope of this paper, we provide *food for thought* and present the readers with hypotheses that may explain some of the facts revealed by these data-driven trends.

6.1 Evolution by Review Publication Year

We applied sentiment and rating average calculations to the whole MARD dataset, grouping album reviews by year of publication of the review. Figure 4a shows the average of the sentiment scores associated to every aspect identified by the sentiment analysis framework in all the reviews published in a specific year, whilst Figure 4b shows average review ratings per year. At first sight, we do not observe any correlation between the trends illustrated in the figures. However, the sentiment curve (Figure 4a) shows a remarkable peak in 2008, a slightly lower one in 2013, and a low between 2003 and 2007, and also between 2009 and 2012. It is not trivial to give a proper explanation of this variations on the average sentiment. We speculate that these curve fluctuations may suggest some influence of economical or geopolitical circumstances in the language used in the reviews, such as the 2008 election of Barack Obama as president of the US. As stated by the political scientist Dominique Moïsi in [20]:

> In November 2008, at least for a time, hope prevailed over fear. The wall of racial prejudice fell as surely as the wall of oppression had fallen in Berlin twenty years earlier [...] Yet the emotional dimension of this election and the sense of pride it created in many Americans must not be underestimated.

Another factor that might be related to the positiveness in use of language is the economical situation. After several years of continuous economic growth, in 2007 a global economic crisis started [9], whose consequences were visible in the society after 2008 (see Figure 4c). In any case, further study of the different implied variables is necessary to reinforce any of these hypotheses.

.

6.2 Evolution by Album Publication Year

In this case, we study the evolution of the polarity of language by grouping reviews according to the album publication date. This date was gathered from MB, meaning that this study is conducted on the 42,1% of the MARD that was successfully mapped. We compared again the evolution of the average sentiment polarity (Figure 4d) with the evolution of the average rating (Figure 4e). Contrary to the results observed by review publication year, here we observe a strong correlation between ratings and sentiment polarity. To corroborate that, we computed first a smoothed version of the average graphs, by applying 1-D convolution (see line in red in Figures 4d and 4e). Then we computed Pearson's correlation between smoothed curves, obtaining a correlation $r = 0.75$, and a p-value $p \ll 0.001$. This means that in fact there is a strong correlation between

[9] https://research.stlouisfed.org

(a) Sentiment (b) Rating (c) USA GDP trend

(d) Sentiment (e) Rating (f) Sentiment by genre

Figure 4: Sentiment and rating averages by review publication year (a and b); GDP trend in USA from 2000 to 2014 (c), and sentiment and rating averages by album publication year (d, e and f)

the polarity identified by the sentiment analysis framework in the review texts, and the rating scores provided by the users. This correlation reinforces the conclusions that may be drawn from the sentiment analysis data.

To further dig into the utility of this polarity measure for studying genre evolution, we also computed the smoothed curve of the average sentiment by genre, and illustrate it with two idiosyncratic genres, namely *Pop* and *Reggae* (see Figure 4f). We observe in the case of *Reggae* that there is a time period where reviews have a substantial use of a more positive language between the second half of the 70s and the first half of the 80s, an epoch which is often called the golden age of *Reggae* [2]. This might be related to the publication of Bob Marley albums, one of the most influential artists in this genre, and the worldwide spread popularity of reggae music. In the case of *Pop*, we observe a more constant sentiment average. However, in the 60s and the beginning of 70s there are higher values, probably consequence by the release of albums by The Beatles. These results show that the use of sentiment analysis on music reviews over certain timelines may be useful to study genre evolution and identify influential events.

7. CONCLUSIONS AND FUTURE WORK

In this work we have presented MARD, a multimodal dataset of album customer reviews combining text, metadata and acoustic features gathered from Amazon, MB and AB respectively. Customer review texts are further enriched with named entity disambiguation along with polarity information derived from aspect-based sentiment analysis. Based on this information, a text-based genre classifier is trained using different combinations of features. A comparative evaluation of features suggests that a combination of bag-of-words and semantic information has higher discriminative power, outperforming competing systems in terms of accuracy. Our diachronic study of the sentiment polarity expressed in customer reviews explores two interesting ideas. First, the analysis of reviews classified by year of review publication suggests that geopolitical events or macro-economical circumstances may influence the way people speak about music. Second, an analysis of the reviews classified by year of album publication is presented. The results show how sentiment analysis can be very useful to study the evolution of music genres. The correlation observed between average rating and sentiment scores suggest the suitability of the proposed sentiment-based approach to predict user satisfaction with musical products. Moreover, according to the observed trend curves, we can state that we are now in one of the best periods of the recent history of music. Further work is necessary to elaborate on these hypotheses. In addition, the combination of audio and textual features is still an open problem, not only for classification but also for the study of the evolution of music. We expect the released dataset will be explored in multiple ways for the development of multimodal research approaches in MIR. In conclusion, the main contribution of this work is a demonstration of the utility of applying systematic linguistic processing on texts about music. Furthermore, we foresee our method to be of interest for musicologists, sociologists and humanities researchers in general.

8. ACKNOWLEDGEMENTS

This work was partially funded by the Spanish Ministry of Economy and Competitiveness under the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502), by the TUNER project (TIN2015-65308-C5-5-R, MINECO/FEDER, UE), by the Keystone COST Action IC1302 and by the Insight Centre for Data Analytics under grant number SFI/12/RC/2289.

9. REFERENCES

[1] C S Alcorta, R Sosis, and D Finkel. Ritual harmony: Toward an evolutionary theory of music. *Behavioral*

and Brain Sciences, 31(5):576–+, 2008.

[2] Michael Randolph Alleyne and Sly Dunbar. *The Encyclopedia of Reggae: The Golden Age of Roots Reggae*. 2012.

[3] Dmitry Bogdanov, Alastair Porter, Perfecto Herrera, and Xavier Serra. Cross-collection evaluation for music classification tasks. In *ISMIR'16*, 2016.

[4] Òscar Celma and Perfecto Herrera. A new approach to evaluating novel recommendations. In *RecSys'08*, pages 179–186, 2008.

[5] Kahyun Choi, Jin Ha Lee, and J. Stephen Downie. What is this song about anyway?: Automatic classification of subject using user interpretations and lyrics. *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pages 453–454, 2014.

[6] Ruihai Dong, Michael P O'Mahony, and Barry Smyth. Further Experiments in Opinionated Product Recommendation. In *ICCBR'14*, pages 110–124, Cork, Ireland, September 2014.

[7] Ruihai Dong, Markus Schaal, Michael P. O'Mahony, and Barry Smyth. Topic Extraction from Online Reviews for Classification and Recommendation. IJCAI'13, pages 1310–1316, 2013.

[8] J. Stephen Downie and Xiao Hu. Review mining for music digital libraries:phase II. *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries*, page 196, 2006.

[9] Daniel P W Ellis. Automatic record reviews. *ICMIR*, 2004.

[10] Andrea Esuli and Fabrizio Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422. Citeseer, 2006.

[11] Paolo Ferragina and Ugo Scaiella. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *Software, IEEE*, 29(1), June 2012.

[12] Maristella Johanna Feustle. Lexicon of Jazz invective: Hurling insults across a century with Big Data. *IAML/IMS'15*, 2015.

[13] Minqing Hu and Bing Liu. Mining Opinion Features in Customer Reviews. In *AAAI'04*, pages 755–760, San Jose, California, 2004.

[14] Xiao Hu and J Stephen Downie. Stylistics in customer reviews of cultural objects. *SIGIR Forum*, pages 49–51, 2006.

[15] Xiao Hu, J Stephen Downie, Kris West, and Andreas Ehmann. Mining Music Reviews: Promising Preliminary Results. *ISMIR*, 2005.

[16] Patrik N Juslin and Daniel Västfjäll. Emotional responses to music: the need to consider underlying mechanisms. *The Behavioral and brain sciences*, 31(5):559–621, 2008.

[17] Matthias Mauch, Robert M MacCallum, Mark Levy, and Armand M Leroi. The evolution of popular music: USA 1960-2010. *Royal Society Open Science*, 2015.

[18] Julian McAuley, Rahul Pandey, and Jure Leskovec. Inferring Networks of Substitutable and Complementary Products. *KDD'15*, page 12, 2015.

[19] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based Recommendations on Styles and Substitutes. *SIGIR'15*, pages 1–11, 2015.

[20] Dominique Moisi. *The Geopolitics of Emotion: How Cultures of Fear, Humiliation, and Hope are Reshaping the World*. Anchor Books, New York, NY, USA, 2010.

[21] Calkin Suero Montero, Myriam Munezero, and Tuomo Kakkonen. *Computational Linguistics and Intelligent Text Processing*, pages 98–114, 2014.

[22] Andrea Moro, Alessandro Raganato, and Roberto Navigli. Entity Linking meets Word Sense Disambiguation: A Unified Approach. *TACL'14*, 2:231–244, 2014.

[23] Tony Mullen and Nigel Collier. Sentiment Analysis using Support Vector Machines with Diverse Information Sources. *EMNLP'04*, pages 412–418, 2004.

[24] Sergio Oramas, Luis Espinosa-anke, Mohamed Sordo, Horacio Saggion, and Xavier Serra. ELMD: An Automatically Generated Entity Linking Gold Standard in the Music Domain. *In LREC'16*, 2016.

[25] Sergio Oramas, Francisco Gómez, Emilia Gómez, and Joaquín Mora. Flabase: Towards the creation of a flamenco music knowledge base. In *ISMIR'15*, 2015.

[26] Alastair Porter, Dmitry Bogdanov, Robert Kaye, Roman Tsukanov, and Xavier Serra. Acousticbrainz: a community platform for gathering music information obtained from audio. *ISMIR'15*, pages 786–792, 2015.

[27] Maria Ruiz-Casado, Enrique Alfonseca, Manabu Okumura, and Pablo Castells. Information extraction and semantic annotation of wikipedia. *Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, pages 145–169, 2008.

[28] Swati Tata and Barbara Di Eugenio. Generating Fine-Grained Reviews of Songs from Album Reviews. *Proceedings of the 48th ACL Anual Meeting*, (July):1376–1385, 2010.

[29] Wei Zheng, Chaokun Wang, Rui Li, Xiaoping Ou, and Weijun Chen. Music Review Classification Enhanced by Semantic Information. *Web Technologies and Applications*, 6612(60803016):5–16, 2011.

FURTHER STEPS TOWARDS A STANDARD TESTBED FOR OPTICAL MUSIC RECOGNITION

Jan Hajič jr.[1] **Jiří Novotný**[2] **Pavel Pecina**[1] **Jaroslav Pokorný**[2]
[1] Charles University, Institute of Formal and Applied Linguistics, Czech Republic
[2] Charles University, Department of Software Engineering, Czech Republic
hajicj@ufal.mff.cuni.cz, novotny@ksi.mff.cuni.cz

ABSTRACT

Evaluating Optical Music Recognition (OMR) is notoriously difficult and automated end-to-end OMR evaluation metrics are not available to guide development. In "Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images", Byrd and Simonsen recently stress that a benchmarking standard is needed in the OMR community, both with regards to data and evaluation metrics. We build on their analysis and definitions and present a prototype of an OMR benchmark. We do not, however, presume to present a complete solution to the complex problem of OMR benchmarking. Our contributions are: (a) an attempt to define a multi-level OMR benchmark dataset and a practical prototype implementation for both printed and handwritten scores, (b) a corpus-based methodology for assessing automated evaluation metrics, and an underlying corpus of over 1000 qualified relative cost-to-correct judgments. We then assess several straightforward automated MusicXML evaluation metrics against this corpus to establish a baseline over which further metrics can improve.

1. INTRODUCTION

Optical Music Recognition (OMR) suffers from a lack of evaluation standards and benchmark datasets. There is presently no publicly available way of comparing various OMR tools and assessing their performance. While it has been argued that OMR can go far even in the absence of such standards [7], the lack of benchmarks and difficulty of evaluation has been noted on multiple occasions [2, 16, 21]. The need for end-to-end system evaluation (at the final level of OMR when musical content is reconstructed and made available for further processing), is most pressing when comparing against commercial systems such as PhotoScore, [1] SmartScore [2] or SharpEye [3]: these typically perform as "black boxes", so evaluating on the level of individual symbols requires a large amount of human effort for assessing symbols and their locations, as done by Bellini et al. [18] or Sapp [19].

OMR systems have varying goals, which should be reflected in evaluation. Helping speed up transcription should be measured by some cost-to-correct metric; a hypothetical automated score interpretation system could require accurate MIDI, but does not need to resolve all slurs and other symbols; digitizing archive scores for retrieval should be measured by retrieval accuracy; etc. We focus on evaluating transcription, as it is most sensitive to errors and most lacking in evaluation metrics.

Some OMR subtasks (binarization, staff identification and removal, symbol localization and classification) have natural ways of evaluating, but the end-to-end task does not: it is difficult to say how good a semantic representation (e.g., MusicXML) is. Manually evaluating system outputs is costly, slow and difficult to replicate; and aside from Knopke and Byrd [12], Szwoch [20] and Padilla et al. [21], we know of no attempts to even define an automatic OMR evaluation metric, much less define a methodology for assessing how well it actually evaluates.

Our contribution does not presume to define an entire evaluation standard. Instead, we propose a robust, cumulative, data-driven methodology for *creating* one. We collect human preference data that can serve as a gold standard for comparing MusicXML automated evaluation metrics, mirroring how the BLEU metric and its derivatives has been established as an evaluation metric for the similarly elusive task of assessing machine translation based on agreement with human judgements [17]. This "evaluating evaluation" approach is inspired by the Metrics track of the Workshop of Statistical Machine Translation competition (WMT) [3, 5, 14]. To collect cost-to-correct estimates for various notation errors, we generate a set of synthetically distorted "recognition outputs" from a set of equally synthetic "true scores". Then, annotators are shown examples consisting of a true score and a pair of the distorted scores, and they are asked to choose the simulated recognition output that would take them less time to correct.

Additionally, we provide an OMR bechmark dataset prototype with ground truth at the symbol and end-to-end levels.

The main contributions of our work are:

- A corpus-based "evaluating evaluation" methodol-

[1] http://www.neuratron.com/photoscore.htm
[2] http://www.musitek.com/index.html
[3] http://www.visiv.co.uk

© Jan Hajič jr., Jiří Novotný, Pavel Pecina, Jaroslav Pokorný. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jan Hajič jr., Jiří Novotný, Pavel Pecina, Jaroslav Pokorný. "Further Steps towards a Standard Testbed for Optical Music Recognition", 17th International Society for Music Information Retrieval Conference, 2016.

ogy that enables iteratively improving, refining and fine-tuning automated OMR evaluation metrics.
- A corpus of 1230 human preference judgments as gold-standard data for this methodology, and assessments of example MusicXML evaluation metrics against this corpus.
- Definitions of ground truths that can be applied to Common Western Music Notation (CWMN) scores.
- MUSCIMA++. A prototype benchmark with multiple levels of ground truth that extends a subset of the CVC-MUSCIMA dataset [9], with 3191 annotated notation primitives.

The rest of this paper is organized as follows: in Sec. 2, we review the state-of-the-art on OMR evaluation and datasets; in Sec. 3, we describe the human judgment data for developing automated evaluation metrics and demonstrate how it can help metric development. In Sec. 4, we present the prototype benchmark and finally, in Sec. 5, we summarize our findings and suggest further steps to take. [4]

2. RELATED WORK

The problem of evaluating OMR and creating a standard benchmark has been discussed before [7, 10, 16, 18, 20] and it has been argued that evaluating OMR is a problem as difficult as OMR itself. Jones et al. [10] suggest that in order to automatically measure and evaluate the performance of OMR systems, we need (a) a standard dataset and standard terminology, (b) a definition of a set of rules and metrics, and (c) definitions of different ratios for each kind of errors. The authors noted that distributors of commercial OMR software often claim the accuracy of their system is about 90 %, but provide no information about how that value was estimated.

Bellini et al. [18] manually assess results of OMR systems at two levels of symbol recognition: low-level, where only the presence and positioning of a symbol is assessed, and high-level, where the semantic aspects such as pitch and duration are evaluated as well. At the former level, mistaking a beamed group of 32nds for 16ths is a minor error; at the latter it is much more serious. They defined a detailed set of rules for counting symbols as recognized, missed and confused symbols. The symbol set used in [18] is quite rich: 56 symbols. They also define *recognition gain*, based on the idea that an OMR system is at its best when it minimizes the time needed for correction as opposed to transcribing from scratch, and stress *verification cost*: how much it takes to verify whether an OMR output is correct.

An extensive theoretical contribution towards benchmarking OMR has been made recently by Byrd and Simonsen [7]. They review existing work on evaluating OMR systems and clearly formulate the main issues related to evaluation. They argue that the complexity of CWMN is the main reason why OMR is inevitably problematic, and suggest the following stratification into levels of difficulty:

1. Music on one staff, strictly monophonic,
2. Music on one staff, polyphonic,
3. Music on multiple staves, but each strictly monophonic, with no interaction between them,
4. "Pianoform": music on multiple staves, one or more having multiple voices, and with significant interaction between and/or within staves.

They provide 34 pages of sheet music that cover the various sources of difficulty. However, the data does not include handwritten music and no ground truth for this corpus is provided.

Automatically evaluating MusicXML has been attempted most significantly by Szwoch [20], who proposes a metric based on a top-down MusicXML node matching algorithm and reports agreement with human annotators, but how agreement was assessed is not made clear, no implementation of the metric is provided and the description of the evaluation metric itself is quite minimal. Due to the complex nature of MusicXML (e.g., the same score can be correctly represented by different MusicXML files), Szwoch also suggests a different representation may be better than comparing two MusicXML files directly.

More recently, evaluating OMR with MusicXML outputs has been done by Padilla et al. [21]. While they provide an implementation, there is no comparison against gold-standard data. (This is understandable, as the paper [21] is focused on recognition, not evaluation.) Aligning MusicXML files has also been explored by Knopke and Byrd [12] in a similar system-combination setting, although not for the purposes of evaluation. They however make an important observation: stems are often mistaken for barlines, so the obvious simplification of first aligning measures is not straightforward to make.

No publicly available OMR dataset has ground truth for end-to-end recognition. The CVC-MUSCIMA dataset for staffline identification and removal and writer identification by Fornés et al. [9] is most extensive, with 1000 handwritten scores (50 musicians copying a shared set of 20 scores) and a version with staves removed, which is promising for automatically applying ground truth annotations across the 50 versions of the same score. Fornés et al. [8] have also made available a dataset of 2128 clefs and 1970 accidentals.

The HOMUS musical symbol collection for online recognition [11] consists of 15200 samples (100 musicians, 32 symbol classes, 4-8 samples per class per musician) of individual handwritten musical symbols. The dataset can be used for both online and offline symbol classification.

A further dataset of 3222 handwritten and 2521 printed music symbols is available upon request [1]. Bellini et al. [18] use 7 selected images for their OMR assessment; unfortunately, they do not provide a clear description of the database and its ground truth, and no more information is publicly available. Another staffline removal dataset is Dalitz's database, [5] consisting of 32 music pages that cov-

[4] All our data, scripts and other supplementary materials are available at `https://github.com/ufal/omreval` as a git repository, in order to make it easier for others to contribute towards establishing a benchmark.

[5] `http://music-staves.sourceforge.net`

ers a wide range of music types (CWMN, lute tablature, chant, mensural notation) and music fonts. Dalitz et al. [6] define several types of distortion in order to test the robustness of the different staff removal algorithms, simulating both image degradation and page deformations. These have also been used to augment CVC-MUSCIMA.

There are also large sources such as the Mutopia project [6] with transcriptions to LilyPond and KernScores [7] with HumDrum. The IMSLP database [8] holds mostly printed scores, but manuscripts as well; however, as opposed to Mutopia and KernScores, IMSLP generally only provides PDF files and no transcription of their musical content, except for some MIDI recordings.

3. EVALUATING EVALUATION

OMR lacks an automated evaluation metric that could guide development and reduce the price of conducting evaluations. However, an automated metric for OMR evaluation needs itself to be evaluated: does it really rank as better systems that *should* be ranked better?

Assuming that the judgment of (qualified) annotators is considered the gold standard, the following methodology then can be used to assess an automated metric:

1. Collect a corpus of annotator judgments to define the expected gold-standard behavior,
2. Measure the agreement between a proposed metric and this gold standard.

This approach is inspired by machine translation (MT), a field where comparing outputs is also notoriously difficult: the WMT competition has an evaluation track [5, 14], where automated MT metrics are evaluated against human-collected evaluation results, and there is ongoing research [3, 15] to design a better metric than the current standards such as BLEU [17] or Meteor [13]. This methodology is nothing surprising; in principle, one could machine-learn a metric given enough gold-standard data. However: how to best design the gold-standard data and collection procedure, so that it encompasses what we in the end want our application (OMR) to do? How to measure the quality of such a corpus – given a collection of human judgments, how much of a gold standard is it?

In this section, we describe a data collection scheme for human judgments of OMR quality that should lead to comparing automated metrics.

3.1 Test case corpus

We collect a corpus C of *test cases*. Each test case $c_1 \dots c_N$ is a triplet of music scores: an "ideal" score I_i and two "mangled" versions, $P_i^{(1)}$ and $P_i^{(2)}$, which we call *system outputs*. We asked our K annotators $a_1 \dots a_K$ to choose the less mangled version, formalized as assigning $r_a(c_i) = -1$ if they preferred $P_i^{(1)}$ over $P_i^{(2)}$, and $+1$ for the opposite preference. The term we use is to "rank" the predictions. When assessing an evaluation metric against this corpus, the test case rankings then constrain the space of well-behaved metrics. [9]

The exact formulation of the question follows the "cost-to-correct" model of evaluation of [18]:

> "Which of the two system outputs would take you less effort to change to the ideal score?"

3.1.1 What is in the test case corpus?

We created 8 ideal scores and derived 34 "system outputs" from them by introducing a variety of mistakes in a notation editor. Creating the system outputs manually instead of using OMR outputs has the obvious disadvantage that the distribution of error types does not reflect the current OMR state-of-the-art. On the other hand, once OMR systems change, the distribution of corpus errors becomes obsolete anyway. Also, we create errors for which we can assume the annotators have a reasonably accurate estimate of their own correction speed, as opposed to OMR outputs that often contain strange and syntactically incorrect notation, such as isolated stems. Nevertheless, when more annotation manpower becomes available, the corpus should be extended with a set of actual OMR outputs.

The ideal scores (and thus the derived system outputs) range from a single whole note to a "pianoform" fragment or a multi-staff example. The distortions were crafted to cover errors on individual notes (wrong pitch, extra accidental, key signature or clef error, etc.: micro-errors on the semantic level in the sense of [16, 18]), systematic errors within the context of a full musical fragment (wrong beaming, swapping slurs for ties, confusing staccato dots for noteheads, etc.), short two-part examples to measure the tradeoff between large-scale layout mistakes and localized mistakes (e.g., a four-bar two-part segment, as a perfect concatenation of the two parts into one vs. in two parts, but with wrong notes) and longer examples that constrain the metric to behave sensibly at larger scales.

Each pair of system outputs derived from the same ideal score forms a test case; there are 82 in total. We also include 18 control examples, where one of the system outputs is identical to the ideal score. A total of 15 annotators participated in the annotation, of whom 13 completed all 100 examples; however, as the annotations were voluntary, only 2 completed the task twice for measuring intra-annotator agreement.

3.1.2 Collection Strategy

While Bellini et al. [18] define how to count individual errors at the level of musical symbols, assign some cost to each kind of error (miss, add, fault, etc.) and define the overall cost as composed of those individual costs, our methodology does not assume that the same type of error has the same cost in a different *context*, or that the overall cost can be computed from the individual costs: for instance, a sequence of notes shifted by one step can be in

[6] http://www.mutopiaproject.org
[7] http://humdrum.ccarh.org
[8] http://imslp.org

[9] We borrow the term "test case" from the software development practice of unit testing: test cases verify that the program (in our case the evaluation metric) behaves as expected on a set of inputs chosen to cover various standard and corner cases.

most editors corrected simultaneously (so, e.g., clef errors might not be too bad, because the entire part can be transposed together).

Two design decisions of the annotation task merit further explanation: why we ask annotators to compare examples instead of rating difficulty, and why we disallow equality.

Ranking. The practice of ranking or picking the best from a set of possible examples is inspired by machine translation: Callison-Burch et al. have shown that people are better able to agree on which proposed translation is better than on how good or bad individual translations are [4]. Furthermore, ranking does not require introducing a cost metric in the first place. Even a simple 1-2-3-4-5 scale has this problem: how much effort is a "1" on that scale? How long should the scale be? What would the relationship be between short and long examples?

Furthermore, this annotation scheme is fast-paced. The annotators were able to do all the 100 available comparisons within 1 hour. Rankings also make it straightforward to compare automated evaluation metrics that output values from different ranges: just count how often the metric agrees with gold-standard ranks using some measure of monotonicity, such as Spearman's rank correlation coefficient.

No equality. It is also not always clear which output would take less time to edit; some errors genuinely are equally bad (sharp vs. flat). These are also important constraints on evaluation metrics: the costs associated with each should not be too different from each other. However, allowing annotators to explicitly mark equality risks overuse, and annotators using *underqualified* judgment. For this first experiment, therefore, we elected not to grant that option; we then interpret disagreement as a sign of uncertainty and annotator uncertainty as a symptom of this genuine tie.

3.2 How gold is the standard?

All annotators ranked the control cases correctly, except for one instance. However, this only accounts for elementary annotator failure and does not give us a better idea of systematic error present in the experimental setup. In other words, we want to ask the question: if all annotators are performing to the best of their ability, **what level of uncertainty should be expected under the given annotation scheme?** (For the following measurements, the control cases have been excluded.)

Normally, inter-annotator agreement is measured: if the task is well-defined, i.e., if a gold standard *can* exist, the annotators will tend to agree with each other towards that standard. However, usual agreement metrics such as Cohen's κ or Krippendorf's α require computing *expected agreement*, which is difficult when we do have a subset of examples on which we do *not* expect annotators to agree but cannot *a priori* identify them. We therefore start by defining a simple agreement metric L. Recall:

- C stands for the corpus, which consists of N examples $c_1 \dots c_N$,
- A is the set of K annotators $a_1 \dots a_K$, $a, b \in A$;
- r_a is the *ranking function* of an annotator a that assigns +1 or -1 to each example in c,

$$L(a,b) = \frac{1}{N} \sum_{c \in C} \frac{|r_a(c) + r_b(c)|}{2}$$

This is simply the proportion of cases on which a and b agree: if they disagree, $r_a(c)+r_b(c) = 0$. However, we expect the annotators to disagree on the genuinely uncertain cases, so some disagreements are not as serious as others. To take the existence of legitimate disagreement into account, we modify $L(a,b)$ to weigh the examples according to how certain the other annotators $A \setminus \{a,b\}$ are about the given example. We define weighed agreement $L_w(a,b)$:

$$L_w(a,b) = \frac{1}{N} \sum_{c \in C} w^{(-a,b)}(c) \frac{|r_a(c) + r_b(c)|}{2}$$

where $w^{(-a,b)}$ is defined for an example c as:

$$w^{(-a,b)}(c) = \frac{1}{K-2} \left| \sum_{a' \in A \setminus a,b} r_{a'}(c) \right|$$

This way, it does not matter if a and b disagree on cases where no one else agrees either, but if they disagree on an example where there is strong consensus, it should bring the overall agreement down. Note that while maximum achievable $L(a,b)$ is 1 for perfectly agreeing annotators (i.e., all the sum terms equal to 1), because $w(c) \leq 1$, the maximum achievable $L_w(a,b)$ will be less than 1, and furthermore depends on the choice of a and b: if we take notoriously disagreeing annotators away from the picture, the weights will increase overall. Therefore, we finally adjust $L_w(a,b)$ to the proportion of maximum achievable $L_w(a,b)$ for the given (a,b) pair, which is almost the same as $L_w(a,a)$ with the exception that *b must also be excluded from computing the weights*. We denote this maximum as $L^*_w(a,b)$, and the adjusted metric $\hat{L}_w$ is then:

$$\hat{L}_w(a,b) = L_w(a,b) / L^*_w(a,b)$$

This metric says: "What proportion of achievable weighed agreement has been actually achieved?" The upper bound of $\hat{L}_w$ is therefore 1.0 again; the lower bound is agreement between two randomly generated annotators, with the humans providing the consensus.

The resulting pairwise agreements, with the lower bound established by averaging over 10 random annotators, are visualized in Fig. 1. The baseline agreement $\hat{L}_w$ between random annotators weighed by the full human consensus was close to 0.5, as expected. There seems to be one group of annotators relatively in agreement (green and above, which means adjusted agreement over 0.8), and then several individuals who disagree with everyone – including among themselves (lines 6, 7, 8, 11, 12, 14).

Interestingly, most of these "lone wolves" reported significant experience with notation editors, while the group more in agreement not as much. We suspect this is because

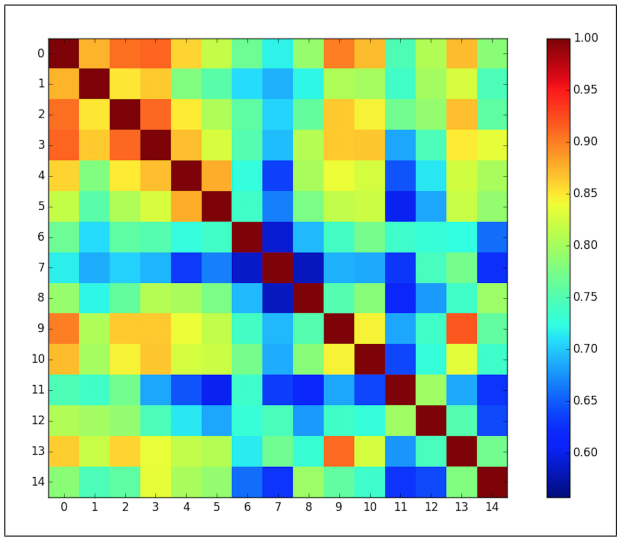


Figure 1. Weighed pairwise agreement. The cell $[a, b]$ represents $\hat{L}_w(a, b)$. The scale goes from the average random agreement (ca. 0.55) up to 1.

with increasing notation editor experience, users develop a personal editing style that makes certain actions easier than others by learning a *subset* of the "tricks" available with the given editing tools – but each user learns a different subset, so agreement on the relative editing cost suffers. To the contrary, inexperienced users might not have spent enough time with the editor to develop these habits.

3.3 Assessing some metrics

We illustrate how the test case ranking methodology helps analyze these rather trivial automated MusicXML evaluation metrics:

1. Levenshtein distance of XML canonization (**c14n**),
2. Tree edit distance (**TED**),
3. Tree edit distance with `<note>` flattening (**TEDn**),
4. Convert to LilyPond + Levenshtein distance (**Ly**).

c14n. Canonize the MusicXML file formatting and measure Levenshtein distance. This is used as a trivial baseline.

TED. Measure Tree Edit Distance on the MusicXML nodes. Some nodes that control auxiliary and MIDI information (`work`, `defaults`, `credit`, and `duration`) are ignored. Replacement, insertion, and deletion all have a cost of 1.

TEDn. Tree Edit Distance with special handling of `note` elements. We noticed that many errors of TED are due to the fact that while deleting a note is easy in an editor, the edit distance is higher because the `note` element has many sub-nodes. We therefore encode the notes into strings consisting of one position per `pitch`, `stem`, `voice`, and `type`. Deletion cost is fixed at 1, insertion cost is 1 for non-note nodes, and 1 + length of code for notes. Replacement cost between notes is the edit distance between their codes; replacement between a note and non-note costs 1 + length of code; between non-notes costs 1.

Metric	r_s	$\hat{r}_s$	ρ	$\hat{\rho}$	τ	$\hat{\tau}$
c14n	0.33	0.41	0.40	0.49	0.25	0.36
TED	0.46	0.58	0.40	0.50	0.35	0.51
TEDn	**0.57**	**0.70**	0.40	0.49	**0.43**	**0.63**
Ly	0.41	0.51	0.29	0.36	0.30	0.44

Table 1. Measures of agreement for some proposed evaluation metrics.

Ly. The LilyPond [10] file format is another possible representation of a musical score. It encodes music scores in its own LaTeX-like language. The first bar of the "Twinkle, twinkle" melody would be represented as `d'8[ d'8] a'8[ a'8] b'8[ b'8] a'4 |` This representation is much more amenable to string edit distance. The **Ly** metric is Levenshtein distance on the LilyPond import of the MusicXML system output files, with all whitespace normalized.

For comparing the metrics against our gold-standard data, we use nonparametric approaches such as Spearman's r_s and Kendall's τ, as these evaluate monotonicity without assuming anything about mapping values of the evaluation metric to the $[-1, 1]$ range of preferences . To reflect the "small-difference-for-uncertain-cases" requirement, however, we use Pearson's ρ as well [14]. For each way of assessing a metric, its maximum achievable with the given data should be also estimated, by computing how the metric evaluates the consensus of one group of annotators against another. We randomly choose 100 splits of 8 vs 7 annotators, compute the average preferences for the two groups in a split and measure the correlations between the average preferences. The expected upper bounds and standard deviations estimated this way are:

- $r_s^* = 0.814$, with standard dev. 0.040
- $\rho^* = 0.816$, with standard dev. 0.040
- $\tau^* = 0.69$, with standard dev. 0.045

We then define $\hat{r_s}$ as $\frac{r_s}{r_s^*}$, etc. Given a cost metric $\mathcal{L}$, we get for each example $c_i = (I_i, P_i^{(1)}, P_i^{(2)})$ the cost difference $\ell(c_i) = \mathcal{L}(I_i, P_i^{(1)}) - \mathcal{L}(I_i, P_i^{(2)})$ and pair it with the gold-standard consensus $r(c_i)$ to get pairwise inputs for the agreement metrics.

The agreement of the individual metrics is summarized in Table 1. When developing the metrics, we did *not* use the gold-standard data against which metric performance is measured here; we used only our own intuition about how the test cases should come out.

4. BENCHMARK DATASET PROTOTYPE

A benchmark dataset should have ground truth at levels corresponding to the standard OMR processing stages, so that sub-systems such as staff removal, or symbol localization can be compared with respect to the end-to-end pipeline they are a part of. We also suspect handwritten music will remain an open problem much longer than printed music. Therefore, we chose to extend the **CVC-MUSCIMA** dataset instead of Byrd and Simonsen's pro-

[10] `http://www.lilypond.org`

posed test bed [7] because of the extensive handwritten data collection effort that has been completed by Fornés et al. and because ground truth for staff removal and binarization is already present. At the same time, CVC-MUSCIMA covers all the levels of notational complexity from [7], as well as a variety of notation symbols, including complex tuples, less common time signatures (5/4), C-clefs and some symbols that could very well expose the differences between purely symbol-based and more syntax-aware methods (e.g., tremolo marks, easily confused for beams). We have currently annotated symbols in printed scores only, with the perspective of annotating the handwritten scores automatically or semi-automatically.

We selected a subset of scores that covers the various levels of notational complexity: single-part monophonic music (F01), multi-part monophonic music (F03, F16), and pianoform music, primarily based on chords (F10) and polyphony (F08), with interaction between staves.

4.1 Symbol-level ground truth

Symbols are represented as bounding boxes, labeled by symbol class. In line with the low-level and high-level symbols discussed by [7], we differentiate symbols at the level of *primitives* and the level of *signs*. The relationship between primitives and signs can be one-to-one (e.g., clefs), many-to-one (composite signs: e.g. notehead, stem, and flag form a note), one-to-many (disambiguation: e.g., a sharp primitive can be part of a key signature, accidental, or an ornament accidental), and many-to-many (the same beam participates in multiple beamed notes, but each beamed note also has a stem and notehead). We include individual numerals and letters as notation primitives, and their disambiguation (tuplet, time signature, dynamics...) as signs.

We currently define 52 primitives plus letters and numerals, and 53 signs. Each symbol can be linked to a MusicXML counterpart. [11] There are several groups of symbols:

- Note elements (noteheads, stems, beams, rests...)
- Notation elements (slurs, dots, ornaments...)
- Part default (clefs, time and key signatures...)
- Layout elements (staves, brackets, braces...)
- Numerals and text.

We have so far annotated the primitive level. There are 3191 primitives marked in the 5 scores. Annotation took about 24 hours of work in a custom editor.

4.2 End-to-end ground truth

We use MusicXML as the target representation, as it is supported by most OMR/notation software, actively maintained and developed and available under a sufficiently permissive license. We obtain the MusicXML data by manually transcribing the music and postprocessing to ensure each symbol has a MusicXML equivalent. Postprocessing mostly consists of filling in default barlines and correcting staff grouping information. Using the MuseScore notation editor, transcription took about 3.5 hours.

[11] The full lists of symbol classes are available in the repository at `https://github.com/ufal/omreval` under `muscima++/data/Symbolic/specification`.

5. CONCLUSIONS AND FUTURE WORK

We proposed a corpus-based approach to assessing automated end-to-end OMR evaluation metrics and illustrated the methodology on several potential metrics. A gold standard annotation scheme based on assessment of relative cost-to-correct of synthetic "system outputs" was described that avoids pre-defining any cost metric, and the resulting corpus of 1230 human judgments was analyzed for inter-annotator agreement, taking into account the possibility that the compared system outputs may not be clearly comparable. This preference-based setup avoids the need to pre-define any notion of cost, requires little annotator training, and it is straightforward to assess an evaluation metric against this preference data.

Our results suggest that the central assumption of a single ground truth for preferences among a set of system outputs is weaker with increasing annotator experience. To make the methodology more robust, we recommend:

- Explicitly control for experience level; do not assume that more annotator experience is better.
- Measure actual cost-to-correct (in time and interface operations) through a notation editor, to verify how much human *estimation* of this cost can be relied on.
- Develop models for computing expected agreement for data where the annotations may legitimately be randomized (the "equally bad" cases). Once expected agreement can be computed, we can use more standard agreement metrics.

The usefulness of the test case corpus for developing automated evaluation metrics was clear: the TEDn metric that outperformed the others by a large margin was developed through analyzing the shortcomings of the TED metric on individual test cases (before the gold-standard data had been collected). As Szwoch [20] suggested, modifying the representation helped. However, if enough human judgments are collected, it should even be possible to sidestep the difficulties of hand-crafting an evaluation metric through machine learning; we can for instance try learning the insertion, deletion, and replacement costs for individual MusicXML node types.

An OMR environment where different systems can be meaningfully compared, claims of commercial vendors are verifiable and progress can be measured is in the best interest of the OMR community. We believe our work, both on evaluation and on a dataset, constitutes a significant step in this direction.

6. ACKNOWLEDGMENTS

This research is supported by the Czech Science Foundation, grant number P103/12/G084. We would also like to thank our annotators from the Janáček Academy of Music and Performing Arts in Brno and elsewhere, and Alicia Fornés for providing additional background and material for the CVC-MUSCIMA dataset.

7. REFERENCES

[1] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre R. S. Marcal, Carlos Guedes, and Jaime S. Cardoso. Optical Music Recognition: State-of-the-Art and Open Issues. *Int J Multimed Info Retr*, 1(3):173–190, Mar 2012.

[2] Baoguang Shi, Xiang Bai, and Cong Yao. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. *CoRR*, abs/1507.05717, 2015.

[3] Ondřej Bojar, Miloš Ercegovčević, Martin Popel, and Omar F. Zaidan. A Grain of Salt for the WMT Manual Evaluation. *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 1–11, 2011.

[4] Chris Callison Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) Evaluation of Machine Translation. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, 2007.

[5] Chris Callison Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar F. Zaidan. Findings of the 2010 Joint Workshop on Statistical Machine Translation and Metrics for Machine Translation. *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53, 2010.

[6] Christoph Dalitz, Michael Droettboom, Bastian Pranzas, and Ichiro Fujinaga. A Comparative Study of Staff Removal Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):753–766, May 2008.

[7] Donald Byrd and Jakob Grue Simonsen. Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images. *Journal of New Music Research*, 44(3):169–195, 2015.

[8] Alicia Fornés, Josep Lladós, Gemma Sánchez, and Horst Bunke. Writer Identification in Old Handwritten Music Scores. *Proceeedings of Eighth IAPR International Workshop on Document Analysis Systems*, pages 347–353, 2008.

[9] Alicia Fornés, Anjan Dutta, Albert Gordo, and Josep Lladós. CVC-MUSCIMA: a ground truth of handwritten music score images for writer identification and staff removal. *International Journal on Document Analysis and Recognition (IJDAR)*, 15(3):243–251, 2012.

[10] Graham Jones, Bee Ong, Ivan Bruno, and Kia Ng. Optical Music Imaging: Music Document Digitisation, Recognition, Evaluation, and Restoration. *Interactive Multimedia Music Technologies*, pages 50–79, 2008.

[11] Jorge Calvo-Zaragoza and Jose Oncina. Recognition of Pen-Based Music Notation: The HOMUS Dataset. *22nd International Conference on Pattern Recognition*, Aug 2014.

[12] Ian Knopke and Donald Byrd. Towards Musicdiff : A Foundation for Improved Optical Music Recognition Using Multiple Recognizers. *International Society for Music Information Retrieval Conference*, 2007.

[13] Alon Lavie and Abhaya Agarwal. Meteor: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, 2007.

[14] Matouš Macháček and Ondřej Bojar. Results of the WMT14 Metrics Shared Task. *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 293–301, 2014.

[15] Matouš Macháček and Ondřej Bojar. Evaluating Machine Translation Quality Using Short Segments Annotations. *The Prague Bulletin of Mathematical Linguistics*, 103(1), Jan 2015.

[16] Michael Droettboom and Ichiro Fujinaga. Microlevel groundtruthing environment for OMR. *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004)*, pages 497–500, 2004.

[17] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, 2002.

[18] Pierfrancesco Bellini, Ivan Bruno, and Paolo Nesi. Assessing Optical Music Recognition Tools. *Computer Music Journal*, 31(1):68–93, Mar 2007.

[19] Craig Sapp. OMR Comparison of SmartScore and SharpEye. `https://ccrma.stanford.edu/~craig/mro-compare-beethoven`, 2013.

[20] Mariusz Szwoch. Using MusicXML to Evaluate Accuracy of OMR Systems. *Proceedings of the 5th International Conference on Diagrammatic Representation and Inference*, pages 419–422, 2008.

[21] Victor Padilla, Alan Marsden, Alex McLean, and Kia Ng. Improving OMR for Digital Music Libraries with Multiple Recognisers and Multiple Sources. *Proceedings of the 1st International Workshop on Digital Libraries for Musicology - DLfM '14*, pages 1–8, 2014.

IMPROVING VOICE SEPARATION BY BETTER CONNECTING CONTIGS

Nicolas Guiomard-Kagan[1] **Mathieu Giraud**[2] **Richard Groult**[1] **Florence Levé**[1,2]
[1] MIS, Univ. Picardie Jules Verne, Amiens, France [2] CRIStAL, UMR CNRS 9189, Univ. Lille, Lille, France
{nicolas,mathieu,richard,florence}@algomus.fr

ABSTRACT

Separating a polyphonic symbolic score into monophonic voices or streams helps to understand the music and may simplify further pattern matching. One of the best ways to compute this separation, as proposed by Chew and Wu in 2005 [2], is to first identify *contigs* that are portions of the music score with a constant number of voices, then to progressively *connect* these contigs. This raises two questions: Which contigs should be connected first? And, how should these two contigs be connected? Here we propose to answer simultaneously these two questions by considering a set of musical features that measures the quality of any connection. The coefficients weighting the features are optimized through a genetic algorithm. We benchmark the resulting connection policy on corpora containing fugues of the *Well-Tempered Clavier* by J. S. Bach as well as on string quartets, and we compare it against previously proposed policies [2, 9]. The contig connection is improved, particularly when one takes into account the whole content of voice fragments to assess the quality of their possible connection.

1. INTRODUCTION

Polyphony, as opposed to monophony, is music created by simultaneous notes coming from several instruments or even from a single polyphonic instrument, such as the piano or the guitar. Polyphony usually implies chords and harmony, and sometimes counterpoint when the melody lines are independent.

Voice separating algorithms group notes from a polyphony into individual voices [2, 4, 9, 11, 13, 15]. These algorithms are often based on perceptive rules, as studied by Huron [7] or Deutsch [5, chapter 2], and at the first place *pitch proximity* – voices tend to have small intervals.

Separating polyphony into voices is not always possible or meaningful: many textures for polyphonic instruments include chords with a variable number of notes. Conversely, one can play several streams on a monophonic instrument. *Stream* separation algorithms focus thus on a narrower scale, extracting groups of coherent notes. These segments are not necessarily connected throughout the whole score: a voice can be split into several streams and a stream can cluster notes from different voices [14, 16].

Both voice and stream segmentation algorithms provide a better understanding of polyphony and make inference and matching for relevant patterns easier. We previously showed that voice and stream separation algorithms are two facets of the same problem that can be compared with similar evaluation metrics [6]. Pertinent evaluation metrics measure how segments or voices of the ground truth are grouped together in the algorithms predictions, as the transition-based evaluation [2] or the measure of mutual information [6, 12].

Based on these metrics, it appears that the contig approach, as initially proposed by Chew and Wu [2] (Section 2), is one of the best approaches to separate voices, starting from *contigs* having a constant number of voices. The results depends on how the contigs are *connected*, larger voice or stream segments being built starting from smaller ones.

In this article we propose and compare several criteria to ground the *connection policy*, that is both the choice of the order of the contigs to be connected, and the connection itself between contigs. In addition to the criteria used in the literacy, we introduce new criteria that take into account *more musical context*, averaging pitches and durations over voice fragments (Section 3). We weight these criteria using a genetic algorithm (Section 4). We show how some values of these criteria can partially simulate the previous methods, and evaluate the results on sets of fugues and string quartets. By improving this contig connection, we improve the precision of voice separation algorithms (Section 5). We further study the distribution of failures, showing that a higher precision can be obtained by stopping the contig connection before the connection quality drops.

2. VOICE SEPARATION BASED ON CONTIGS

The contig approach, proposed by Chew and Wu (denoted by CW in the following) first separates the music score into contigs that have a constant number of notes played at the same time then progressively connect these contigs to the whole score [2].

The first step splits the input polyphonic data into blocks called *contigs* such that the number of simultaneous notes in a contig does not change (Figure 1). Notes cross-

© Nicolas Guiomard-Kagan, Mathieu Giraud, Richard Groult, Florence Levé. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Nicolas Guiomard-Kagan, Mathieu Giraud, Richard Groult, Florence Levé. "Improving voice separation by better connecting contigs", 17th International Society for Music Information Retrieval Conference, 2016.

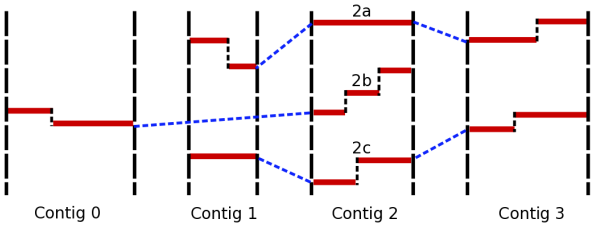


Figure 1. In this piano-roll symbolic representation, each segment describes a note. The horizontal axis represents time and the vertical axis represents pitches. The notes can be grouped in four *contigs*, each of them containing a constant number of notes played at the same time. Contig 2 contains three *voice fragments* $2a$, $2b$ and $2c$. The challenge of contig-based voice separation algorithms is to connect these voice fragments across contigs to build coherent voices throughout the score. The non-vertical dotted lines show a possible solution of the voice separation.

ing the border of several contigs are split in several notes. The idea behing building contigs is that the voice separation is relatively easy inside them: Notes in each contig are grouped by pitch height to form *voice fragments*.

The second step links together fragments from distinct contigs, following some musical principles (Figure 2). The algorithm has now to take two kinds of decisions, following what we call a *connection policy*:

- *which* contigs should be connected first?
- *how* should these two contigs be connected?

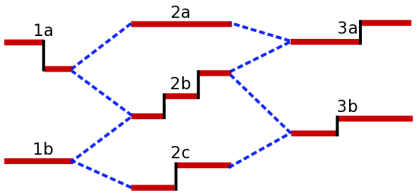


Figure 2. Any connection policy should decide which contigs should be connected (such as, for example, 1 and 2) and how to do this connection. There are here three possible connections (without voice crossing) between the contigs 1 and 2: $C_1 = \{(1a, 2a), (1b, 2b)\}$, $C_2 = \{(1a, 2a), (1b, 2c)\}$, and $C_3 = \{(1a, 2b), (1b, 2c)\}$.

Order of connection of contigs. In CW algorithm, the connection starts from the *maximal contigs* (i.e. contigs containing the maximal number of voices). Since the voices tend not to cross, the voice separation and connection in these contigs with many voices were thought to be more reliable. Then, CW continues the connection process to the left and to the right of these maximal contigs. In Figure 1, the CW policy will thus connect contigs $1, 2, 3$, then finally $0, 1, 2, 3$.

Ishigaki, Matsubara and Saito (denoted IMS in the following) suggested another connection policy, starting with minimal contigs and connecting contigs with an *increasing* number of fragments (i.e. the number of fragments in the left contig is lower or equal to the number of fragments in the right contig) [9]. The idea is that the (local) start of a new voice is a more perceptible event than the (local) end of a voice. Once all those possible connections are done, maximal contigs are considered as in CW algorithm to terminate the process. In Figure 1, IMS policy will connect contigs $0, 1$, then $0, 1, 2$, and finally $0, 1, 2, 3$.

Fragment connection. The policy to connect fragments of the original CW algorithm, reused by IMS, is based on two principles: Intervals are minimized between successive notes in the same stream or voice (pitch proximity); Voices tend not to cross. Formally, the connection between two contigs is a set of (ℓ, r) fragments that maximize a *connection score*. This score is here based on the absolute difference between the pitch of the last note of the left fragment ℓ and the pitch of the first note of the right fragment r. There is moreover a very large score for the connection of notes split between two contigs to keep them in the same final voice.

3. MORE MUSICAL FEATURES TO IMPROVE THE CONNECTION POLICY

3.1 A new view on the contig-based approach

We argue that the two questions of the connection policy (*which* contigs should be connected? *how* to connect them?) should be handled at a same time: to build coherent voices across a piece, one should always connect the contigs yielding the "safest" connections between voice fragments. The quality of these connections should be properly evaluated with musical features that will be introduced below.

Given two successive contigs i and $i + 1$, and one way C to connect them (set of pairs of fragments), we define a *connection score* $S(i, C)$, computed as a weighted sum of musical features, that measures the quality of this connection: The higher the connection score, the safer the connection. The connection scores will extend the ones used by CW and IMS, that did not systematically explore the relation between the two decisions of the connection policy.

At each step of the algorithm, the (i, C) maximizing S is selected, giving both the "best contigs" to connect and the "best way" to connect them. Once this connection is made, the connections scores between the newly formed contig and its left and right neighbors have to be computed.

Definitions. Let n be the maximal number of simultaneous notes in the piece. Let n_i (respectively n_{i+1}) be the maximal number of voices of the contig i ($i + 1$). After some connections have been made, a contig may have a different number of simultaneous notes at its both extremities, but the hanging voices are "projected" to these extremities.

For two successive contigs i and $i + 1$, let C be a set of pairs (ℓ, r), where ℓ is a fragment of the (left) contig i and r a fragment of the (right) contig $i + 1$, each

fragment appearing at most once in C (Figure 2). C has thus at most $m = \min(n_i, n_{i+1})$ elements, and, in the following, we only consider sets with m elements, that is with the highest possible number of connections. Denoting $M = \max(n_i, n_{i+1})$, there are $M!/(M-m)!$ different such combinations for C, and only $\binom{M}{m}$ if one restricts to the combinations without voice crossing.

We consider that we have N features $f_1(i, C), f_2(i, C) \dots f_N(i, C)$ characterizing some musical properties of the connection C between contigs i and $i + 1$. Each feature $f_k(i, C)$ has a value between 0 and 1. Finally let $\alpha_1, \alpha_2, \dots, \alpha_N$ be N coefficients such that $\sum_{k=1}^{N} \alpha_k = 1$. We then define the connection score as a linear combination of the features $S(i, C) = \sum_{k=1}^{N} \alpha_k f_k(i, C)$.

In the two following paragraphs, we propose different features $f_k(i, C)$ depending on the musical properties of contigs and fragments. The values of the coefficients α_k will be discussed in Section 4.

3.2 Features on the contigs

First we consider features that are not related to the connection C but depend only on the contigs, more precisely on the maximum number of voices in each contig.

- *maximal_voices(i)* $= \max(n_i, n_{i+1})/n$. The closer the number of voices to the maximal number of voices, the higher the connection score .
- *minimal_voices(i)* $= (n+1-\min(n_i, n_{i+1}))/n$. The closer the number of voices to 1, the higher the connection score.

One can in particular favor some contig connection based on the comparison of the number of voices between the left and the right contigs:

- *difference_nb_voices(i)* $= 1-(|n_i - n_{i+1}|/(n-1))$. The closer the number of voices of the left and the right contigs, the higher the connection score.

Or with the following binary features, that will equal 0 if the condition is not met:

- *increase(i)* $= 1$ iff $n_i < n_{i+1}$;
- *increase_one(i)* $= 1$ iff $n_i + 1 = n_{i+1}$;
- *increase_equal(i)* $= 1$ iff $n_i \leq n_{i+1}$;
- *decrease(i)* $= 1$ iff $n_i > n_{i+1}$;
- *decrease_one(i)* $= 1$ iff $n_i - 1 = n_{i+1}$;
- *decrease_equal(i)* $= 1$ iff $n_i \geq n_{i+1}$.

Those features are inspired by the connection policy of the existing algorithms. The *maximal_voices(i)* feature reflects the idea used by the CW algorithm: It is safer to first connect contigs having a large number of voices. The reverse idea, as measured by *minimal_voices(i)*, was proposed together with the *increase(i)* idea by the IMS algorithm, favoring the connection of contigs with an increasing number of voices. The idea is that the (local) start of a new voice is a more perceptible event than its (local) end. This is even more remarkable in contrapuntal music such as fugues where enterings of voice on thematic patterns (subjects, counter-subjects) are often clearly heard.

We propose to further use the *increase_one(i)* feature that should better assert an entry of exactly *one* new voice. Conversely, we also evaluate the opposite idea (*decrease(i)*, *decrease_one(i)*, *decrease_equal(i)*).

Finally the connection could favor successive contigs sharing a same note:

- *maximal_sim_notes(i)* $= n^{=}/\min(n_i, n_{i+1})$, where $n^{=}$ is the number of notes with the same pitch and same onset (i.e. note split in two) at the extremities of contigs i and $i + 1$. The more the contigs share common notes, the higher the connection score is.

This feature derives from the original implementation of CW, where connectig contigs with shared notes was awarded a very large score.

3.3 Features on the fragments

Now we consider features based on the individual fragment connections (ℓ, r) composing C.

Pitches. How can we measure the quality of connecting a fragment ℓ to a fragment r? The main criterion of the CW and IMS algorithms was to follow the pitch proximity principle, favoring connections of fragments having a small pitch interval. Given C and $(\ell, r) \in C$, let *last_pitch*(ℓ) and *first_pitch*(r) be the pitches of the extreme note of the left fragment ℓ and the right fragment r:

- *extreme_pitch(C)* $= 1 - \sum_{(\ell,r)\in C} |$*last_pitch*$(\ell) -$ *first_pitch*$(r)|/\nu$. The closer the pitches between the connected notes, the higher the connection score.

The normalization factor $\nu = 60 \cdot |C|$ semitones was chosen in order to range the feature value between 0 (5 octaves between connected pitches) and 1 (equal pitches). However, this *extreme_pitch(C)* score only considers one note on each side. We propose to extend this feature by evaluating the *pitch range coherence*, taking into account the average pitch (*average_pitch*) of *all notes* of one or both fragments. Indeed, voices tend to have the same pitch range throughout the piece, and moreover through the fragments:

- *avg_pitch_right(C)* $= 1 - \sum_{(\ell,r)\in C} |$*last_pitch*$(\ell) -$ *average_pitch*$(r)|/\nu$;
- *avg_pitch_left(C)* $= 1 - \sum_{(\ell,r)\in C} |$*average_pitch*$(\ell) -$ *last_pitch*$(r)|/\nu$;
- *avg_pitch(C)* $= 1 - \sum_{(\ell,r)\in C} |$*average_pitch*$(\ell) -$ *average_pitch*$(r)|/\nu$.

Some voice separation algorithms assign each note to the voice with the closest average pitch [10]. These algorithms are quite efficient, and the *avg_pitch(C)* feature reproduces this idea at a local scale: Given a fragment with

a few notes, even if one may not know to which (global) voice it belongs, one already knows a local pitch range.

Durations. Similarly, we can measure the difference of durations to favor connection of contiguous fragments with a same rhythm. Indeed, the musical textures of each voice tend to have coherent rhythms. For instance, a voice in whole notes and another one in eights will often be heard as two separate voices, even if they use very close pitches. Given C and $(\ell, r) \in C$, let *last_dur*(ℓ) and *first_dur*(r) be the durations, taken in a log scale, of the extreme notes of the left fragment ℓ and the right fragment r:

- *extreme_dur(C)* $= 1 - (\sum_{(\ell,r)\in C} |$*last_dur*$(\ell) -$ *first_dur*$(r)|/\lambda)$. The closer the durations between the connected notes, the higher the connection score.

The normalization factor $\lambda = 6 \cdot |C|$ accounts for the maximal difference (in a log scale) between whole notes (6) and 64th notes, the shortest notes in our corpora (0). Once more, this feature can also be extended to take into account the average log duration (*average_dur*) of one or both fragments instead of the duration of the extreme note:

- *avg_dur_right(C)* $= 1 - \sum_{(\ell,r)\in C} |$*last_dur*$(\ell) -$ *average_dur*$(r)|/\lambda$;
- *avg_dur_left(C)* $= 1 - \sum_{(\ell,r)\in C} |$*average_dur*$(\ell) -$ *last_dur*$(r)|/\lambda$;
- *avg_dur(C)* $= 1 - \sum_{(\ell,r)\in C} |$*average_dur*$(\ell) -$ *average_dur*$(r)|/\lambda$.

These features measure how a fragment may be "mostly in eights" or "mostly in long notes", even if it contains other durations as for ending notes. They handle also rhythmic patterns: a fragment repeating the pattern "one quarter, two eights" has an *average_dur* of about $3 + 1/3$.

Voice crossings. Finally, two features control the voice crossing. On one hand, voice crossings do exist, on the other hand, they are hard to predict. Voice separation algorithms (such as CW and IMS) usually prevent them.

- *crossed_voices(C)* $= 1$ if C contains a crossing voice, and 0 otherwise;
- *no_crossed_voices(C)* $= 1$ if C does not contain a crossing voice, and 0 otherwise.

4. LEARNING COEFFICIENTS THROUGH A GENETIC ALGORITHM

The selection of features coefficients $\alpha = (\alpha_1, \alpha_2, \ldots \alpha_N)$ was achieved with a genetic algorithm with mutation and crossover operators [1]. For computation efficiency, a generation is a set of 60 solutions, each solution being a set of coefficients totaling 1. The first generation G_0 is a set of solutions drawn with random values. The following generations are built through mutations and crossovers.

Mutation. Given a generation G_t, each solution is mutated 4 times, giving 4×60 mutated solutions. Each mutation consists in randomly transferring a part of the value of a randomly chosen coefficient into another one. A new set of 40 solutions is selected from both the original solutions and the mutated solutions, by taking the 30 best solutions and 10 random other solutions.

Crossover. The solutions in this set are then used to generate 20 children solutions by taking random couples of parents. Each parent is taken only once, and a child solution is the average of the coefficients of the parent solutions. The new generation G_{t+1} is formed by the 40 parents and the 20 children solutions.

5. RESULTS

We trained the coefficients weighting the features with the genetic algorithm on the 24 fugues in the first book of the *Well-Tempered Clavier* by J. S. Bach (corpus "wtc-i"). This gives the set of coefficients GA1 after 36 generations (the process stabilized after that). We then evaluated these GA1 coefficients and other connection policies on the 24 fugues of the second book of the *Well-Tempered Clavier* (corpus "wtc-ii") and on 17 first movements of classical and romantic string quartets (Haydn op. 33-1 to 33-6, op. 54-3, op. 64-4, Mozart K80, K155, K156, K157 and K387, Beethoven op. 18-2, Brahms op. 51-1 and Schubert op. 125-1). Our implementation is based on the Python framework music21 [3], and we worked on `.krn` files downloaded from `kern.ccarh.org` [8]. The explicit voice separation coming from the *spines* of these files forms the ground truth on which the algorithms are trained and evaluated.

5.1 Learned coefficients

The column GA1 of Table 1 shows the learned coefficients of the best solution. The high *no_crossed_voices(C)* coefficient confirms that trying to predict crossing voices currently gives many false connections. It may suggest that such detection should be avoided until specific algorithms could handle these cases. We draw two other observations:

- The pitch is the most important feature (the four *pitch* coefficients totaling 0.271). However, *avg_pitch_right(C)* is higher than *extreme_pitch(C)* – and summing *avg_pitch_left(C)*, *avg_pitch_right(C)* and *avg_pitch(C)* gives 0.181, twice *extreme_pitch(C)*. This confirms that using the pitch range coherence is more reliable than using the pitch proximity alone;
- The durations are also important features, especially when one takes the average durations (*avg_dur(C)* or *avg_dur_right(C)*, totaling 0.121). Note that the *extreme_dur(C)* coefficient is very low, confirming the idea that even if the individual durations change, rhythmic textures or small-scale patterns are conserved inside voice fragments.

Finally, the *increase_equal(i)* feature as suggested by IMS is high, but, surprisingly, the *decrease_equal(i)* feature is also high. These two features combined seem to underline that the contig connection is safer when both fragments have the same number of notes. Further experiments should be made to explore these features.

5.2 Quality of the connection policy

Evaluation metrics. The *transition recall (TR-rec)* (or *completeness*) is the ratio of correctly assigned transitions (pair of notes in the same voice) over the number of transitions in the ground truth. The *transition precision (TR-prec)* (or *soundness*) is the ratio of correctly assigned transitions over the number of transitions in the predicted voices [2,6,11]. The TR-rec and TR-prec metrics are equal for voice separation algorithms connecting voices throughout all the piece. Stream segmentation algorithms usually lead to higher TR-prec values as they predict fewer transitions. The ground truth and the output of the algorithms can also be considered as an assignation of a label to every note, enabling to compute the S_o *and* S_u *metrics* based on normalized entropies $H(\text{output}|\text{truth})$ and $H(\text{truth}|\text{output})$. These scores report how an algorithm may over-segment (S_o) or under-segment (S_u) a piece [6, 12]. They measure whether the clusters are coherent, even when streams cluster simultaneous notes. Moreover, we point out the *contig connection correctness (CC)*, that is the ratio of correct connections over all connections done.

Results. Table 2 details the evaluation metrics on the training set and the evaluation sets, both for the GA1 coefficients and for coefficients SimCW and SimIMS simulating the CW and IMS policies, displayed on Table 1. The metrics reported here may be slightly different from the results reported in the original CW and IMS implementations [2, 9]. The goal of our evaluation is to evaluate connection policies inside a same implementation. On all corpora, the GA1 coefficients obtain better TR-prec/TR-rec/CC results than the SimCW and SimIMS coefficients. The GA1 coefficients indeed make better connections (more than 87% of correct connections on the test corpus "wtc-ii"). The main source of improvement comes from the new features that consider the average pitches and/or lengths, as showed by the example on Figure 3.

5.3 Lowering the failures by stopping the connections

The first step of CW, the creation of contigs, is very reliable: TR-prec is more than 99% on both fugues corpora (lines "no connection" in Table 2). Most errors come from the connection steps. We studied the distribution of these errors. With the SimIMS coefficients, and even more with the GA1 coefficients, the first connections are generally reliable, more errors being done in the last connections (Figure 4). This confirms that considering more musical features improves the connections.

By stopping the algorithm with the GA1 coefficients when 75% of the connections have been done, almost half of the bad connections are avoided, giving streams with a good compromise between precision and consistency (lines "GA1-75%" in Table 2).

Feature	GA1	SimCW	SimIMS
increase(i)	0.004	0	0
increase_one(i)	0.004	0	0
increase_equal(i)	**0.137**	0	0.250
decrease(i)	0.013	0	0
decrease_one(i)	0.019	0	0
decrease_equal(i)	**0.112**	0	0
difference_nb_voices(i)	0.009	0	0
maximal_voices(i)	0.026	0.500	0
minimal_voices(i)	0.007	0	0.250
maximal_sim_notes(i)	0.007	0	0
crossed_voices(C)	0.009	0	0
no_crossed_voices(C)	**0.248**	0.250	0.250
extreme_pitch(C)	**0.090**	0.250	0.250
avg_pitch_right(C)	**0.117**	0	0
avg_pitch_left(C)	0.023	0	0
avg_pitch(C)	0.041	0	0
extreme_dur(C)	0.007	0	0
avg_dur_right(C)	0.048	0	0
avg_dur_left(C)	0.006	0	0
avg_dur(C)	**0.073**	0	0

Table 1. Coefficients weighting the musical features used to measure the connection quality, with best coefficients learned on the wtc-i corpus (GA1) and coefficients simulating the connection policy of CW and IMS.

5.4 Other sets of coefficients

To assess reproducibility, we ran the experiment two other times. The learned coefficients GA1′ and GA1″ are very close to GA1 (data not shown) and give comparable results on the learning corpus "wtc-i" (TR-prec = 97.83% and 97.81%, instead of 97.84%). We also optimized coefficients to find a worst solution (data not shown). The coefficients values *crossed_voices(C)* and *minimal_voices(i)* stand out. This confirms that predicting crossing voices is difficult and than small contigs are difficult to connect.

6. CONCLUSION

Voice and stream separation are improved when one optimizes at the same time *when* and *how* the voice fragments should be connected. We explored several features to evaluate the quality of these connections on fugues and string quartets. Taking into account the average pitches and durations of fragments leads to better connections. The resulting algorithm connects voice fragments more reliably than with the previous contig policies, and especially computes high-quality connections at the first steps. This work could be extended by considering more corpora and by evaluating further melodic or structural analysis on the resulting voices or streams. The proposed principles apply to contig-based algorithms but may also be used by other methods clustering notes into voices or streams.

Corpus	Connection policy	CC	TR-rec	TR-prec	S_o	S_u
wtc-i (training set)	no connection	–	86.78%	**99.32%**	**0.98**	0.34
	GA1-75%	**92.61%**	93.45%	98.54%	0.91	0.42
	GA1	**89.30%**	**97.84%**		**0.72**	**0.72**
	worst	16.93%	85.25%		0.06	0.09
	SimCW	81.26%	96.58%		0.65	0.64
	SimIMS	80.62%	96.55%		0.68	0.69
wtc-ii	no connection	–	86.66%	**99.29%**	**0.98**	0.35
	GA1-75%	**92.54%**	92.53%	98.36%	0.91	0.40
	GA1	**87.50%**	**97.14%**		**0.71**	**0.71**
	worst	25.06%	84.22%		0.05	0.07
	SimCW	83.27%	96.22%		0.69	0.68
	SimIMS	81.61%	96.07%		0.69	0.68
string quartets	no connection	–	82.61%	**97.00%**	**0.94**	0.29
	GA1-75%	**85.30%**	87.06%	94.80%	0.83	0.32
	GA1	**78.44%**	**92.59%**		0.44	0.44
	worst	31.88%	80.59%		0.12	0.13
	SimCW	75.99%	92.29%		0.39	0.38
	SimIMS	74.53%	91.79%		**0.62**	**0.61**

Table 2. Evaluation of the quality of various connection policies. Note that the two first policies (No connection, GA1-75%) do not try to connect the whole voices: they have very high TR-prec/S_o metrics, but poorer TR-rec/S_u metrics.

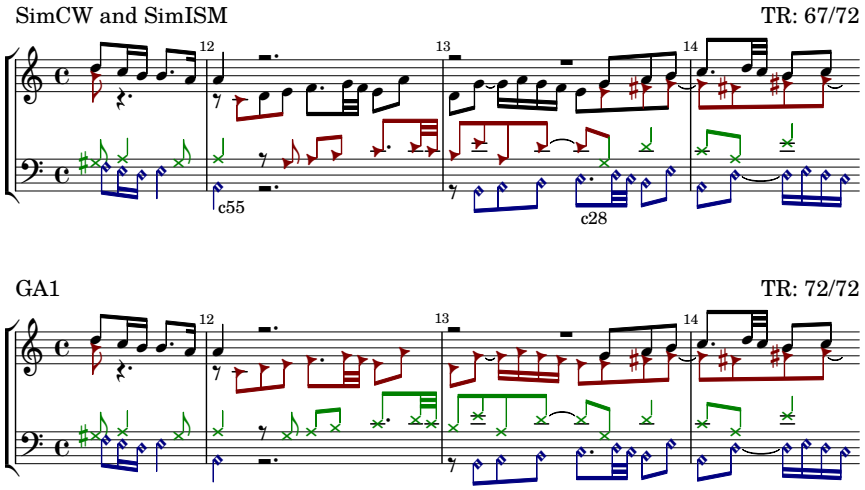


Figure 3. Extract of the fugue in C major BWV 846 by J.-S. Bach. (Top.) The connection policy of previous algorithms fails on connection c28 because of the fifth leap between the D and the G in the tenor voice. This error leads to the wrong connection c55 at a later stage of the algorithm. (Bottom.) Because the coefficients GA1 take into account the feature *avg_pitch(C)* and the related features, the connection is correct here.

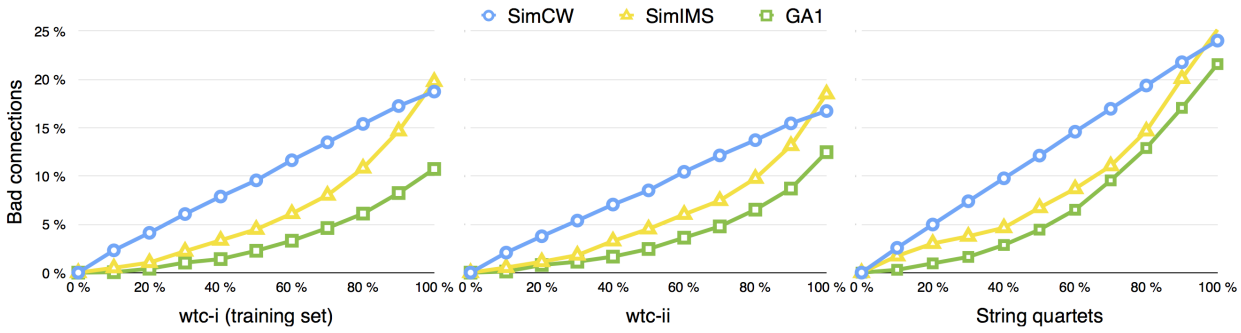


Figure 4. Errors done during the successive connection steps. The lower the curves, the better. Coefficients SimCW (blue): the error rate is almost constant. Coefficients SimIMS (yellow): the first connections are more reliable. Coefficients GA1 (green): the first connections are even more reliable, enabling to improve the algorithm by stopping before too much bad connections happen. The highest number of bad connections for string quartets (compared to fugues) is probably due to a less regular polyphonic writing, with in particular stylistic differences leading to larger intervals.

7. REFERENCES

[1] Albert Donally Bethke. Genetic algorithms as function optimizers. In *ACM Computer Science Conference*, 1978.

[2] Elaine Chew and Xiaodan Wu. Separating voices in polyphonic music: A contig mapping approach. In *International Symposium on Computer Music Modeling and Retrieval (CMMR 2005)*, pages 1–20. 2005.

[3] Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. In *International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 637–642, 2010.

[4] Reinier de Valk, Tillman Weyde, and Emmanouil Benetos. A machine learning approach to voice separation in lute tablature. In *International Society for Music Information Retrieval Conference (ISMIR 2013)*, pages 555–560, 2013.

[5] Diana Deutsch, editor. *The psychology of music*. Academic Press, 1982.

[6] Nicolas Guiomard-Kagan, Mathieu Giraud, Richard Groult, and Florence Levé. Comparing voice and stream segmentation algorithms. In *International Society for Music Information Retrieval Conference (ISMIR 2015)*, pages 493–499, 2015.

[7] David Huron. Tone and voice: A derivation of the rules of voice-leading from perceptual principles. *Music Perception*, 19(1):1–64, 2001.

[8] David Huron. Music information processing using the Humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):11–26, 2002.

[9] Asako Ishigaki, Masaki Matsubara, and Hiroaki Saito. Prioritized contig combining to segragate voices in polyphonic music. In *Sound and Music Computing Conference (SMC 2011)*, volume 119, 2011.

[10] Jürgen Kilian and Holger H Hoos. Voice separation – a local optimization approach. In *International Conference on Music Information Retrieval (ISMIR 2002)*, 2002.

[11] Phillip B Kirlin and Paul E Utgoff. Voise: Learning to segregate voices in explicit and implicit polyphony. In *International Conference on Music Information Retrieval (ISMIR 2005)*, pages 552–557, 2005.

[12] Hanna M Lukashevich. Towards quantitative measures of evaluating song segmentation. In *International Conference on Music Information Retrieval (ISMIR 2008)*, pages 375–380, 2008.

[13] Andrew McLeod and Mark Steedman. HMM-based voice separation of MIDI performance. *Journal of New Music Research*, 45(1):17–26, 2016.

[14] Dimitrios Rafailidis, Alexandros Nanopoulos, Yannis Manolopoulos, and Emilios Cambouropoulos. Detection of stream segments in symbolic musical data. In *International Conference on Music Information Retrieval (ISMIR 2008)*, pages 83–88, 2008.

[15] Dimitris Rafailidis, Emilios Cambouropoulos, and Yannis Manolopoulos. Musical voice integration/segregation: Visa revisited. In *Sound and Music Computing Conference (SMC 2009)*, pages 42–47, 2009.

[16] David Temperley. *The Cognition of Basic Musical Structures*. The MIT Press, 2001.

INTEGER PROGRAMMING FORMULATION OF THE PROBLEM OF GENERATING MILTON BABBITT'S ALL-PARTITION ARRAYS

Tsubasa Tanaka
STMS Lab : IRCAM, CNRS, UPMC
Paris, France
tsubasa.tanaka@ircam.fr

Brian Bemman
Aalborg University
Aalborg, Denmark
bb@create.aau.dk

David Meredith
Aalborg University
Aalborg, Denmark
dave@create.aau.dk

ABSTRACT

Milton Babbitt (1916–2011) was a composer of twelve-tone serial music noted for creating the *all-partition array*. The problem of generating an all-partition array involves finding a rectangular array of pitch-class integers that can be partitioned into regions, each of which represents a distinct integer partition of 12. Integer programming (IP) has proven to be effective for solving such combinatorial problems, however, it has never before been applied to the problem addressed in this paper. We introduce a new way of viewing this problem as one in which restricted overlaps between integer partition regions are allowed. This permits us to describe the problem using a set of linear constraints necessary for IP. In particular, we show that this problem can be defined as a special case of the well-known problem of set-covering (SCP), modified with additional constraints. Due to the difficulty of the problem, we have yet to discover a solution. However, we assess the potential practicality of our method by running it on smaller similar problems.

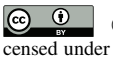
1. INTRODUCTION

Milton Babbitt (1916–2011) was a composer of twelve-tone serial music noted for developing complex and highly constrained music. The structures of many of his pieces are governed by a structure known as the *all-partition array*, which consists of a rectangular array of pitch-class integers, partitioned into regions of distinct "shapes", each corresponding to a distinct integer partition of 12. This structure helped Babbitt to achieve *maximal diversity* in his works, that is, the presentation of as many musical parameters in as many different variants as possible [13].

In this paper, we formalize the problem of generating an all-partition array using an integer programming paradigm in which a solution requires solving a special case of the set-covering problem (SCP), where the subsets in the cover are allowed a restricted number of overlaps with one another and where the ways in which these overlaps can occur is constrained. It turns out that this is a hard combinatorial problem. That this problem was solved by Babbitt and one of his students, David Smalley, without the use of a computer is therefore interesting in itself. Moreover, it suggests that there exists an effective procedure for solving the problem.

Construction of an all-partition array begins with an $I \times J$ matrix, A, of pitch-classes, $0, 1, \ldots, 11$, where each row contains $J/12$ twelve-tone rows. In this paper, we only consider matrices where $I = 6$ and $J = 96$, as matrices of this size figure prominently in Babbitt's music [13]. This results in a 6×96 matrix of pitch classes, containing 48 twelve-tone rows. In other words, A will contain an approximately uniform distribution of 48 occurrences of each of the integers from 0 to 11. On the musical surface, rows of this matrix become expressed as 'musical voices', typically distinguished from one another by instrumental register [13]. A complete all-partition array is a matrix, A, partitioned into K regions, each of which must contain each of the 12 pitch classes exactly once. Moreover, each of these regions must have a distinct "shape", determined by a distinct *integer partition* of 12 (e.g., $2+2+2+3+3$ or $1+2+3+1+2+3$) that contains I or fewer summands greater than zero [7]. We denote an integer partition of an integer, L, by $\mathrm{IntPart}_L(s_1, s_2, \ldots, s_I)$ and define it to be an ordered set of non-negative integers, $\langle s_1, s_2, \ldots, s_I \rangle$, where $L = \sum_{i=1}^{I} s_i$ and $s_1 \geq s_2 \geq \cdots \geq s_I$. For example, possible integer partitions of 12 when $I = 6$, include $\mathrm{IntPart}_{12}(3, 3, 2, 2, 1, 1)$ and $\mathrm{IntPart}_{12}(3, 3, 3, 3, 0, 0)$. We define an *integer composition* of a positive integer, L, denoted by $\mathrm{IntComp}_L(s_1, s_2, \ldots, s_I)$, to also be an ordered set of I non-negative integers, $\langle s_1, s_2, \ldots, s_I \rangle$, where $L = \sum_{i=1}^{I} s_i$, however, unlike an integer partition, the summands are not constrained to being in descending order of size. For example, if $L = 12$ and $I = 6$, then $\mathrm{IntComp}_{12}(3, 3, 3, 3, 0, 0)$ and $\mathrm{IntComp}_{12}(3, 0, 3, 3, 3, 0)$ are two distinct integer compositions of 12 defining the same integer *partition*, namely $\mathrm{IntPart}_{12}(3, 3, 3, 3, 0, 0)$.

Figure 1 shows a 6×12 excerpt from a 6×96 pitch-class matrix, A, and a region determined by the integer composition, $\mathrm{IntComp}_{12}(3, 2, 1, 3, 1, 2)$, containing each possible pitch class exactly once. Note, in Figure 1, that each summand (from left to right) in $\mathrm{IntComp}_{12}(3, 2, 1, 3, 1, 2)$, gives the number of elements in the corresponding row of the matrix (from top to bottom) in the region determined by the integer composition. We call this part of a region



© Tsubasa Tanaka, Brian Bemman, David Meredith. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Tsubasa Tanaka, Brian Bemman, David Meredith. "Integer Programming Formulation of the Problem of Generating Milton Babbitt's All-partition Arrays", 17th International Society for Music Information Retrieval Conference, 2016.

11	4	3	5	9	10	1	8	2	0	7	6
6	7	0	2	8	1	10	9	5	3	4	11
5	6	11	1	7	0	9	8	4	2	3	10
2	9	10	8	4	3	0	5	11	1	6	7
0	5	4	6	10	11	2	9	3	1	8	7
1	8	9	7	3	2	11	4	10	0	5	6

Figure 1: A 6×12 excerpt from a 6×96 pitch-class matrix with the integer composition, $\mathrm{IntComp}_{12}(3, 2, 1, 3, 1, 2)$ (in dark gray), containing each pitch class exactly once.

11	4	3	**3**	5	9	10	1	8	2	0	7	6
6	7	**7**	0	2	8	1	10	9	5	3	4	11
5	6	11	1	7	0	9	8	4	2	3	10	
2	9	10	**10**	8	4	3	0	5	11	1	6	7
0	5	4	6	10	11	2	9	3	1	8	7	
1	8	9	7	3	2	11	4	10	0	5	6	

Figure 2: A 6×12 excerpt from a 6×96 pitch-class matrix with a region whose shape is determined by the integer composition, $\mathrm{IntComp}_{12}(3, 3, 3, 3, 0, 0)$ (in light gray), where three elements (in bold) are horizontal insertions of pitch classes from the previous integer partition region. Note that the two indicated regions represent distinct integer partitions.

in a given row of the matrix a *summand segment*. For example, in Figure 1, the summand segment in the first row for the indicated integer partition region contains the pitch classes 11, 4 and 3. On the musical surface, the distinct shape of each integer composition helps contribute to a progression of 'musical voices' that vary in textural density, allowing for relatively thick textures in, for example, $\mathrm{IntComp}_{12}(2, 2, 2, 2, 2, 2)$ (with six participating parts) and comparatively sparse textures in, for example, $\mathrm{IntComp}_{12}(11, 0, 1, 0, 0, 0)$ (with two participating parts).

There exist a total of 58 distinct integer partitions of 12 into 6 or fewer non-zero summands [13]. An all-partition array with six rows will thus contain $K = 58$ regions, each containing every pitch class exactly once and each with a distinct shape determined by an integer composition representing a distinct integer partition. However, the number of pitch-class integers required to satisfy this constraint, $58 \times 12 = 696$, exceeds the size of a 6×96 matrix containing 576 elements, by 120. In order to satisfy this constraint, additional pitch-classes therefore have to be inserted into the matrix, with the added constraint that only horizontal insertions of at most one pitch class in each row are allowed for each of the 58 integer partition regions. Each inserted pitch class is identical to its immediate neighbor to the left, this being the right-most element of a summand segment belonging to a previous integer partition region. This constraint ensures that the order of pitch classes in the twelve-tone rows of a given row of A is not altered [13]. Figure 2 shows a second integer partition region, $\mathrm{IntComp}_{12}(3, 3, 3, 3, 0, 0)$, in the matrix shown in Figure 1 (indicated in light gray), where three of its elements result from horizontal insertions of pitch classes from the previous integer partition region. Note, in Figure 2, the three horizontal insertions of pitch-class integers, 3 (in row 1), 7 (in row 2), and 10 (in row 4), required to have each pitch class occur exactly once in the second integer partition region. Not all of the 58 integer partitions must contain one or more of these insertions, however, the total number of insertions must equal the 120 additional pitch classes required to satisfy the constraint that all 58 integer partitions are represented. Note that, in order for each of the resulting integer partition regions to contain every pitch class exactly once, ten occurrences of each of the 12 pitch classes must be inserted into the matrix. This typically results in the resulting matrix being irregular (i.e., "ragged" along its right side).

In this paper, we address the problem of generating an all-partition array by formulating it as a set of linear constraints using the integer programming (IP) paradigm. In section 2, we review previous work on general IP problems and their use in the generation of musical structures. We also review previous work on the problem of generating all-partition arrays. In section 3, we introduce a way of viewing insertions of elements into the all-partition array as fixed locations in which overlaps occur between contiguous integer partition regions. In this way, our matrix remains regular and we can define the problem as a special case of the well-known IP problem of set-covering (SCP), modified so that certain overlaps are allowed between the subsets. In sections 4 and 5, we present our IP formulation of this problem as a set of linear constraints. Due to the difficulty of the problem, we have yet to discover a solution using our formulation. Nevertheless, in section 6, we present the results of using our implementation to find solutions to smaller versions of the problem and in this way explore the practicality of our proposed method. We conclude in section 7 by mentioning possible extensions to our formulation that could potentially allow it to solve the complete all-partition array generation problem.

2. PREVIOUS WORK

Babbitt himself laid the foundations for the construction of what would become the all-partition array during the 1960s, and he would continue to use the structure in nearly all of his later works [1–4]. Subsequent composers made use of the all-partition array in their own music and further developed ways in which its structure could be formed and used [5, 6, 11, 12, 14, 15, 17, 18, 21]. Most of these methods focus on the organization of pitch classes in a twelve-tone row and how their arrangement can make the construction of an all-partition array more likely. We propose here a more general purpose solution that will take any matrix and attempt to generate a successful structure. Furthermore, many of these previous methods were music-theoretical in nature and not explicitly computational. Work by Bazelow and Brickle is one notable exception [5, 6]. We agree here with their assessment that 'partition problems in twelve-tone theory properly belong to the study of combinatorial algorithms' [6]. However, we differ considerably in our approach and how we conceive of the structure of the all-partition array.

More recent efforts to automatically analyze and generate all-partition arrays have been based on backtracking algorithms. [7–9]. True to the structure of the all-partition array (as it appears on the musical surface) and the way in which Babbitt and other music theorists conceive of its structure, these attempts to generate an all-partition array form regions of pitch classes according to the process described in section 1, where horizontal repetitions of pitch-classes are added, resulting in an irregular matrix. While these existing methods have further proposed various heuristics to limit the solution space or allow for incomplete solutions, they were unable to generate a complete all-partition array [7–9].

In general, for difficult combinatorial problems, more efficient solving strategies than backtracking exist. One such example is integer programming (IP). IP is a computationally efficient and practical paradigm for dealing with typically NP-hard problems, such as the traveling salesman, set-covering and set-partitioning problems, where these are expressed using only linear constraints (i.e., equations and inequalities) and a linear objective function [10, 16]. One benefit of using IP, is that it allows for the separation of the formulation of a problem by users and the development by specialists of an algorithm for solving it. Many of these powerful solvers dedicated to IP problems have been developed and used particularly in the field of operations research. Compared to approximate computational strategies, such as genetic algorithms, IP formulations and their solvers are suitable for searching for solutions that strictly satisfy necessary constraints. For this reason, we expect that the IP paradigm could provide an appropriate method for approaching the problem of generating all-partition arrays.

In recent work, IP has been applied to problems of analysis and generation of music [19, 20]. This is of importance to the research presented here as it demonstrates the relevance of these traditional optimization problems of set-covering (SCP) and set-partitioning (SPP), to general problems found in computational musicology, where SPP has been used in the segmentation of melodic motifs and IP has been used in describing global form. In the next section, we address the set-covering problem (SCP) in greater detail and show how it is related to the problem of generating all-partition arrays.

3. SET-COVERING PROBLEM FORMULATION OF ALL-PARTITION ARRAY GENERATION

The set-covering (SCP) problem is a well-known problem in computer science and operations research that can be shown to be NP-hard [10]. Let E be a set whose elements are $\{E_1, E_2, \cdots, E_{\#E}\}$ (where $\#E$ denotes the number of elements in E), F be a family of subsets of E, $\{F_1, F_2, \cdots, F_{\#F}\}$, and S be a subset of F. By assigning a constant cost, c_s, to each F_s, the objective of the

11	4	3	5	9	10	1	8	2	0	7	6
6	7	0	2	8	1	10	9	5	3	4	11
5	6	11	1	7	0	9	8	4	2	3	10
2	9	10	8	4	3	0	5	11	1	6	7
0	5	4	6	10	11	2	9	3	1	8	7
1	8	9	7	3	2	11	4	10	0	5	6

Figure 3: A 6×12 excerpt from a 6×96 pitch-class matrix with two integer compositions, $\mathrm{IntComp}_{12}(3, 2, 1, 3, 1, 2)$ (in dark gray and outlined) and $\mathrm{IntComp}_{12}(3, 3, 3, 3, 0, 0)$ (in light gray), that form distinct integer partition regions. Note, that the second composition overlaps three fixed locations in the first.

set-covering problem (SCP) is to

$$\underset{S \subset F}{Minimize} \sum_{F_s \in S} c_s$$
$$subject\ to \bigcup_{F_s \in S} F_s = E.$$

In other words, a solution S is a *cover* of E that allows for the same elements to appear in more than one subset, F_s. In this section, we suggest that our problem can be viewed as an SCP with additional constraints.

3.1 All-partition array generation as a set-covering problem (SCP) with additional constraints

When viewing the all-partition array in the context of $\bigcup_{F_s \in S} F_s = E$ above, E is the set that consists of all locations (i, j) in the matrix, A, and F_s are the sets of locations (i, j) that correspond to the "shapes" of integer compositions. We call each F_s a *candidate set*. A candidate set F_s is characterized by two conditions that we call *containment* and *consecutiveness*. Containment means that the elements (i.e., locations (i, j)) of F_s correspond to twelve distinct integers, $0, 1, \ldots, 11$, in A. Consecutiveness means that each of its elements belonging to the same row in A are consecutive. In this sense, F includes all sets found in A that satisfy the conditions of consecutiveness and containment.

As the expression $\bigcup_{F_s \in S} F_s = E$ implies, a candidate set is allowed to share elements with another candidate set. Similarly, the pitch classes in A (i.e., corresponding to elements in E) that become insertions in the original problem can be instead regarded as shared elements or *overlaps* between contiguous integer composition regions, with the result that the matrix remains regular. Figure 3 shows how these overlaps would occur in the two integer composition regions shown in Figure 2.

Viewed in this way, a solution to the problem of generating an all-partition array thus satisfies the basic criterion of an SCP, namely, the condition for *set-covering*, $\bigcup_{F_s \in S} F_s = E$. However, this criterion alone fails to account for the unique constraints under which such a covering is formed in an all-partition array. In the original SCP, there are no constraints on the order of subsets, the order of their elements or the number of overlaps and the ways in

which they can occur. On the other hand, an all-partition array must satisfy such additional conditions. We denote the constraints for satisfying such additional conditions by $Add.\ Conditions$.

$Add.\ Conditions$ includes the conditions in the all-partition array governing (1) the left-to-right order of contiguous candidate sets, (2) permissible overlaps between such sets, and (3) the *distinctness* of sets in S. This last condition of distinctness ensures that the integer compositions used in a cover, S, define every possible integer partition once and only once. On the other hand, the conditions for set-covering, $\bigcup_{F_s \in S} F_s = E$, are conditions of (1) candidate sets (which satisfy containment and consecutiveness) and (2) *covering*, meaning that each element in E is covered no less than once.

We can now state that our problem of generating an all-partition array is to

$$\begin{aligned} &Minimize \underset{S \subset F}{\sum_{F_s \in S}} c_s \\ &subject\ to \bigcup_{F_s \in S} F_s = E, \\ &\qquad Add.\ Conditions. \end{aligned}$$

where the associated cost, c_s, of each F_s, can be interpreted as a preference for one integer composition or another. It is likely that, in the interest of musical expression, Babbitt may have preferred the shapes of some integer partition regions over others [13]. However, as his preference is unknown, we can regard these costs to have the same value for each F_s.

Due to the condition of distinctness (just described), $|S|$ can be fixed at 58. This feature, combined with the equal costs of each F_s, means that the objective function, $\sum_{F_s \in S} c_s$, for this problem, is constant. For these reasons, the above formulation is a *constraint satisfaction problem*. This motivates our discussions in sections 6 and 7 on possible alternative objective functions.

In the next two sections, we implement the constraint satisfaction problem defined above using integer programming (IP). In particular, section 4 addresses the conditions for set-covering, $\bigcup_{F_s \in S} F_s = E$, and section 5 addresses those in $Add.\ Conditions$. It is because of our new way of viewing this problem, with a regular matrix and overlaps, that we are able to introduce variables for use in IP to describe these conditions.

4. IP IMPLEMENTATION OF CONDITIONS FOR SET-COVERING IN ALL-PARTITION ARRAY GENERATION

In this section, we introduce our set of linear constraints for satisfying the general conditions for set-covering, $\bigcup_{F_s \in S} F_s = E$, in the generation of an all-partition array. Before we introduce these constraints, we define the necessary variables and constants used in our implementation of the conditions for set-covering. We begin with a given matrix found in one of Babbitt's works based on the all-partition array. Examples of the matrices used in this paper can be found in Babbitt's *Arie da Capo* (1974) and *None but the Lonely Flute* (1991), among others. Let $(A_{i,j})$ be a $(6, 96)$-matrix whose elements are the pitch-class integers, $0, 1, \ldots, 11$. We denote the number of rows and columns by I and J, respectively.

Let $x_{i,j,k}$ $(1 \leq i \leq I,\ 1 \leq j \leq J)$ be a binary variable corresponding to each location (i, j) in A and a subset (i.e., integer partition) identified by the integer k, where $1 \leq k \leq K$ and $K = 58$. Here, we consider the case where $I = 6$ and $J = 96$, so there are 58 sets of 576 such variables. Each of these variables will indicate whether or not a location (i, j) belongs to a candidate set for the kth position in the sequence of 58 integer partition regions. We denote the set of locations (i, j) whose corresponding value for $x_{i,j,k}$ is 1, to be C_k. Subject to conditions for consecutiveness and containment, C_k will be a candidate set.

Let $(B^p_{i,j})$ $(0 \leq p \leq 11)$ be constant matrices, equal in size to A, where $B^p_{i,j} = 1$ if and only if $A_{i,j} = p$ and $B^p_{i,j} = 0$ otherwise. The locations (i, j) whose values of $B^p_{i,j}$ equal 1, correspond to the locations of pitch-class p in A.

4.1 Conditions for C_k to contain twelve distinct integers in A (condition of containment)

A condition for C_k to satisfy the condition of containment is that its number of elements is 12 and each corresponds to a distinct pitch-class in A. These conditions are expressed by the following two equations:

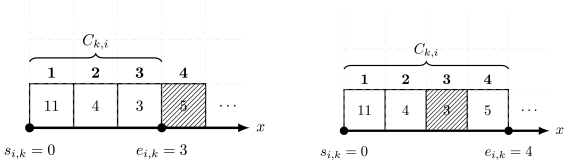
$$\forall k \in [1, K],\ \sum_{i=1}^{I} \sum_{j=1}^{J} x_{i,j,k} = 12, \tag{1}$$

$$\forall p \in [0, 11], \forall k \in [1, K],\ \sum_{i=1}^{I} \sum_{j=1}^{J} B^p_{i,j} \cdot x_{i,j,k} = 1. \tag{2}$$

Because $x_{i,j,k}$ equals 1 if (i, j) is included in C_k and 0 if it is not, Equation 1 means that there are 12 elements in C_k. In Equation 2, we ensure that each corresponding pitch-class integer p for the elements in C_k, appears once and only once.

4.2 Conditions for C_k to be integer compositions in A (condition of consecutiveness)

Let $C_{k,i}$ be the ith-row part of C_k (i.e., the summand segment of composition k for row i). Let $s_{i,k}$ be an integer variable corresponding to the x-coordinate of a 'starting point', which lies at the left side of the leftmost component of $C_{k,i}$. The value of $s_{i,k}$ is then equal to the column number of the leftmost component of $C_{k,i}$, minus 1. The origin point of this coordinate lies along the left hand side of the matrix A, and we set the width of each location (i, j) to be 1. Similarly, let $e_{i,k}$ be an integer variable corresponding to the x-coordinate of an 'ending point', which lies at the right side of the rightmost component belonging to $C_{k,i}$. The value of $e_{i,k}$ is then equal to the column number of the



(a) $C_{k,i}$ contains pitch classes $11, 4, 3$ $(j = 1, 2, 3)$ and satisfies consecutiveness.

(b) $C_{k,i}$ contains pitch classes $11, 4, 5$ $(j = 1, 2, 4)$ and does not satisfy consecutiveness.

Figure 4: Two $C_{k,i}$ and corresponding $s_{i,k}$ and $e_{i,k}$ from Figure 3 when $k = 1$ and $i = 1$. Shaded elements indicate $x_{i,j,k} = 0$ and unshaded elements indicate $x_{i,j,k} = 1$.

rightmost component of $C_{k,i}$. Figure 4 shows an example of two possible $C_{k,i}$ from Figure 3. If there is no component in $C_{k,i}$ $(k \geq 2)$, we define $s_{i,k}$ to be $e_{i,k-1}$ and $e_{i,k}$ to be $s_{i,k}$. If there is no component in $C_{1,i}$, we define $s_{i,k}$ and $e_{i,k}$ to be 0. Then, $s_{i,k}$ and $e_{i,k}$ are subject to the following constraint of range:

$$\forall i \in [1, I], \forall k \in [1, K],\ 0 \leq s_{i,k} \leq e_{i,k} \leq J. \quad (3)$$

The condition under which C_k $(k \in [1, K])$ forms an integer composition—that is, satisfies the condition of consecutiveness, is expressed by the following three constraints:

$$\forall i \in [1, I], \forall j \in [1, J], \forall k \in [1, K], \quad j \cdot x_{i,j,k} \leq e_{i,k}, \quad (4)$$

$$\forall i \in [1, I], \forall j \in [1, J], \forall k \in [1, K], \quad J - s_{i,k} \geq (J + 1 - j) \cdot x_{i,j,k}, \quad (5)$$

$$\forall i \in [1, I], \forall k \in [1, K],\ \sum_{j=1}^{J} x_{i,j,k} = e_{i,k} - s_{i,k}. \quad (6)$$

In Equation 4, each element of $C_{k,i}$ must be located at column $e_{i,k}$ or to the left of column $e_{i,k}$. Equation 5 states that each element of $C_{k,i}$ must be located at column $s_{i,k} + 1$ or to the right of column $s_{i,k} + 1$. Equation 6, combined with the previous two constraints, states that the length of $C_{k,i}$ must be equal to $e_{i,k} - s_{i,k}$, implying that the column numbers j of the elements in $C_{k,i}$ are consecutive from $s_{i,k} + 1$ to $e_{i,k}$, where $C_{k,i}$ contains at least one element.

4.3 Condition for covering A

As every location (i, j) in A (i.e., E in our SCP) must be covered at least once, we pose the following condition of covering:

$$\forall i \in [1, I], \forall j \in [1, J],\ \sum_{k=1}^{K} x_{i,j,k} \geq 1. \quad (7)$$

Equation 1 states that for all $K = 58$ integer partitions, there are $12 \cdot K = 696$ variables, $x_{i,j,k}$, that will equal 1. A successful cover of A by Equation 7, however, states that all of $I \cdot J = 576$ places (i, j) in A, are covered once or more than once. Collectively, these imply that there are 120 or less than 120 places (i.e., combinations of (i, j)) that are covered twice or more than twice. These 120 overlaps correspond to the 120 insertions of pitch-class integers used when constructing an all-partition array in its original form. By satisfying all of the constraints above, each C_k forms a candidate set (i.e., a member of F in our SCP) and the condition for set-covering, $\bigcup_{F_s \in S} F_s = E$, is satisfied.

5. IP IMPLEMENTATION OF ADDITIONAL CONDITIONS IN ALL-PARTITION ARRAY GENERATION

In this section, we introduce our set of additional linear constraints beyond those required for satisfying the condition of set-covering in the SCP.

5.1 Left-to-right order of C_k and permissible overlaps

C_k must be located immediately to the right of C_{k-1}. This is expressed by

$$\forall i \in [1, I], \forall k \in [2, K],\ e_{i,k-1} \leq e_{i,k}, \quad (8)$$

$C_{k-1,i}$ and $C_{k,i}$ may overlap by no more than one element. This is expressed by the following inequality:

$$\forall i \in [1, I], \forall k \in [2, K],\ e_{i,k-1} - 1 \leq s_{i,k} \leq e_{i,k-1}, \quad (9)$$

meaning that $s_{i,k}$ will be equal to $e_{i,k-1}$ if there is no overlap and $s_{i,k}$ will be equal to $e_{i,k-1} - 1$ if there is an overlap.

5.2 Conditions for C_k to be integer compositions defining distinct integer partitions (condition of distinctness)

Let $y_{i,k,l}$ be a binary variable that indicates whether or not the length of $C_{k,i}$ is greater than or equal to l $(1 \leq l \leq L, L = 12)$, by introducing the following constraints:

$$\forall i \in [1, I], \forall k \in [1, K],\ e_{i,k} - s_{i,k} = \sum_{l=1}^{L} y_{i,k,l}, \quad (10)$$

$$\forall i \in [1, I], \forall k \in [1, K], \forall l \in [2, L], \quad y_{i,k,l-1} \geq y_{i,k,l}. \quad (11)$$

Equation 10 states that the sum of all elements in $\langle y_{i,k,1}, y_{i,k,2}, \ldots, y_{i,k,L} \rangle$ is equal to the length of $C_{k,i}$, while Equation 11 states that its elements equal to 1 begin in the first position and are consecutive (e.g., $\langle 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle$, when the length of $C_{k,i}$ is 3.)

The number of the lengths of $C_{k,i}$ $(1 \leq i \leq I)$ that are greater than or equal to l is given by $\sum_{i=1}^{I} y_{i,k,l}$. The twelve values of $\sum_{i=1}^{I} y_{i,k,l}$ $(1 \leq l \leq L)$ then, will precisely represent the type of partition. For example, if C_k is $\mathrm{IntComp}_{12}(3, 2, 1, 3, 1, 2)$, then $y_{i,k,l}$ $(\forall i \in [1, I], \forall l \in [1, L])$ would be

1,1,1,0,0,0,0,0,0,0,0,0
1,1,0,0,0,0,0,0,0,0,0,0
1,0,0,0,0,0,0,0,0,0,0,0
1,1,1,0,0,0,0,0,0,0,0,0
1,0,0,0,0,0,0,0,0,0,0,0
1,1,0,0,0,0,0,0,0,0,0,0

and $\sum_{i=1}^{I} y_{i,k,l}$ would be $[6, 4, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0]$.

We denote the number of all integer partitions by N ($N = K = 58$) and denote a single integer partition n ($1 \leq n \leq N$) by P_n. We can express P_n as $[P_{n,1}, P_{n,2}, \ldots, P_{n,L}]$ ($1 \leq n \leq N$), where $P_{n,l}$ corresponds to the twelve values $\sum_{i=1}^{I} y_{i,k,l}$ ($1 \leq l \leq L$) described above.

Then, by implementing the following expression:

$$\forall k \in [1, K], \forall n \in [1, N], \forall l \in [1, L], \quad P_{n,l} \cdot z_{k,n} \leq \sum_{i=1}^{I} y_{i,k,l}. \tag{12}$$

we can express whether C_k defines the integer partition n or not by the binary variable $z_{k,n}$. For example, if $z_{k,n} = 0$, the value of $P_{n,l} \cdot z_{k,n} = 0$ constrains nothing, and thus C_k cannot be the integer partition n (because of the next equation). On the other hand, if $z_{k,n} = 1$, C_k must be the integer partition n. Accordingly, $z_{k,n}$ will equal 1 only if the twelve values $\sum_{i=1}^{I} y_{i,k,l}$ correspond to P_n. From this, determining whether or not all different partitions are present can be expressed by the following equation:

$$\forall n \in [1, N], \ \sum_{k=1}^{K} z_{k,n} = 1. \tag{13}$$

6. EXPERIMENTS

In order to determine whether or not our formulation works as intended, we implemented the constraints described in sections 4 and 5 and supplied these to an IP solver based on branch-and-bound (Gurobi Optimizer). As the objective function in our formulation amounts to a constant-cost function (described in section 3), we replaced it with a non-constant objective function, $\sum_{i,j,k} c_{i,j,k} \cdot x_{i,j,k}$, where $c_{i,j,k}$ assumes a randomly generated integer for promoting this process of branch and bound. When the first feasible solution is found, we stop the search.

Although we first attempted to find a complete all-partition array, we were unable to discover a solution after one day of calculation. This highlights the difficulty of the problem and reinforces those findings by previous methods that were similarly unable to find a complete all-partition array [7]. As the target of our current formulation is only solutions which strictly satisfy all constraints, we opted to try finding complete solutions to smaller-sized problems, using the first j columns of the original matrix. Because we cannot use all 58 integer partitions in the case $K < N$, a slight modification to Equation 13 was needed for this change. Its equality was replaced by $\leq$ and an additional constraint, $\forall k \in [1, K]$, $\sum_{n=1}^{N} z_{k,n} = 1$, for allocating one partition to each C_k, was added.

Figure 5 shows the duration (vertical axis) of time spent on finding a solution in matrices of varying size. The number of integer compositions, K, was set to $(J+2)/2$, where J is an even number. This ensures that a given solution will always contain 12 overlaps. These findings suggest that the necessary computational time in finding a solution tends to dramatically increase with an increase in J. However, this increase fluctuates, suggesting that each small matrix represents a unique problem space with different sets of difficulties (e.g., the case $J = 14$ was unfeasible). For this reason, finding a solution in a complete matrix (6,96) within a realistic limitation of time would be difficult for our current method, even using a fast IP solver. This strongly motivates future improvements as well as the possibility of an altogether different strategy.

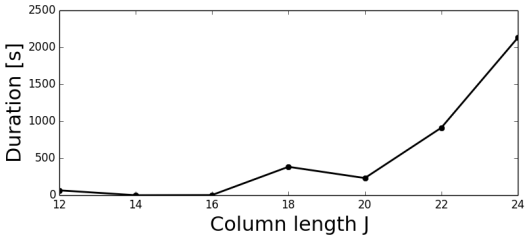


Figure 5: Duration of time spent on finding the first solution for each small matrix, whose column length is $J (12 \leq J \leq 24, J \in 2\mathbb{N})$. K is set to $(J + 2)/2$, resulting in 12 overlaps. Note, that no feasible solution exists when $J = 14$.

7. CONCLUSION AND FUTURE WORK

In this paper, we have introduced a novel integer-programming-based perspective on the problem of generating Milton Babbitt's all-partition arrays. We have shown that insertions and the irregular matrix that results can be replaced with restricted overlaps, leaving the regular matrix unchanged. This view allows us to formulate the problem as a set-covering problem (SCP) with additional constraints and then implement it using integer programming. Due to the difficulty of the problem, we have so far been unable to find a solution. However, we have been able to produce solutions in a practical running time (< 2500 seconds) when the matrix is reduced in size to 24 columns or less. These results motivate possible extensions to our formulation. First, a relaxation of the problem is possible, for example, by using an objective function that measures the degree of incompleteness of a solution. This could allow for approximate solutions to be discovered, such as those found in previous work [7]. Second, it may be the case that a solution to the full problem may be achievable by combining solutions to smaller subproblems that we have shown to be solvable in a practical running time.

8. ACKNOWLEDGEMENTS

The work of Tsubasa Tanaka reported in this paper was supported by JSPS Postdoctoral Fellowships for Research Abroad. The work of Brian Bemman and David Meredith was carried out as part of the project Lrn2Cre8, which acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET grant number 610859.

9. REFERENCES

[1] Milton Babbitt. Twelve-tone invariants as compositional determinants. *Journal of Music Theory*, 46(2):246–259, 1960.

[2] Milton Babbitt. Set structure as a compositional determinant. *Journal of Music Theory*, 5(1):72–94, 1961.

[3] Milton Babbitt. Twelve-tone rhythmic structure and the electronic medium. *Perspectives of New Music*, 1(1):49–79, 1962.

[4] Milton Babbitt. Since Schoenberg. *Perspectives of New Music*, 12(1/2):3–28, 1973.

[5] Alexander R. Bazelow and Frank Brickle. A partition problem posed by Milton Babbitt (Part I). *Perspectives of New Music*, 14(2):280–293, 1976.

[6] Alexander R. Bazelow and Frank Brickle. A combinatorial problem in music theory: Babbitt's partition problem (I). *Annals of the New York Academy of Sciences*, 319(1):47–63, 1979.

[7] Brian Bemman and David Meredith. Comparison of heuristics for generating all-partition arrays in the style of Milton Babbitt. In *CMMR 11th International Symposium on Computer Music Multidisciplinary Research, 16–19 June 2015*, Plymouth, UK, 2015.

[8] Brian Bemman and David Meredith. Exact cover problem in Milton Babbitt's all-partition array. In Tom Collins, David Meredith, and Anja Volk, editors, *Mathematics and Computation in Music: Fifth International Conference, MCM 2015, London, UK, June 22–25, 2015, Proceedings*, volume 9110 of *Lecture Notes in Artificial Intelligence*, pages 237–242. Springer, Berlin, 2015.

[9] Brian Bemman and David Meredith. Generating Milton Babbitt's all-partition arrays. *Journal of New Music Research*, 45(2), 2016. http://www.tandfonline.com/doi/full/10.1080/09298215.2016.1172646.

[10] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 3rd edition, 2009.

[11] David Kowalski. *The array as a compositional unit: A study of derivational counterpoint as a means of creating hierarchical structures in twelve-tone music*. PhD thesis, Princeton University, 1985.

[12] David Kowalski. The construction and use of self-deriving arrays. *Perspectives of New Music*, 25(1), 1987.

[13] Andrew Mead. *An Introduction to the Music of Milton Babbitt*. Princeton University Press, Princeton, NJ., 1994.

[14] Robert Morris. *Composition with Pitch-Classes: A Theory of Compositional Design*. Yale University Press, New Haven, CT, 1987.

[15] Robert Morris. *The Whistling Blackbird: Essays and Talks on New Music*. The University of Rochester Press, Princeton, NJ, 2010.

[16] A. J. Orman and H. Paul Williams. A survey of different integer programming formulations of the travelling salesman problem. In Erricos John Kontoghiorghes and Cristian Gatu, editors, *Optimisation, Econometric and Financial Analysis*, volume 9 of *Advances in computational management science*, pages 93–108. Springer-Verlag Berlin Heidelberg, Berlin, 2006.

[17] Daniel Starr and Robert Morris. A general theory of combinatoriality and the aggregate, part 1. *Perspectives of New Music*, 16(1):3–35, 1977.

[18] Daniel Starr and Robert Morris. A general theory of combinatoriality and the aggregate, part 2. *Perspectives of New Music*, 16(2):50–84, 1978.

[19] Tsubasa Tanaka and Koichi Fujii. Describing global musical structures by integer programming. In Tom Collins, David Meredith, and Anja Volk, editors, *Mathematics and Computation in Music: Fifth International Conference, MCM 2015, London, UK, June 22–25, 2015, Proceedings*, volume 9110 of *Lecture Notes in Artificial Intelligence*, pages 52–63. Springer, Berlin, 2015.

[20] Tsubasa Tanaka and Koichi Fujii. Melodic pattern segmentation of polyphonic music as a set partitioning problem. In *International Congress on Music and Mathematics, Puerto Vallarta, November 26–29, 2014, Proceedings*. Springer, Berlin, to be published.

[21] Godfrey Winham. Composition with arrays. *Perspectives of New Music*, 9(1):43–67, 1970.

INTEGRATION AND QUALITY ASSESSMENT OF HETEROGENEOUS CHORD SEQUENCES USING DATA FUSION

Hendrik Vincent Koops [1] **W. Bas de Haas** [2] **Dimitrios Bountouridis** [1] **Anja Volk** [1]

[1] Department of Information and Computing Sciences, Utrecht University, the Netherlands
{h.v.koops, d.bountouridis, a.volk}@uu.nl
[2] Chordify, the Netherlands {bas}@chordify.net

ABSTRACT

Two heads are better than one, and the many are smarter than the few. Integrating knowledge from multiple sources has shown to increase retrieval and classification accuracy in many domains. The recent explosion of crowd-sourced information, such as on websites hosting chords and tabs for popular songs, calls for sophisticated algorithms for data-driven quality assessment and data integration to create better, and more reliable data. In this paper, we propose to integrate the heterogeneous output of multiple automatic chord extraction algorithms using data fusion. First we show that data fusion creates significantly better chord label sequences from multiple sources, outperforming its source material, majority voting and random source integration. Second, we show that data fusion is capable of assessing the quality of sources with high precision from source agreement, without any ground-truth knowledge. Our study contributes to a growing body of work showing the benefits of integrating knowledge from multiple sources in an advanced way.

1. INTRODUCTION AND RELATED WORK

With the rapid growth and expansion of online sources containing user-generated content, a large amount of conflicting data can be found in many domains. For example, different encyclopediæ can provide conflicting information on the same subject, and different websites can provide conflicting departure times for public transportation. A typical example in the music domain is provided by websites offering data that allows for playing along with popular songs, such as tabs or chords. These websites often provide multiple, conflicting chord label sequences for the same song. The availability of these large amounts of data poses the interesting problem of how to combine the knowledge from different sources to obtain better, and more reliable data. In this research, we address the problem of finding the most appropriate chord label sequence for a piece out of conflicting chord label sequences. Because the correctness of chord labels is hard to define (see e.g. [26]), we define "appropriate" in the context of this research as agreeing with a ground truth. An example of another evaluation context could be user satisfaction.

© Hendrik Vincent Koops, W. Bas de Haas, Dimitrios Bountouridis, Anja Volk. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Hendrik Vincent Koops, W. Bas de Haas, Dimitrios Bountouridis, Anja Volk. "Integration and Quality Assessment of Heterogeneous Chord Sequences using Data Fusion", 17th International Society for Music Information Retrieval Conference, 2016.

A pivotal problem for integrating data from different sources is determining which source is more trustworthy. Assessing the trustworthiness of a source from its data is a non-trivial problem. Web sources often supply an external quality assessment of the data they provide, for example through user ratings (e.g. three or five stars), or popularity measurements such as search engine page rankings. Unfortunately, Macrae et al. have shown in [18] that no correlation was found with the quality of tabs and user ratings or search engine page ranks. They propose that a better way to assess source quality is to use features such as the agreement (concurrency) between the data. Naive methods of assessing source agreement are often based on the assumption that the value provided by the majority of the sources is the correct one. For example, [1] integrates multiple symbolic music sequences that originate from different optical music recognition (OMR) algorithms by picking the symbol with the absolute majority at every position in the sequences. It was found that OMR may be improved using naive source agreement measures, but that substantial improvements may need more elaborate methods.

Improving results by combining the power of multiple algorithms is an active research area in the music domain, whether it is integrating the output of similar algorithms [28], or the integration of the output of different algorithms [15], such as the integration of features into a single feature vector to combine the strengths of multiple feature extractors [12, 19, 20]. Nevertheless, none of these deal with the integration and quality assessment of heterogeneous categorical data provided by different sources.

Recent advancements in data science have resulted in sophisticated data integration techniques falling under the umbrella term *data fusion*, in which the notion of source agreement plays a central role. We show that data fusion can achieve a more accurate integration than naive methods by estimating the trustworthiness of a source, compared to the more naive approach of just looking at which value is the most common among sources. To our knowledge no research into data fusion exists in the music domain. Re-

search in other domains has shown that data fusion is capable of assessing correct values with high precision, and significantly outperforms other integration methods [7,25].

In this research, we apply data fusion to the problem of finding the most appropriate chord label sequence for a piece by integrating heterogeneous chord label sequences. We use a method inspired by the ACCUCOPY model that was introduced by Dong et al. in [7, 8] to integrate conflicting databases. Instead of databases, we propose to integrate chord label sequences. With the growing amount of crowd-sourced chord label sequences online, integration and quality assessment of chord label sequences are important for a number of reasons. First, finding the most appropriate chord labels from a large amount of possibly noisy sources by hand is a very cumbersome process. An automated process combining the shared knowledge among sources solves this problem by offering a high quality integration. Second, to be able to rank and offer high quality data to their users, websites offering conflicting chord label data need a good way to separate the wheat from the chaff. Nevertheless, as was argued above, both integration and quality assessment have shown to be hard problems.

To measure the quality of chord label sequence integration, we propose to integrate the outputs of different MIREX Audio Chord Estimation (ACE) algorithms. We chose this data, because it offers us the most reliable ground truth information, and detailed analysis of the algorithms to make a high quality assessment of the integrated output. Our hypothesis is that through data fusion, we can create a chord label sequence that is significantly better in terms of comparison to a ground truth than the individual estimations. Secondly, we hypothesize that the results of integrated chord label sequences have a lower standard deviation on their quality, hence are more reliable.

Contribution. The contribution of this paper is threefold. First, we show the first application of data fusion in the domain of symbolic music. In doing so, we address the question how heterogeneous chord label sequences describing a single piece of music can be combined into an improved chord label sequence. We show that data fusion outperforms majority voting and random picking of source values. Second, we show how data fusion can be used to accurately estimate the relative quality of heterogeneous chord label sequences. Data fusion is better at capturing source quality than the most frequently used source quality assessment methods in multiple sequence analysis. Third, we show that our purely data-driven method is capable of capturing important knowledge shared among sources, without incorporating domain knowledge.

Synopsis. The remainder of this paper is structured as follows: Section 2 provides an introduction to data fusion. Section 3 details how integration of chord label sequences using data fusion is evaluated. Section 4 details the results of integrating submissions of the MIREX 2013 automatic chord extraction task. The paper closes with conclusions and a discussion, which can be found in Section 5.

2. DATA FUSION

We investigate the problem of integrating heterogeneous chord label sequences using data fusion. Traditionally, the goal of data fusion is to find the correct values within autonomous and heterogeneous databases (e.g. [9]). For example, if we obtain meta-data (fields such as year, composer, etc) from different web sources of the song "Black Bird" by The Beatles, there is a high probability that some sources will contradict each other on some values. Some sources will attribute the composer correctly to "Lennon - McCartney", but others will provide just "McCartney", "McCarthey", etc. Typos, malicious editing, data corruption, incorrectly predicted values, and human ignorance are some of the reasons why sources are hardly ever error-free.

Nevertheless, if we assume that most of the values that sources provide are correct, we can argue that values that are shared among a large amount of sources are often more *probable* to be correct than values that are provided by only a single source. Under the same assumption, we can also argue that sources that agree more with other sources are more *accurate*, because they share more values that are likely to be correct. Therefore, if a value is provided by only a single but very accurate source, we can prefer it over values with higher probabilities from less accurate sources, the same way we are more open to accepting a deviating answer from a reputable source in an everyday discussion.

In the above examples, we assume that each source is independent. In real-life this is rarely the case: information can be copied from one website to the other, students repeat what their teacher tells them and one user can enter the same values in a database twice, which can lead to inappropriate values being copied by a large number of sources: *"A lie told often enough becomes the truth"* (Lenin [1]) [8]. Intuitively, we can predict the *dependency* of sources from their sharing of inappropriate values. In general, inappropriate values are assumed to be uniformly distributed, which implies that sharing a couple of identical inappropriate values is a rare event. For example, the rare event of two students sharing a number of identical inappropriate answers on an exam is indicative of copying from each other. Therefore, by analyzing which values with low probabilities are shared between sources, we can calculate a probability of their dependence.

In this research, instead of using databases, we address these issues through data fusion on heterogeneous chord label sequences. Our goal is to take heterogeneous chord label sequences of the same song and create a chord label sequence that is better than the individual ones. We take into account: 1) the *accuracy* of sources, 2) the probabilities of the values provided by sources, and 3) the probability of *dependency* between sources. In the following sections, we refer to different versions of the same song as *sources*, each providing a sequence of values called *chord labels*. See Table 1 for an example, showing four sources ($S_{0...3}$), each providing a sequence of three chord labels, and FUSION, an example of data fusion output.

[1] Ironically, this quote's origin is unclear, but *most* sources cite Lenin.

S_0	C:maj	A:min	A:min	F:maj
S_1	C:maj	F:maj	G:maj	F:maj
S_2	C:maj	F:maj	A:min	D:min
S_3	C:maj	F:maj	A:min	D:min
MV	C:maj	F:maj	A:min	?
DF	C:maj	F:maj	A:min	D:min

Table 1: Example of four sources ($S_{0...3}$) providing different chord label sequences for the same song. DF shows an example output of data fusion on these sources. DF is identical to majority vote (MV) on the first three chord labels. For the last chord label, DF chooses D:min by taking into account source accuracy, while majority vote would randomly pick either F:maj or D:min.

2.1 Source Accuracy

By taking into account the accuracy of a source, we can deal with issues that arise from simple majority voting. For example in Table 1, the final chord labels in the sequence (F:maj and D:min) are provided by the same number of sources. Solving which chord to choose here would require picking randomly one of the two, or using auxiliary knowledge such as harmony theory to make a good choice.

Another problem is that sometimes a source can provide an appropriate chord label that contradicts all other sources. Majority vote would assign the lowest probability to this chord, although it might come from a source that overall agrees a lot with other sources. Intuitively, we have more trust in a source that we believe is more accurate, which is implemented as follows. The chord labels of a source are weighted according to the overall performance of that source: if a source provides a large number of values that agree with other sources, we consider it to be more accurate and more trustworthy, and vice versa.

The accuracy of a source is defined by Dong et al. in [7] as follows. We calculate source accuracy by taking the arithmetic mean of the probabilities of all chord labels the source provides. As an example, suppose we estimate the probabilities of the chords in Table 1 based on their frequency count (c.q. likelihood). That is, C:maj for the first column is 1, A:min for the second column is $1/4$, etc. Then, if we take the average of the chord label probabilities of the first source in our example of Table 1 we can calculate the source accuracy $A(S_0)$ of S_0 as follows:

$$A(S_0) = \frac{1 + 1/4 + 3/4 + 1/2}{4} = 0.625 \qquad (1)$$

In the same way, we can calculate the source accuracies for the other three sources which are 0.625, 0.75 and 0.75 for S_1,S_2 and S_3 respectively.

Assuming that the sources are independent, then the probability that a source provides an appropriate chord label is its source accuracy. Conversely, the probability that a source provides an inappropriate chord is the fraction of the inverse of the source accuracy over all possible inappropriate values n: $\frac{(1-A(S))}{n}$. For example, for major and minor chord labels we have 12 roots and 2 modes, which means that for every correct chord label there are $n = (12 * 2) - 1 = 23$ inappropriate chord labels. With more complex chord labels (sevenths, added notes, inversions), n increases combinatorially.

The chord labels of sources with higher accuracies will be more likely to be selected through the use of *vote counts*, which are used as weights for the probabilities of the chord labels they provide. With n and $A(S_i)$ we can derive a vote count $VS(S_i)$ of a source S_i. The vote count of a source is computed as follows:

$$VS(S_i) = \ln \frac{nA(S_i)}{1 - A(S_i)} \qquad (2)$$

Applied to our example, this results in vote counts of 2.62 for S_0 and S_1, and 2.80 for S_2 and S_3. The higher vote count for S_2 and S_3 means that its values are more likely to be appropriate than those of S_0 and S_1.

2.2 Chord Label Probabilities

After having defined the accuracy of a source, we can now determine which chord labels provided by all the sources are most likely the appropriate labels, by taking into account source accuracy. In the computation of chord label probabilities we take into account a) the number of sources that provide those chord labels and b) the accuracy of their sources. With these values we calculate the vote count $VC(\mathcal{L})$ of a chord label $\mathcal{L}$, which is computed as the sum of the vote counts of its providers:

$$VC(\mathcal{L}) = \sum_{\sigma \in S^{\mathcal{L}}} VS(\sigma) \qquad (3)$$

where $S^{\mathcal{L}}$ is the set of all sources that provide the chord label $\mathcal{L}$. For example, for the vote count of F:maj in the last column of the example in Table 1, we take the sum of the vote counts of S_0 and S_1. For the vote count of D:min we take the sum of the vote counts of S_2 and S_3. To calculate chord label probabilities from chord label vote counts, we take the fraction of the chord label vote count and the chord label vote counts of all possible chord labels (D):

$$P(\mathcal{L}) = \frac{exp(\,VC(\mathcal{L}))}{\Sigma_{l \in D}\ exp(\,VC(l))} \qquad (4)$$

Applied to our example from Figure 1, we see that solving this equation for F:maj results in a probability of $P(\texttt{F:maj}) \approx 0.39$, and for D:min results in a probability of $P(\texttt{D:min}) \approx 0.56$. Instead of having to choose randomly as would be necessary in a majority vote, we can now see that D:min is more probable to be the correct chord label, because it is provided by sources that are overall more trustworthy.

2.3 Source Dependency

In the sections above we assumed that all sources are independent. This is not always the case when we deal with real-world data. Often, sources derive their data from a common origin, which means there is some kind of dependency between them. For example, a source can copy chord labels from another source before changing some labels, or some Audio Chord Estimation (ACE) algorithm can estimate multiple (almost) equal chord label sequences with different parameter settings. This can create a bias in computing appropriate values. To account for the bias that can arise from source dependencies, we weight the values of sources we suspect to have a dependency lower. In a sense, we award independent contributions from sources

and punish values that we suspect are dependent on other sources.

In data fusion, we can detect source dependency directly from the data by looking at the amount of shared uncommon (rare) chord labels between sources. The intuition is that sharing a large number of uncommon chord labels is evidence for source dependency. With this knowledge, we can compute a weight $I(S_i, \mathcal{L})$ for the vote count $VC(\mathcal{L})$ of a chord label $\mathcal{L}$. This weight tells us the probability that a source S_i provides a chord label $\mathcal{L}$ independently.

2.4 Solving Catch-22: Iterative Approach

The chord label probabilities, source accuracy and source dependency are all defined in terms of each other, which poses a problem for calculating these values. As a solution, we initialize the chord label probabilities with equal probabilities and iteratively compute source dependency, chord label probabilities and source accuracy until the chord label probabilities converge or oscillation of values is detected. The resulting chord label sequence is composed of the chord labels with the highest probabilities.

For detailed Bayesian analyses of the techniques mentioned above we refer to [7,10]. With regard to the scalability of data fusion, it has been shown that DF with source dependency runs in polynomial time [7]. Furthermore, [17] propose a scalability method for very large data sets, reducing the time for source dependency calculation by two to three orders of magnitude.

3. EXPERIMENTAL SETUP

To evaluate the improvement of chord label sequences using data fusion we use the output of submissions to the Music Information Retrieval Evaluation eXchange (MIREX) Audio Chord Estimation (ACE) task. For the task, participants extract a sequence of chord labels from an audio music recording. The task requires the estimation chord labels sequences that include the full characterization of chord labels (root, quality, and bass note), as well as their chronological order, specific onset times and durations.

Our evaluation uses estimations from twelve submissions for two Billboard datasets (Section 3.1). Each of these estimations is sampled at a regular time interval to make them suitable for data fusion (Section 3.2). We transform the chord labels of the sampled estimations to different representations (root only, major/minor and major/minor with sevenths) (Section 3.3) to evaluate the integration of different chord types. The sampled estimations are integrated using data fusion per song. To measure the quality of the data fusion integration, we calculate the Weighted Chord Symbol Recall (WCSR) (Section 3.4).

3.1 Billboard datasets

We evaluate data fusion on chord label estimations for two subsets of the Billboard dataset [2], which was introduced by Burgoyne et al. in [3]. The Billboard dataset contains time-aligned transcriptions of chord labels from songs that appeared in the *Billboard* "Hot 100" chart in the United States between 1958 and 1991. All transcriptions are annotated by trained jazz musicians and verified by independent music experts. For the MIREX 2013 ACE task, two subsets of the Billboard dataset were used: the 2012 Billboard set (BB12) and the 2013 Billboard (BB13) set. BB12 contains chord label annotations for 188 songs, corresponding to entries 1000—1300 in the Billboard set. BB13 contains the annotations for 188 different songs: entries 1300—1500.

Twelve teams participated for both datasets, some with multiple submissions: CB3 & CB4 [5], CF2 [4], KO1 & KO2 [16], NG1 & NG2 [13], NMSD1 & NMSD2 [21], PP3 & PP4 [22], and SB [27]. Their submissions are used to evaluate data fusion, for which the Billboard annotations serve as a ground truth.

3.2 Sampling

The MIREX ACE task requires teams to not only estimate *which* chord labels appear in a song, but also *when* they appear. Because of differences in approaches, timestamps of the estimated chord labels do not necessarily agree between teams. This is a problem for data fusion, which expects an equal length and sampling rate of the sources that will be integrated. As a solution, we sample the estimations at a regular interval.

In the past, MIREX used a 10 millisecond sampling approach to calculate the quality of an estimated chord label sequence. Since MIREX 2013, the ground-truth and estimated chord labels are viewed as continuous segmentations of the audio [23]. Because of our data constraint, we use the pre-2013 10 millisecond sampling approach. An initial evaluation using different sampling frequencies in the range 0.1 millisecond to 0.5 seconds, we found only minor differences in data fusion output. The estimated chord label sequences are sampled per song from each team, and used as input to the data fusion algorithm.

3.3 Chord Types

The MIREX ACE task is evaluated on different chord types. To accurately compare our results with those of the teams, and to investigate the effect of integrating different chord types, we follow the chord vocabulary mappings that were introduced by [23] and are standardized in the MIREX evaluation. We map the sampled sequences of estimated chord labels into three chord vocabularies before applying data fusion: root notes only (R), major/minor only chords (MM), and major/minor with sevenths (MM7).

Note that the MIREX 2013 evaluation also includes major/minor with inversions and major/minor seventh chords with inversions. Since there are only two teams that estimated inversions we did not take these into account in our evaluation.

3.4 Evaluation

From the data fusion output sequences for all songs, we calculate the Weighted Chord Symbol Recall (WCSR). The WCSR reflects the proportion of correctly labeled chords in a single song, weighted by the length of the song [14, 23]. To measure the improvement of data fusion, we compare

[2] available from http://ddmal.music.mcgill.ca/billboard

its WCSR with the WCSR of the best scoring team. In addition to data fusion, we compute baseline measurements. We compare the data fusion results with a majority vote (MV) and random picking (RND) technique.

For MV we simply take the most frequent chord label every 10 milliseconds. In case multiple chord labels are most frequent, we randomly pick from the most frequent chord labels. For the example in Table 1, the output would be either `C:maj, F:maj, A:min, F:maj` or `C:maj, F:maj, A:min, D:min`. For RND we select a chord from a random source every 10 milliseconds. For the example in Table 1, RND essentially picks one from 4^4 possible chord label combinations by picking a chord label from a randomly chosen source per column.

4. RESULTS

We are interested in obtaining *improved*, *reliable* chord sequences from *quality assessed* existing estimations. Therefore, we analyze our results in three ways. Firstly, to measure improvement, we show the difference in WCSR between the best scoring team and RND, MV and DF. This way, we can analyze the performance increase (or decrease) for each of these integration methods. The differences are visualized in Figure 1 for the BB12 and BB13 datasets. For each of the three methods, it shows the difference in WCSR for root notes R major/minor only chords MM, and major/minor + sevenths chords (MM7). For detailed individual results an analyses of the teams on both datasets, we refer to [2] and MIREX. [3]

Secondly, to measure the reliability of the integrations, we analyze the standard deviation of the scores of MV and DF. We leave RND out of this analysis because of its poor results. The ideal integration should have 1) a high WCSR and 2) a low standard deviation, because this means that the integration is 1) good and 2) reliable. Table 2 shows the difference with the average standard deviation of the teams. Sections 4.1 - 4.2 report the results in WCSR difference and standard deviation.

Thirdly, in Section 4.3 we analyze the correlation between source accuracy and WCSR, and compare the correlation with other source quality assessments. These correlations will tell us to which extent DF is capable of assessing the quality of sources compared to other, widely used multiple sequence analysis methods.

[3] http://www.music-ir.org/mirex/wiki/2013:MIREX2013_Results

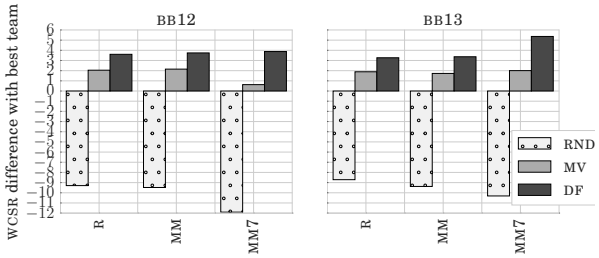


Figure 1: Difference in WCSR with best team for random picking (RND), majority vote (MV) and data fusion (DF). R = root notes, MM = major/minor chords and MM7 = major/minor + sevenths.

	BB12			BB13		
	R	MM	MM7	R	MM	MM7
DF	**-2.5**	**-2.8**	**-2.2**	**-0.5**	**-0.9**	**-1.8**
MV	-1.4	-1.8	-0.97	-0.3	-0.4	-1

Table 2: Difference in standard deviation for DF and MV compared to the average standard deviation of the teams. Lower is better, best values are bold.

4.1 Results of Integrating R, MM and MM7

The left hand sides of the triple-bar groups in Figure 1 show that for both BB12 and BB13, RND performs the worst among RND, MV and DF. RND decreases the WCSR between 8.7% and 12% point, compared to the best performing teams (CB3 and KO1 for BB12 and BB13 respectively) for all chord types. This means that picking random values from sources does not capture shared knowledge in a meaningful way. The middle bars in Figure 1 show that MV integrates knowledge better than RND. MV moderately improves the best algorithm with a difference between 0.6% and 2.1% point.

The right hand sides of the bar groups in Figure 1 show that in both datasets and in all chord types, DF outperforms all other methods with an increase between 3.6% point and 5.4% point compared to the best team. We tested the scores of RND, MV and DF and the best performing teams using a Friedman test for repeated measurements, accompanied by Tukeys Honest Significant Difference tests for each pair of algorithms. We find that DF significantly outperforms the best submission, RND and MV on all datasets on all datasets ($p < 0.01$). These results combined show that DF is capable of capturing knowledge shared among sources needed to outperform all other methods.

In Table 2, we find that for both BB12 and BB13, both MV and DF decrease the standard deviation compared to the average standard deviation of the teams. In fact, we find that DF outperforms MV, improving the standard deviation by a factor two compared to MV. Together, these results mean that on average, DF creates the best sequences with the least errors for all datasets and all chord types.

4.2 Influence of Chord Types on Integration

The results detailed above show that DF is not only capable to significantly outperform all other tested methods on all tested chord labels types, but also produces the most reliable output, because of the low standard deviation.

Comparing the RND, MV and DF results between chord types in Figure 1, we see that the WCSR of RND decreases with a larger chord vocabulary. Because specificity increases the probability of random errors for any algorithm, the probability that RND will pick a good chord label randomly goes down with an increase of the chord vocabulary. For MV, we see that the results are somewhat stable with an increase of the chord vocabulary. Nevertheless, MV is also sensitive for randomly matching chord labels, which explains the drop in accuracy for MM7 for BB13 on the left hand side of Figure 1. Most interestingly, we observe that the performance of DF increases with a larger chord vocabulary. The explanation is that specificity helps DF to separate good sources from bad sources. With a larger chord vocabulary, sources will agree with each other on

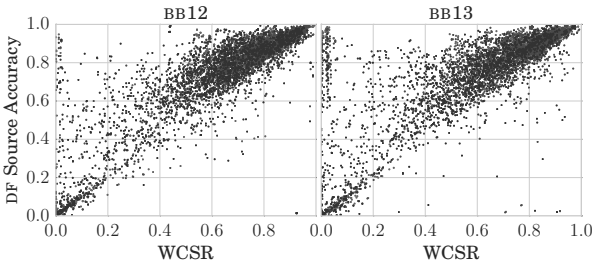


Figure 2: Correlation between WCSR and source accuracy. Plotted are R, MM and MM7. One dot is one estimated chord label sequence for one song from one team.

more specific chord labels, which decreases the probability of unwanted random source agreement.

4.3 Source Quality Assessment

The previous sections show that data fusion is capable of selecting good chord labels from the coherence between the sources, without ground truth knowledge. A pivotal part of data fusion is the computation of source accuracy, which provides a relative score for each source compared to the other sources. There are circumstances in which we are more interested in the estimation of source accuracy than the actual integration of source data. For example, ranking a number of different crowd sourced chord label sequences of the same song obtained from web sources, (e.g. investigated by [18]). Investigating the relationship between source accuracy and the WCSR provides insight whether data fusion is capable of assessing the accuracy of the sources in a way that reflects WCSR. WCSR reflects the quality of the chord sequences and therefore the quality of the algorithm. This relationship is shown in Figure 2, in which the WCSR is plotted against the DF source accuracy.

Initial observation of Figure 2 shows that for both BB12 and BB13, WCSR and source accuracy are distributed along a more or less diagonal line, meaning that a higher WCSR is associated with a higher DF source accuracy, and vice versa. This indicates a strong correlation, which is confirmed by the Spearman's rank correlation coefficient (SRCC). To analyze the relative performance of source quality assessment of DF, we compare its correlation with widely used sequence scoring methods. These are often used in bioinformatics, where sequence ranking is at the root of a multitude of problems. Table 3 compares the SRCC of different similarity scoring methods for BB12 and BB13. The table shows the correlations between WCSR and DF, bigrams (BIGRAM), profile hidden Markov models (PHMM), percentage identity (PID), and neigbor-joining trees (NJT). BIGRAM compares the relative balance of specific character pairs appearing in succession, also known as bigrams. Sequences belonging to the same group should be stochastic products of the same probabilistic model [6]. PHMM turns the sources into a position-specific scoring system by creating a profile with position-probabilities. A source is scored through comparison with the profile of all other sources [11]. PID is the fraction of equal characters divided by the length of the source. NJT is a bottom-up clustering method for the creation of phylogenetic trees, in which the distance from the root is the score [24].

	BB12			BB13		
	R	MM	MM7	R	MM	MM7
DF	**0.87**	**0.85**	**0.82**	**0.77**	**0.77**	**0.76**
BIGRAM	0.18	0.18	0.16	0.2	0.22	0.29
PHMM [4]	0.22	—	—	0.22	—	—
PID	0.18	0.2	0.19	0.25	0.27	0.29
NJT	0.2	0.22	0.21	0.24	0.25	0.27

Table 3: Spearman's rank correlation coefficient (ρ) of WCSR and other source scoring methods. Best performing algorithms are bold. All values are significant with $p < 0.01$.

The table shows that DF source accuracy has the highest correlation with WCSR among all other methods. These results show that data fusion is capable of assessing the quality of the sources without any ground-truth knowledge in a way that is closely related to the actual source quality.

5. DISCUSSION AND CONCLUSION

Through this study, we have shown for the first time that using data fusion, we can integrate the knowledge contained in heterogeneous ACE output to create improved, and more reliable chord label sequences. Data fusion integration outperforms all individual ACE algorithms, as well as majority voting and random picking of source values. Furthermore, we have shown that with data fusion, one can not only generate high quality integrations, but also accurately estimate the quality of sources from their coherence, without any ground truth knowledge. Source accuracy outperforms other popular sequence ranking methods.

Our findings demonstrate that knowledge from multiple sources can be integrated effectively, efficiently and in an intuitive way. Because the proposed method is agnostic to the domain of the data, it could be applied to melodies or other musical sequences as well. We believe that further analysis of data fusion in crowd-sourced data has the potential to provide non-trivial insights into musical variation, ambiguity and perception. We believe that data fusion has many important applications in music information retrieval research and in the music industry for problems relating to managing large amounts of crowd-sourced data.

Acknowledgements We thank anonymous reviewers for providing valuable comments on an earlier draft on this text. H.V. Koops and A. Volk are supported by the Netherlands Organization for Scientific Research, through the NWO-VIDI-grant 276-35-001 to A. Volk. D. Bountouridis is supported by the FES project COMMIT/.

6. REFERENCES

[1] E.P. Bugge, K.L. Juncher, B.S. Mathiesen, and J.G. Simonsen. Using sequence alignment and voting to improve optical music recognition from multiple recognizers. In *Proc. of the International Society for Music Information Retrieval Conference*, pages 405–410, 2011.

[2] J.A. Burgoyne, W.B. de Haas, and J. Pauwels. On comparative statistics for labelling tasks: What can we learn from MIREX ACE 2013. In *Proc. of the 15th*

[4] The MM and MM7 chord label alphabets are too large for the used PHMM application, which only accepts a smaller bioinformatics alphabet.

Conference of the International Society for Music Information Retrieval, pages 525–530, 2014.

[3] J.A. Burgoyne, J. Wild, and I. Fujinaga. An expert ground truth set for audio chord recognition and music analysis. In *Proc. of the International Society for Music Information Retrieval Conference*, volume 11, pages 633–638, 2011.

[4] C. Cannam, M. Mauch, M.E.P. Davies, S. Dixon, C. Landone, K. Noland, M. Levy, M. Zanoni, D. Stowell, and L.A. Figueira. MIREX 2013 entry: Vamp plugins from the centre for digital music, 2013.

[5] T. Cho and J.P. Bello. MIREX 2013: Large vocabulary chord recognition system using multi-band features and a multi-stream hmm. *Music Information Retrieval Evaluation eXchange (MIREX)*, 2013.

[6] M.J. Collins. A new statistical parser based on bigram lexical dependencies. In *Proc. of the 34th annual meeting on Association for Computational Linguistics*, pages 184–191. Association for Computational Linguistics, 1996.

[7] X.L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: the role of source dependence. *Proc. of the VLDB Endowment*, 2(1):550–561, 2009.

[8] X.L. Dong and F. Naumann. Data fusion: resolving data conflicts for integration. *Proc. of the VLDB Endowment*, 2(2):1654–1655, 2009.

[9] X.L. Dong and D. Srivastava. Big data integration. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 1245–1248. IEEE, 2013.

[10] X.L. Dong and D. Srivastava. Big data integration. *Synthesis Lectures on Data Management*, 7(1):1–198, 2015.

[11] S.R. Eddy. Profile hidden markov models. *Bioinformatics*, 14(9):755–763, 1998.

[12] R. Foucard, S. Essid, M. Lagrange, G. Richard, et al. Multi-scale temporal fusion by boosting for music classification. In *Proc. of the International Society for Music Information Retrieval Conference*, pages 663–668, 2011.

[13] N. Glazyrin. Audio chord estimation using chroma reduced spectrogram and self-similarity. *Music Information Retrieval Evaluation Exchange (MIREX)*, 2012.

[14] C. Harte. *Towards automatic extraction of harmony information from music signals*. PhD thesis, Department of Electronic Engineering, Queen Mary, University of London, 2010.

[15] A. Holzapfel, M.E.P. Davies, J.R. Zapata, J.L. Oliveira, and F. Gouyon. Selective sampling for beat tracking evaluation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(9):2539–2548, 2012.

[16] M. Khadkevich and M. Omologo. Time-frequency reassigned features for automatic chord recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 181–184. IEEE, 2011.

[17] X. Li, X.L. Dong, K.B. Lyons, W. Meng, and D. Srivastava. Scaling up copy detection. *arXiv preprint arXiv:1503.00309*, 2015.

[18] R. Macrae and S. Dixon. Guitar tab mining, analysis and ranking. In *Proc. of the International Society for Music Information Retrieval Conference*, pages 453–458, 2011.

[19] A. Meng, P. Ahrendt, and J. Larsen. Improving music genre classification by short time feature integration. In *Acoustics, Speech, and Signal Processing, 2005. Proc..(ICASSP'05). IEEE International Conference on*, volume 5, pages v–497. IEEE, 2005.

[20] A. Meng, J. Larsen, and L.K. Hansen. *Temporal feature integration for music organisation*. PhD thesis, Technical University of Denmark, Department of Informatics and Mathematical Modeling, 2006.

[21] Y. Ni, M. Mcvicar, R. Santos-Rodriguez, and T. De Bie. Harmony progression analyzer for MIREX 2013. *Music Information Retrieval Evaluation eXchange (MIREX)*.

[22] J. Pauwels, J-P. Martens, and G. Peeters. The ircamkeychord submission for MIREX 2012.

[23] J. Pauwels and G. Peeters. Evaluating automatically estimated chord sequences. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 749–753. IEEE, 2013.

[24] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425, 1987.

[25] N.T. Siebel and S. Maybank. Fusion of multiple tracking algorithms for robust people tracking. In *Computer VisionECCV 2002*, pages 373–387. Springer, 2002.

[26] J.B.L Smith, J.A. Burgoyne, I. Fujinaga, D. De Roure, and J.S. Downie. Design and creation of a large-scale database of structural annotations. In *Proc. of the International Society for Music Information Retrieval Conference*, volume 11, pages 555–560, 2011.

[27] Nikolaas Steenbergen and John Ashley Burgoyne. Joint optimization of an hidden markov model-neural network hybrid for chord estimation. *MIREX-Music Information Retrieval Evaluation eXchange. Curitiba, Brasil*, pages 189–190, 2013.

[28] C. Sutton, E. Vincent, M. Plumbley, and J. Bello. Transcription of vocal melodies using voice characteristics and algorithm fusion. In *2006 Music Information Retrieval Evaluation eXchange (MIREX)*, 2006.

LANDMARK-BASED AUDIO FINGERPRINTING FOR DJ MIX MONITORING

Reinhard Sonnleitner[1], **Andreas Arzt**[1], **Gerhard Widmer**[1,2]
Department of Computational Perception, Johannes Kepler University, Linz, Austria[1]
Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria[2]
reinhard.sonnleitner@jku.at

ABSTRACT

Recently, the media monitoring industry shows increased interest in applying automated audio identification systems for revenue distribution of DJ performances played in discotheques. DJ mixes incorporate a wide variety of signal modifications, e.g. pitch shifting, tempo modifications, cross-fading and beat-matching. These signal modifications are expected to be more severe than what is usually encountered in the monitoring of radio and TV broadcasts. The monitoring of DJ mixes presents a hard challenge for automated music identification systems, which need to be robust to various signal modifications while maintaining a high level of specificity to avoid false revenue assignment. In this work we assess the fitness of three landmark-based audio fingerprinting systems with different properties on real-world data – DJ mixes that were performed in discotheques. To enable the research community to evaluate systems on DJ mixes, we also create and publish a freely available, creative-commons licensed dataset of DJ mixes along with their reference tracks and song-border annotations. Experiments on these datasets reveal that a recent quad-based method achieves considerably higher performance on this task than the other methods.

1. INTRODUCTION

Automated audio identification systems, also referred to as audio fingerprinters, identify a piece of query audio from a collection of known reference audio pieces. In general, such systems search for characteristic features in the query audio, which are then compared to features of known audio pieces. The features are the so-called fingerprints, which should embody a favourable trade-off in storage demands, computation complexity, comparability, specificity, and robustness. The importance of the individual properties of the fingerprints is dictated by the use case. The industry uses audio identifications systems to monitor radio and TV broadcast channels to create detailed lists of the specific content that was played at any given time. In addition to radio and TV broadcast monitoring, performance rights organizations show interest in monitoring music performances, for example in discotheques. Without using automated identification systems, royalty collection depends on the broadcasters who are expected to create detailed lists of played content.

Musical content that is played in discotheques is usually performed by DJs, who can introduce severe signal modifications by mixing sets of songs in a homogeneous fashion. This frequently involves temporally changing the pitch or tempo of the audio to achieve a smooth transition from one track to the other, and often DJs will add effects in response to the mood or atmosphere in the club.

Signal content that is modified by DJs arguably puts enormous robustness demands on automated systems. It seems hard to quantify the type and severity of signal manipulations that can be introduced by DJs, as several effects can be applied in combination. For the same reason we believe it is hard to manually create meaningful test cases that reflect the possible modifications for system evaluation.

In this work, we investigate the fitness and performance of systems that belong to the class of so-called *landmark-based* audio fingerprinting methods. Landmark-based systems extract highly robust feature points, i.e. local energy maxima, from the two dimensional time-frequency representation of the audio signal, and combine groups of these landmarks to form the individual fingerprints.

We show via experiments that it is hard to achieve accurate results on DJ mixes. To do this, we test three implementations with different robustness properties, and report on their abilities to correctly identify known audio pieces while correctly abstaining from reporting a match if the correct song is not contained in the given reference database.

While the algorithmic approaches that we use in this work are extensively evaluated in the literature, we show that the application to DJ mixes indeed unveils shortcomings, specifically in the ability to prevent false detections. In the context of media monitoring, falsely detecting a song can lead to incorrect royalty management.

We contribute a new dataset which poses difficulties to automated identification systems, and investigate the different properties of three landmark based systems via experiments on these datasets.

The paper is organized as follows. Section 2 discusses prior and related work, in Section 3 we introduce the

© Reinhard Sonnleitner[1], Andreas Arzt[1], Gerhard Widmer[1,2]. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Reinhard Sonnleitner[1], Andreas Arzt[1], Gerhard Widmer[1,2]. "Landmark-based Audio Fingerprinting for DJ Mix Monitoring", 17th International Society for Music Information Retrieval Conference, 2016.

datasets that are the basis for the experiments and analysis and interpretation of results. Section 4 gives an overview of the methods we test in this work. Then, in Section 5 we describe the setup of experiments and their evaluation. An analysis of the different properties of the tested methods is given in Section 6. Finally, in Section 7 we conclude our work.

2. RELATED WORK

The field of audio fingerprinting enjoys high research activity and numerous systems are described in the literature that approach the task [6, 7, 10, 12, 14, 15, 18]. Excellent reviews of earlier systems are presented in [2, 3].

The system described in [13] achieves pitch scale change robustness to small scaling factors by describing content based on short term band energies. In addition, the system is robust to small time scale modifications.

The basic algorithm of the Shazam system, a well known representative method for landmark-based audio fingerprinting, is described in [18]. It pairs spectral peaks (i.e. the so-called landmarks) that are extracted from the audio to obtain compact hashes, which are used as index into a hashtable to search for matches. The fingerprints are highly robust to environmental noise and signal degradations that result from digital-analog conversions of audio.

Another system that achieves a certain amount of robustness to changes in playback speed is described in [4]. As the change of the playback speed of audio influences the pitch scale, a system is described that can mitigate this effect by first searching for common pitch offsets of query and reference pieces, and then rescaling the query accordingly. This system also is a member of landmark based identification methods.

The work described in [1, 19, 20] incorporates techniques from the domain of computer vision to audio identification. The authors of [1] apply a wavelet transform to signals and create compact bit vector descriptors of content that can be efficiently indexed via the Min-Hash algorithm. The approach shown in [20] uses the image retrieval and similarity approach by applying the SIFT [9] method on logarithmically scaled audio spectrograms, and later propose a matching method using LSH [5] in [19].

The concept of extracting features based on time-chroma patches from the time frequency representation of the audio to describe robust features for audio identification is discussed in [11].

We proposed to perform audio identification using compact scale invariant quad descriptors that are robust to time, pitch or speed modifications in [16], and later refined and extended that approach in [17].

The systems we use for the experiments in this work are described in Section 4.

3. DATA SETS

We perform experiments on two different datasets, called disco set, and mixotic set. In the following we introduce these datasets, and summarize their properties in Table 1.

Disco	tracks	ref.	$+[s]$	$-[s]$
set0	25	18	5661	2179
set1	12	12	3760	0
set2	12	11	3206	294
set3	11	4	1054	2006
set20	19	17	3123	457
set35	20	7	324	996
set36	28	13	872	768
set37	21	10	720	720
total: 8	148	92	18 720	7420
Mixotic	tracks	ref.	$+[s]$	$-[s]$
set044	14	14	4640	0
set123	12	12	3320	0
set222	18	11	3543	2097
set230	9	7	2560	780
set275	17	11	3398	1622
set278	12	11	3576	284
set281	18	15	3300	280
set282	14	8	2200	1740
set285	15	15	4540	0
set286	14	14	3140	0
total: 10	143	118	34 217	6803

Table 1: Data set properties of the disco set (top) and the mixotic set (bottom). The column "tracks" gives the number of played tracks in the DJ mix, "ref" denotes the number of these tracks that are present in the reference database, and the columns "$+[s]$, $-[s]$" hold the number of seconds of referenced audio and not-referenced audio for the individual DJ mixes.

The first dataset, the *disco set*, contains eight mixes that were performed in discotheques, and digitally recorded from the DJ mixing desk. The duration of the mixes is approximately 7 hours and 16 minutes. For this dataset we have 296 reference tracks, only some of which are actually played in the mixes. The genres of the mixes include pop and rock, electronic music and German folk.

Because of copyright reasons, we cannot make the disco set publicly available, therefore we compile a second dataset, called *mixotic set*. We created this dataset from free, CC-licensed DJ mixes that were published on the mixotic netlabel [1] , and collected their respective reference songs, which are available under the same license. The mixotic set consists of 10 mixes with a total duration of 11 hours and 23 minutes. For this dataset we collected a set of 723 reference tracks, 118 of which are actually played in the mixes. According to the artists, this set contains genres like Techno, Chicago House, Deep-Tech, Dub-Techno, Tech-House, and the like. To be able to evaluate the fingerprinting results, we annotated the song borders of the tracks that are played in the individual mixes. Due to the long fading regions and sometimes very homogeneous track transitions, these annotations cannot be exact. We tried to mark the positions in time where the

[1] Mixotic is accessible via http://www.mixotic.net.

previous track is fully faded out.

We think that the mixotic set may be useful to the research community, and could help to design well balanced identification systems and to uncover specific strengths and potential shortcomings of various methods, therefore we publish the mixotic set along with the annotations [2].

4. METHOD OVERVIEW

We use the datasets that we described in the previous section to experiment with the following three methods: *Audfprint*, *Panako* and the quad based audio fingerprinter, henceforth referred to as simply *Qfp*.

Audfprint Audfprint is a MIT-licensed implementation [3] of a landmark-based audio identification algorithm based on the method described in [18]. The published algorithm utilizes quantized hash fingerprints that represent pairs of spectral peaks. The hashes are described by the time-frequency position of the first peak and its distance in time and frequency to the second peak. The hashes that are computed from a snippet of query audio are used as the keys into a suitable reference data structure, e.g. a hash table, to retrieve reference hashes with the same key. For each query hash, a lookup is performed and the result sets are collected. Matched query and reference hashes which happen to have a constant time offset in their individual peak-time identify the reference audio, along with its position in which the query snippet could be located.

Panako Panako [15], available [4] under the GNU Affero General Public License is a free audio identification system. It transforms the time domain audio signal into a two dimensional time frequency representation using the Constant Q transform, from which it extracts event coordinates. Instead of peak pairs, the method uses triples, which allows for a hash representation that is robust to small time and pitch scale modifications of the query audio. Thus, the system can also report the scale change factors of the query audio with respect to the identified reference. The system is evaluated on queries against a database of 30 000 full length songs, and on this data set achieves perfect specificity while being able to detect queries that were changed in time or frequency scale of up to around 8%. In this work we use Version 1.4 of Panako.

Qfp The Qfp method [16, 17] is a landmark based method that is robust to time and pitch scale changes of query audio. Its evaluation shows high average accuracy of more than 95% and an average precision of 99% on queries that are modified in pitch and/or time scale by up to $\pm 30\%$. The evaluation is performed on a reference data base consisting of 100 000 full length songs. The average query run time is under two seconds for query snippets of 20 seconds in length. The system also correctly uncovers any underlying scale changes of query audio. While some robust audio identification systems are using methods from the field of computer vision (c.f. Section 2), Qfp is inspired by a method used in astronomy [8], which proposes to use n-tuples (with $n > 2$) of two dimensional point coordinates to describe continuous feature descriptors that are invariant to rotation and *isotropic* scaling. The Qfp method adapts the described findings to represent *non-isotropic*-scale invariant features that allow for robust and efficient audio identification. The system uses range queries against a spatial data structure, and a subsequent verification stage to reliably discard false matches. The verification process accepts matches within individual match sequences if spectral peaks in a region around the candidate match in the reference audio are also present in the query audio excerpt. Evaluation results of the Qfp method along with a parameter study and resulting run times are given in [17].

These methods are well performing identification systems. An evaluation of experiments using Audfprint and Panako is given in [15]. While all three methods are landmark-based, the systems employ different inner mechanisms and thus are expected to perform differently on the datasets used in this work. Note that we use Audfprint and Panako as published, without tuning to the task at hand. We do this because we believe that the methods are published with a set of standard parameters that turned out to be well suited for general use cases according to experimentation performed by their authors. Likewise, we also use the same set of parameters for Qfp, as they are described in [17]. We incorporated improvements for runtime, but these do not have any impact on the identification results at all. For the task at hand, we want to investigate the fitness of the underlying algorithms of the methods, rather than discussing their specific implementations.

5. EXPERIMENT SETUP

Experiments are performed individually on the datasets we described in Section 3. The general experimental setup is as follows. The mixes are split into non-overlapping query snippets of 20 seconds in length. To create query snippets from the DJ mix we use the tool SoX [5] along with switches to prevent clipping, and convert the snippets into .wav files.

The methods process each query snippet and store the results. The implementations of the three tested systems behave differently in answering a query: if the query excerpt could be matched, Audfprint and Panako by default report the whole query duration as matched sequence. Qfp gives a more detailed answer and reports the start time and end time of the matched portion within the query excerpt. Likewise, as Qfp, Audfprint allows to report the exact part of the query that it could actually match (using the option `--find-time-range`), but for Panako we did not find such an option. For best comparability of the evaluation results, for all of the three methods we assign the reported match file ID to its whole query of 20 seconds.

[2] Available on http://www.cp.jku.at/datasets/fingerprinting/

[3] Audfprint is available on https://github.com/dpwe/audfprint.

[4] Panako is available on http://www.panako.be/.

[5] SoX is available on http://sox.sourceforge.net/.

Dataset				Referenced					not ref.		
	$+[s]$	$-[s]$	M.	TP	FP	FN	**acc.**	**prec.**	TN	FP	**spec.**
Disco.	18720	7420	**A**	7838	7440	3442	0.419	0.513	3611	3809	0.487
			P	4624	5596	8500	0.247	0.452	5539	1881	0.746
			Q	13879	1253	3588	**0.741**	**0.917**	6996	424	**0.942**
			Q^{v}	14316	1523	2881	0.765	0.904	6587	833	0.888
			Q^{ϵ}	3423	152	15145	0.183	0.957	7413	7	0.999
Mixotic	34 217	6803	**A**	21783	10233	2201	0.637	0.680	1735	5068	0.255
			P	12326	16181	5710	0.360	0.432	2371	4432	0.349
			Q	29985	1262	2970	**0.876**	**0.959**	6304	499	**0.927**
			Q^{v}	30445	1680	2092	0.889	0.948	4395	2408	0.647
			Q^{ϵ}	19497	349	14371	0.570	0.982	6715	88	0.987

Table 2: Evaluation results for the data sets. The column "+" shows the number of seconds of the DJ mix, for which a reference is present. The column "−" likewise gives the number of seconds for which no reference track is present in the database. The methods (M.) Audfprint, Panako and Qfp are abbreviated as "A", "P" and "Q". The column "Q^{v}" shows Qfp results without the verification stage, and "Q^{ϵ}" shows the results for reduced search neighbourhood. "acc." is the accuracy, "prec." is the precision and "spec." is the specificity. The experiment setup and the meaning of the measures is defined in Section 5. Because of space constraints we omit showing the individual statistics of each DJ mix that is contained in the dataset, and directly present the overall values. Detailed results are available in the published dataset.

It is important to note that we do not perform smoothing over time on the individual results but rather test the *raw* identification performance of each method based on each individual query.

We compare the fingerprinting results to the ground truth on a one second basis, i.e. for each second of the DJ mix we check whether the corresponding query result is correct.

Here we distinguish the following two cases: Case 1 $(C1)$ *identifiable*, and Case 2 $(C2)$ *not identifiable* portions of the mixes. We investigate how the systems perform in cases where a song is identifiable, because it is present in the reference database $(C1)$, and how well behaving a system is in not producing a match result in cases where this is correct, i.e. because the track is in fact not present in the reference $(C2)$.

For all cases $(C1)$, we count the number of seconds of true positives (TP), false positives (FP) and false negatives (FN). True positives are cases in which the system correctly identified a track from a query. The false positives denote situations in which the wrong track is claimed to be present, and the false negatives are cases in which the system did not report a result at all. For this evaluation there exist no true negatives, i.e. $TP+FP+FN = N$. For this case $(C1)$ we define the following two performance measures.

Accuracy, as the proportion of correctly identified queries:

$$Accuracy = \frac{TP}{TP + FP + FN} = \frac{TP}{N} \tag{1}$$

Precision, as the proportion of cases in which the system reports an identification and this claim is correct, i.e. a system that operates with high precision produces a low proportion of false positives:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

To assess system performance for cases $(C2)$, in which the reference track is unknown, i.e. not present in the database, we compute a third evaluation measure, the specificity:

$$Specificity = \frac{TN}{TN + FP} \tag{3}$$

Here, TN denotes the number of seconds in which no result was produced, and at the same time the reference track is absent. The number of FP are the cases where the system reports a match despite the fact that there is no reference. Specificity expresses the capability of a system to avoid false predictions by not reporting a result.

The identification performance of all three methods is listed in Table 2. We will discuss the results in the Section below, and analyze the properties and differences of the methods.

6. DISCUSSION OF RESULTS

Table 2 summarizes the results of each method on the disco set and the mixotic set (rows Q^{v} and Q^{ϵ} become relevant at a later point of this section). For the disco set, the accuracy shows that just between 25% and 74% of detectable seconds were assigned to the correct reference track. This reveals that DJ mix track identification indeed is a tough problem. The precision values show that Audfprint and Panako claim a wrong track in around 50% of the cases where the correct track should be identifiable. The specificity of the systems shows that Audfprint correctly abstains from claiming a match in roughly 50% of the cases where no track can be found because it is not referenced in the database. Panako shows higher specificity at around 75%. Qfp manages to correctly treat TN in 94% of the cases.

The results obtained from the experiment on the mixotic set show better accuracy for all three methods, and Audfprint and Qfp operate with higher precision than on the disco set. For the mixotic set, all three systems show lower

specificity than for the disco set. We believe that this is a result of the larger reference database (723 songs rather than 296 in the disco set) and the highly repetitive tracks in the mixotic set. In total, Qfp performs at higher accuracy, precision and specificity than Audfprint and Panako. Panako shows higher specificity than Audfprint on both datasets.

The low specificity of the algorithm that is implemented in Audfprint indicates that its fingerprints are too general. Panako uses triples of peaks, which inherently capture more specific information of the local signal. Indeed, its specificity on the disco set is considerably higher than that of Audfprint, i.e. its fingerprint descriptors are less general, which may be the reason for it to correctly refuse to make a claim in around 75% of the cases on the disco set, and in roughly 35% of the cases on the mixotic set.

Analysis Qfp performs best on the tested datasets. To find out which properties of the system are responsible for that, we perform two additional experiments. The first experiment is intended to investigate the impact of the verification process, and the second experiment highlights the effect of the range query for Qfp. For a detailed explanation on the parameters that are mentioned in this section, we ask the reader to consult [17].

First, we want to find out if it is the verification process that allows it to maintain high performance.

If we switch off the verification [6] and run the experiments, this results in an overall accuracy of 0.76, a precision of 0.90, and a specificity of 0.89 on the disco set. For the mixotic set this results in the accuracy of 0.89, precision of 0.95 and a specificity of 0.65 (c.f. Table 2, row Q^v). In terms of accuracy and precision, the results for both datasets are comparable to those with active verification. The specificity on the mixotic set, however, is notably lower.

We now investigate the performance of the Qfp method using a reduced neighbourhood for the range queries. We argue, that this loosely translates to using quantized hashes, i.e. if a query peak moves with respect to the others, the corresponding reference hash cannot be retrieved. This neighbourhood is specified as distance in the continuous hash space of the quad descriptor. For this experiment we reduce this distance from $0.0035, 0.012$ for pitch and time to $0.001, 0.001$ for pitch and time. For the disco set, this results in a low accuracy of 0.18, precision of 0.96 and specificity of 0.99. On the mixotic set, the small range query neighbourhoods result in an accuracy and precision of 0.57 and 0.98, and specificity of 0.99 (c.f. Table 2, Q^ϵ).

Extended Database We now add the reference tracks of both, the disco set and the mixotic set to a reference database that consists of $430\,000$ full length tracks (this captures almost the entire Jamendo corpus [7]), and inspect how the Qfp method responds to that amount of additional tracks. The overall result for the disco set (with standard settings for the range query and verification) is 0.69 for accuracy and 0.80 for precision. The specificity is 0.71. On the mixotic set, the results are as follows: Accuracy 0.83, precision 0.87 and specificity 0.56. The low specificity here is also impacted by a song duplicate in the DJ mixes and Jamendo corpus, i.e. in the case of mixotic set 282, Qfp could correctly identify the track "Akusmatic - Scamos" within the additional $430\,000$ songs, but the evaluation treats this as FP, because according to the ground truth this track is not present. The issue with song duplicates does not influence any other experiments in this work, since we use the extended reference database only with the Qfp method.

The experiment shows that there is a certain negative impact, causing more FP when trying to identify tracks in DJ mixes on larger databases. Note that these results also depend on the experiment setup as defined in Section 5, where we chose to assign the identified track ID to the whole query of 20 seconds in length. If we respect the reported start and end time of identified queries, the results on the disco set give an accuracy of 0.60, precision of 0.88, and a specificity of 0.89. For the mixotic set the accuracy then is 0.76, precision is 0.93 and the specificity results in 0.80.

Qfp turns out to maintain – what we think is – acceptable performance, on a database with $430\,000$ full length songs. According to precision and specificity, the other methods tested in this work seem to get distracted by 723 reference songs. This leads us to suggest that the monitoring of DJ mixes via automated fingerprinting systems indeed is a challenging task.

Visual analysis The different behaviour of the systems can be conveyed visually. In Figure 1 we show an excerpt of the *mixotic set* mix-ID 222 [8], from second 1500 to 4300. Vertical lines represent song borders. The figure shows the scattered query identification results, where the x-axis position is the query time, and the y-axis position locates the query within the reference song that the system could identify. Thus, scattered positions of songs that are correctly identified over several successive queries usually take the shape of a sawtooth function. In DJ mixes this will not always be the case, as the DJ can loop content. The different track names are encoded as markers, to be able to see if a system tends to confuse the same two tracks, or whether it reports many different tracks for a portion that it fails to identify correctly. The larger markers shown on top, between song borders, are the reference. A missing reference marker means that the song is not present in the database. Note that the evaluation does not consider whether the predicted position within the reference is correct, as this is not meaningful for highly repetitive musical content.

[6] Strictly speaking, the implementation does not allow to switch off the verification. Therefore we instead relax the verification constraints such that no candidate can be rejected.

[7] Jamendo is accessible via https://www.jamendo.com.

[8] The mix-IDs are listed and explained in the published dataset.

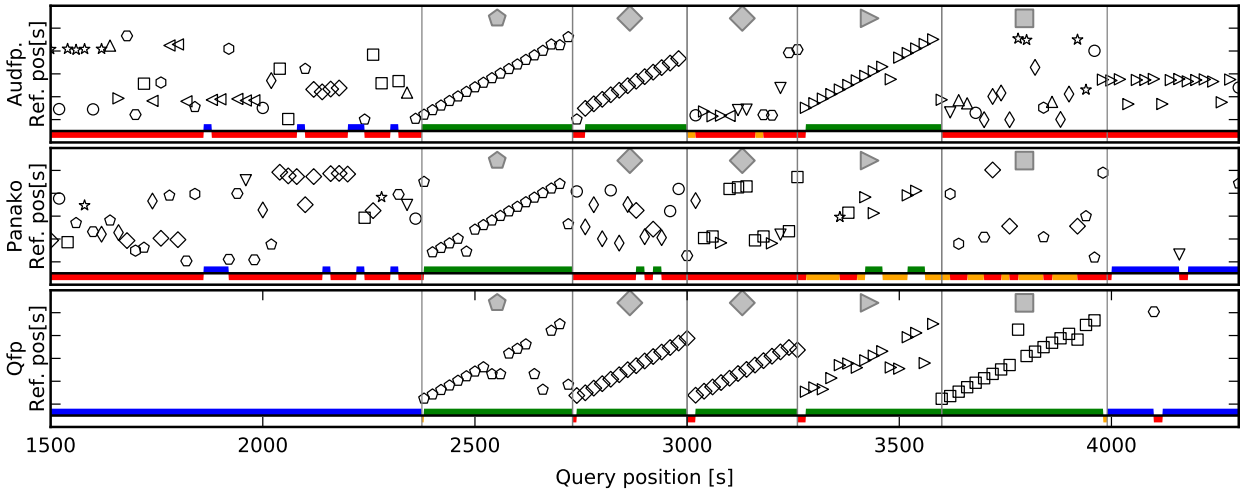


Figure 1: Query visualisation of an excerpt of mixotic set-ID 222. The rows show the results of individual, non-overlapping $20s$ queries without smoothing of predictions for Audfprint (top), Panako (middle) and Qfp (bottom). The vertical lines are the annotated song borders. The identification claims of the systems are encoded in the shown markers, where each marker represents a reference track. The x-axis position shows the query excerpt position, and y-axis the location of the matched query within the identified reference track. A missing large marker between song borders means that the reference song is not present in the database. The figures show a bar at the bottom, which represents the confusions. TP (green) and TN (blue) are shown on top of the horizontal line, FP (red) and FN (yellow) are shown below.

7. CONCLUSIONS

The results obtained from the experiments shown in this work support the intuition that automated audio identification on DJ mixes is a challenging problem. We observe that the Qfp method performs best on the tested datasets, and believe that it constitutes a well suited method to further investigate the analysis of DJ mixes via audio fingerprinting.

For future work and experiments we strive to collect DJ mixes with accurate annotations and timestamps, that are exported from the specific software or the midi controller used by the DJ. This would allow to gain insight on what kinds of effects and combinations thereof prevent automated identification systems from correctly identifying certain portions of query audio.

8. ACKNOWLEDGEMENTS

This work is supported by the Austrian Science Fund FWF under projects TRP307 and Z159, and by the European Research Council (ERC Grant Agreement 670035, project CON ESPRESSIONE).

9. REFERENCES

[1] Shumeet Baluja and Michele Covell. Audio fingerprinting: Combining computer vision & data stream processing. In *Acoustics, Speech and Signal Processing (ICASSP), 2007. IEEE International Conference on*, volume 2, pages II–213. IEEE, 2007.

[2] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma. A review of algorithms for audio fingerprinting. In *Multimedia Signal Processing, 2002 IEEE Workshop on*, pages 169–173. IEEE, 2002.

[3] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma. A review of audio fingerprinting. *Journal of VLSI signal processing systems for signal, image and video technology*, 41(3):271–284, 2005.

[4] Elsa Dupraz and Gaël Richard. Robust frequency-based audio fingerprinting. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 281–284. IEEE, 2010.

[5] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[6] Frank Kurth, Thorsten Gehrmann, and Meinard Müller. The cyclic beat spectrum: Tempo-related audio features for time-scale invariant audio identification. In *ISMIR*, pages 35–40, 2006.

[7] Frank Kurth, Andreas Ribbrock, and Michael Clausen. Identification of highly distorted audio material for querying large scale data bases. In *Audio Engineering Society Convention 112*. Audio Engineering Society, 2002.

[8] Dustin Lang, David W Hogg, Keir Mierle, Michael Blanton, and Sam Roweis. Astrometry.net: Blind astrometric calibration of arbitrary astronomical images. *The Astronomical Journal*, 137:1782–2800, 2010. arXiv:0910.2233.

[9] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[10] Chun-Shim Lu. Audio fingerprinting based on analyzing time-frequency localization of signals. In *Multimedia Signal Processing, 2002 IEEE Workshop on*, pages 174–177. IEEE, 2002.

[11] Mani Malekesmaeili and Rabab K Ward. A local fingerprinting approach for audio copy detection. *Signal Processing*, 98:308–321, 2014.

[12] Meinard Müller, Frank Kurth, and Michael Clausen. Audio matching via chroma-based statistical features. In *ISMIR*, volume 2005, page 6th, 2005.

[13] Mathieu Ramona and Geoffroy Peeters. Audioprint: An efficient audio fingerprint system based on a novel cost-less synchronization scheme. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 818–822. IEEE, 2013.

[14] Joren Six and Olmo Cornelis. A robust audio fingerprinter based on pitch class histograms applications for ethnic music archives. In *Proceedings of the Folk Music Analysis conference (FMA 2012)*, 2012.

[15] Joren Six and Marc Leman. Panako - a scalable acoustic fingerprinting system handling time-scale and pitch modification. In *ISMIR*, pages 259–264, 2014.

[16] Reinhard Sonnleitner and Gerhard Widmer. Quad-based audio fingerprinting robust to time and frequency scaling. In *Proceedings of the 17th International Conference on Digital Audio Effects, DAFx-14, Erlangen, Germany, September 1-5, 2014*, pages 173–180, 2014.

[17] Reinhard Sonnleitner and Gerhard Widmer. Robust quad-based audio fingerprinting. *IEEE/ACM Trans. Audio, Speech & Language Processing*, 24(3):409–421, 2016.

[18] Avery L Wang. An industrial-strength audio search algorithm. In *ISMIR*, pages 7–13, 2003.

[19] Xiu Zhang, Bilei Zhu, Linwei Li, Wei Li, Xiaoqiang Li, Wei Wang, Peizhong Lu, and Wenqiang Zhang. Sift-based local spectrogram image descriptor: a novel feature for robust music identification. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015(1):1–15, 2015.

[20] Bilei Zhu, Wei Li, Zhurong Wang, and Xiangyang Xue. A novel audio fingerprinting method robust to time scale modification and pitch shifting. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 987–990. ACM, 2010.

LEARNING AND VISUALIZING MUSIC SPECIFICATIONS USING PATTERN GRAPHS

Rafael Valle[1] **Daniel J. Fremont**[2] **Ilge Akkaya**[2]
Alexandre Donze[2] **Adrian Freed**[1] **Sanjit S. Seshia**[2]
[1] UC Berkeley, CNMAT
[2] UC Berkeley
rafaelvalle@berkeley.edu

ABSTRACT

We describe a system to learn and visualize specifications from song(s) in symbolic and audio formats. The core of our approach is based on a software engineering procedure called specification mining. Our procedure extracts patterns from feature vectors and uses them to build pattern graphs. The feature vectors are created by segmenting song(s) and extracting time and and frequency domain features from them, such as chromagrams, chord degree and interval classification. The pattern graphs built on these feature vectors provide the likelihood of a pattern between nodes, as well as start and ending nodes. The pattern graphs learned from a song(s) describe formal specifications that can be used for human interpretable quantitatively and qualitatively song comparison or to perform supervisory control in machine improvisation. We offer results in song summarization, song and style validation and machine improvisation with formal specifications.

1. INTRODUCTION AND RELATED WORK

In software engineering literature, specification mining is an efficient procedure to automatically infer, from empirical data, general rules that describe the interactions of a program with an application programming interface (API) or abstract datatype (ADT) [3]. It has convenient properties that facilitate and optimize the process of developing formal specifications. Specification mining is a procedure that is either entirely automatic, or only requires the relatively simple task of creating templates. It offers valuable information on commonalities in large datasets and exploits latent properties that are unknown to the user but reflected in the data. Techniques to automatically generate specifications date back to the early seventies, including [5, 24]. More recent research on specification mining includes [2,3,10,17]. In general, specification mining tools mine temporal properties in the form of mathematical logic or automata. Figure 1 describes a simple musical specification. Broadly speaking, the two main strategies for building these automata include: 1) learning a single automaton and inferring specifications from it; 2) learning small templates and designing a complex automaton from them. For example, [3] learns a single probabilistic finite state automaton from a trace and then extracts likely properties from it. The other strategy circumvents the NP-hard challenge of directly learning a single automaton [14, 15] by first learning small specifications and then post-processing them to build more complex state machines. The idea of mining simple alternating patterns was introduced by [10], and subsequent efforts include [12, 13, 25, 26].

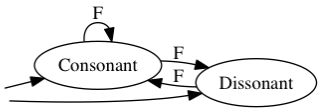


Figure 1: This graph describes three specifications: 1) a sequence must start (unlabelled incoming arrow) with a note of any type; 2) every note that does not belong to the underlying chord (dissonant) must be followed by a note that belongs to that chord (consonant); 3) a consonant note must be followed by a dissonant note or another consonant note; F means followed.

Manually describing such general rules from music is a complex problem, even for experts, due to music's parameter space complexity and richness of interpretation. Specification mining is a very attractive solution because it offers a systematic and automatic mechanism for learning these specifications from large amounts of data. Similar to specification mining strategies, algorithms for pattern discovery in music such as [6, 20, 21] combine segmentation and exhaustive search to find patterns that will be condensed to create a statisticailly significant description of the song(s). Our method avoids the exhaustive search by searching for specific patterns and creates a complex pattern graph by combining the patterns found, combining pattern graphs, and recursively building pattern graphs learned from pattern graphs. The pattern graph allows the representation of edges and nodes as mathematical objects, e.g. multidimensional point sents or Gaussian Mixture Models (GMM), hence it is not limited to strings.

© Rafael Valle, Daniel J. Fremont, Ilge Akkaya, Alexandre Donze, Adrian Freed, Sanjit S. Seshia. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Rafael Valle, Daniel J. Fremont, Ilge Akkaya, Alexandre Donze, Adrian Freed, Sanjit S. Seshia. "Learning And Visualizing Music Specifications Using Pattern Graphs", 17th International Society for Music Information Retrieval Conference, 2016.

2. SPECIFICATIONS AND PATTERN GRAPH

This paper adapts the work of [17] to formally describe specification mining in music. It expands our previous efforts in [9] by developing an inference engine that uses pre-defined templates to mine from a collection of traces (songs) specifications in the form of pattern graphs.

2.1 Formal Definition

Let F be a list of features extracted from a song $\mathcal{S}$, e.g. pitch, duration, chroma, etc. The notation $v_{f,t}$ indicates the value of $f \in F$ at time t.

Definition 1 (Event) *Formally, we define an event with the tuple $(\vec{f}, \vec{v}, t)$, where $\vec{f}$ is a set of features and $\vec{v}$ is their corresponding values at time t. The alphabet Σ_f is the set of distinct events given feature f, and a finite trace τ is a sequence of events ordered by their time of occurrence.*

Definition 2 (Projection) *The projection π of a trace τ onto an alphabet Σ, $\pi_\Sigma(\tau)$, is defined as τ with all events not in Σ deleted.*

Definition 3 (Specification Pattern) *A specification pattern is a finite state automata, FSA, over symbols Σ. Patterns can be parametrized by the events used in this alphabet; for example, we use "the $\mathcal{A}$ pattern between events a and b" to indicate the pattern obtained by taking a FSA $\mathcal{A}$ with $|\Sigma| = 2$ and using a as the first element of Σ and b as the second. A pattern is satisfied over a trace τ with alphabet $\Sigma_\tau \supseteq \Sigma$ iff $\pi_\Sigma(\tau) \in \mathcal{L}(\mathcal{A})$, that is, if and only if the projection of the trace onto the alphabet Σ is in the language of A.*

Definition 4 (Binary Pattern) *A binary pattern is a specification pattern with alphabet size 2. We denote a binary pattern between events a and b as a **R** b, where **R** is a label identifying the pattern.* [1]

Definition 5 (Pattern Graph) *A pattern graph is a labelled directed multigraph whose nodes are elements of Σ_f , i.e. values of a feature f . A node can be labelled as a starting node, an ending node, or neither. Edges are labelled with a type of binary pattern and a count indicating how many times the pattern occurred in the dataset used to build the pattern graph.*

For example, an edge (a, b) labelled (R, 3) in the pattern graph means the pattern a **R** b occurred 3 times in the dataset. Figure 3 provides a complete example of a pattern graph learned from the example in Figure 2. We have indicated starting nodes with an unlabelled incoming arrow and ending nodes with a double circle (by analogy to the standard notation for FSAs). [2]

[1] Although we explored binary patterns in this paper, our method supports patterns with more than two events.

[2] A pattern graph can be converted into an automaton, but is not itself an automaton.

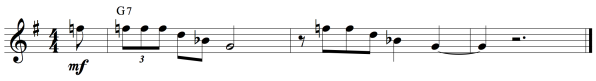


Figure 2: First phrase of Crossroads Blues by Robert Johnson as transcribed in the Real Book of Blues. The transition from chord degree 10 (note f) to chord degree 7 (note d) is always preceded by two or several occurrences of chord degree 10. Not merging *10* **F** *7* with *10* **F** *7* represents a musical inconsistency and the pattern graph would accept words such as $(10, 7, 10, 7)$.

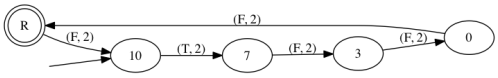


Figure 3: Pattern graph learned on the chord degree feature (interval from root) extracted from the phrase in Fig. 2. The **F** pattern between chord degrees 10 and 7 has been merged into the pattern 10 **T** 7.

2.2 Patterns

We generate specifications by mining small patterns from a set of traces and combining the mined patterns into a pattern graph. The patterns in this paper as described as regular expressions, *re*, and were chosen based on idiomatic music patterns such as repetition and ornamentation. Other patterns can be mined by simply writing their *re*.

Followed(F): This pattern occurs when event a is immediately followed by event b. It provides information about immediate transitions between events, e. g. resolution of non-chord tones. We denote the followed pattern as *a* **F** *b* and describe it with the *re* (ab).

'Til(T): This pattern occurs when event a appears two or more times in sequence and is immediately followed by event b. It provides information about what transitions are possible after self-transitions are taken. We denote the 'til pattern as *a* **T** *b* and describe it with the *re* (aaa^*b).

Surrounding(S): This pattern occurs when event a immediately precedes and succeeds event b. It provides information over a time-window of three events and we musically describe it as an ornamented self-transition. We denote the surrounding pattern as *a* **S** *b* and describe it with the *re* (aba).

2.3 Pattern Merging

If every match to a pattern $P_2 = a\ \mathbf{R}\ b$ occurs inside a match to a pattern $P_1 = a\ \mathbf{Q}\ b$, we say that P_1 *subsumes* P_2 and write $P_1 \implies P_2$. When this happens, we only add the *stronger* pattern P_1 to the pattern graph, with the purpose of emphasizing longer musical structures. Given the patterns described in this paper:

1. *a* **T** *b* $\implies$ *a* **F** *a*, *a* **F** *a* is merged into *a* **T** *b*
2. *a* **T** *b* $\implies$ *a* **F** *b*, *a* **F** *b* is merged into *a* **T** *b*
3. *a* **S** *b* $\implies$ *a* **F** *b*, *a* **F** *b* is merged into *a* **S** *b*
4. *a* **S** *b* $\implies$ *b* **F** *a*, *b* **F** *a* is merged into *a* **S** *b*

Shorter patterns not included will be added *iff* they occur outside the scope of longer patterns. Nonetheless, the pattern graph is designed such that it accepts traces that satisfy the longer pattern, e.g. a **T** b accepts the sequences aab and $aaab$, but not ab or aac.

3. LEARNING AND ENFORCING SPECIFICATIONS

3.1 Learning Specifications

Given a song dataset and their respective features, we build pattern graphs $\mathcal{G}_f \in \mathcal{G}$ by mining the patterns described in 2. The patterns in $\mathcal{G}$ correspond to the set of allowed patterns, while all others are forbidden.

The synchronous product of $\mathcal{G}_f$ can be used to build a specification graph $\mathcal{G}^s$ that can be used to supervise the output of a machine improviser. This concept originates from the Control Improvisation framework, which we first introduced in [8, 9] and have used in IoT applications [1]. We refer the reader to [11] for a thorough explanation.

Algorithm 1 describes the specification mining algorithm. $\mathfrak{D}$ is a dataset, e.g. Table 1, containing time and frequency domain features, described in section 4, extracted from songs with or without phrase boundary annotations; $\mathcal{P}$ is a list containing string representations of the regular expressions that are used to mine patterns. The pattern graph implementation and the code used to generated this paper can be found on our github repository [3]

Algorithm 1: Specification Mining Algorithm

Input: dataset $\mathfrak{D}$ over features $\mathcal{F}$; patterns $\mathcal{P}$
Output: a pattern graph $\mathcal{G}_f$ for each $f \in \mathcal{F}$

```
for f ∈ F do
    G_f ← new pattern graph on vertices Σ_f
    for song ∈ D do
        for phrase ∈ song do
            phrase_f ← the sequence of values of the
            feature f in phrase
            label the first element of phrase_f as a
            starting node in G_f
            label the last element of phrase_f as an
            ending node in G_f
            for a, b ∈ Σ_f do
                counts ←
                countPatternMatches(a, b, phrase_f, P)
                foreach pattern P with
                counts(P) > 0 do
                    add to G_f the edge (a, b) with
                    label (P, counts(P))
```

In the next section we describe some of the features, or viewpoints, that we used in this paper to build specifications that describe relevant musical properties of a song(s).

[3] https://github.com/rafaelvalle/music_pattern_graphs

4. MUSIC SPECIFICATION MINING

We abstract and formalize a *song* into a sequence of feature values possibly aligned with a *chord progression*, phrase-segmented and including key signature changes. In this paper, the time unit is the *beat*, including respective integer subdivisions. To encode all events in a score, we use an alphabet which is the product of five alphabets: $\Sigma = \Sigma_p \times \Sigma_d \times \Sigma_a \times \Sigma_b \times \Sigma_{12}$, where

- Σ_p is the *pitches* alphabet, i.e. $\Sigma_p = \{\text{𝄽}, \texttt{a0}, \texttt{a\#0}, \cdots\}$;
- Σ_d is the *durations* alphabet, i.e. $\Sigma_d = \{\text{𝅘𝅥𝅯}, \text{♩}, \text{𝅗𝅥}, \ldots\}$ with $\text{♩} = 1$ beat. Note that Σ_d also includes positive integer subdivisions of the beat, e.g. for tuplets.
- Σ_c is the *chords* alphabet, i.e. $\Sigma_c = \{\text{C, D7\#4}, \ldots\}$;
- Σ_b is the *beat* alphabet. For example, if the smallest duration (excluding fractional durations) is the eighth and the meter is in 4, then Σ_b is $\{0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5\}$.
- Σ_{12} is the binary *chroma* alphabet. For this, we interpret the binary chroma as a binary number and encode it with the respective Unicode string.

Note that the full alphabet enables the creation of data abstractions, e.g. chord degree. Below we describe the data abstractions implemented using the alphabet above. A similar strategy is used in [6, 7], where data abstractions (derived types, viewpoints) are implemented. In our current implementation, all the specifications implicitly use the full alphabet Σ via the product of pattern graphs.

4.1 Time Domain Features

- **Event Duration**: This feature describes the duration in beats of silences and notes. It imposes hard constraints on duration diversity but provides weak guarantees on rhythmic complexity because it has no awareness of beat location. Figure 4 provides one example of such weak guarantees. Further constraints can be imposed by combining event duration and beat onset location.

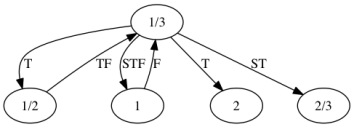


Figure 4: Selection of event duration specifications learned from a blues songs dataset. The pattern 1/3 **S** 1 (1/3, 1, 1/3) is allowed but can produce incomplete tuplets.

- **Beat onset location**: This feature describes where events happen within the beat. Cooperatively, event duration and beat onset location produce complex specifications that allow for rhythmic diversity. These specifications extend the work in [9] by replacing handmade specifications designed to ensure rhythmic tuplet completeness with specifications learned from data. Figure 5 provides an example of such specifications learned from 4/4 songs.

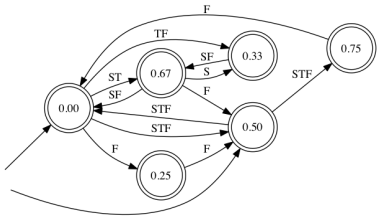


Figure 5: Beat onset location specifications learned from a blues songs dataset.

4.2 Frequency Domain Features

- **Scale Degree**: The scale degree is the identification of a note disregarding its octave but regarding its distance from a reference tonality. Songs usually impose soft constraints on the pitch space, defining the set of appropriate scale degrees and transitions thereof. Figure 6 provides a selection of scale degree mined specifications. Since scale degree can only provide overall harmonic constraints to each tone over the scope of the entire song, we use another feature to provide harmonic constraints based on chord progression, therefore increasing the temporal granularity of the harmonic specifications.

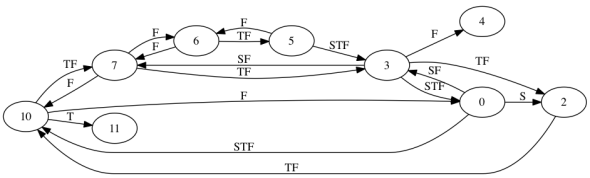


Figure 6: Selection of scale degree specifications learned from a blues songs dataset. These specifications conform with the general consent that blues songs include the main key's major scale with the "flat seven" (scale degree 10) and the blue note (scale degree 3). Note that sharp fourths (scale degree 6) are used as approach tones to scale degrees 5 and 6.

- **Interval Classification**: Expanding on [9], we replace the hand-designed tone classification specifications, here called interval classification, with mined specifications. These specifications provide information about the size (diatonic or leap) and quality (consonant or dissonant) of the music interval that precedes each tone. Figure 7 illustrates mined specifications. Although scale degree and interval classification specifications ensure desirable harmonic guarantees given key and chord progression, they provide no melodic contour guarantees.

- **Melodic Interval**: This feature operates on the first difference of pitch values and is associated with the contour of a melody. Combined with scale degree and interval classification, it provides harmonic and melodic constraints, including melodic contour.

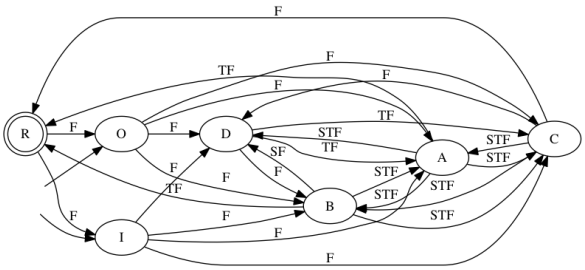


Figure 7: Interval class specifications learned from a blues songs dataset. The symbols A, B, C, and D, describe tones reached by consonant step, consonant leap, dissonant (non-chord tones) step, and dissonant leap respectively. Consonant and dissonant notes preceded by rests, R, are described with the symbols I and O respectively.

- **Chord Degree**: The chord degree is the identification of a note regarding its distance in semitones to the root of a chord. It adds harmonic specificity to the interval class.

Table 1 provides the reader with a selection of features extracted from a blues song with chord and phrase number annotations. The next section analyzes in detail the application of pattern graphs and specifications in song summarization, song and style validation, and machine improvisation with formal specifications.

5. EXPERIMENTAL RESULTS

For the experiments in this paper, we learned pattern graphs and pattern sequences from three non-overlapping datasets, namely:

$\mathfrak{D}^{train}$ a dataset of 20 blues songs with chord and phrase annotations, transcoded from the Real Book of Blues [18];

$\mathfrak{D}^{test}$ a dataset of 10 blues songs with chord and phrase annotations, transcoded from the Country Blues songbook [16];

SAC a dataset, with 10 genres and 25 pieces of music per genre [19].

pretty_midi [22] is used for handling midi data.

5.1 Style and Song Summarization

Pattern graph plots can be used to understand and visualize the patterns of a song or musical style. In section 4 we provided pattern graph visualizations that described significant musical properties of $\mathfrak{D}^{train}$. Pattern sequence plots, on the other hand, offer a visualization that is directly related to a song's formal structure. A pattern sequence plot is a color sequence visualization of a pattern sequence extracted from a song; for example, the chroma pattern sequence (100010010000, T, 000000000000, F, 100000000000), describes: play any inversion of the C

	chord	dur	measure	phrase	...	pitch	mel_interval	beat	interval_class
0	F7	14/3	1	1	...	69	R	1	I
1	F7	1/3	2	1	...	65	-4	5/3	B
2	F7	2/3	2	1	...	67	2	1	C
...	...	...	...	...	...	...	...	.....	
23	B-7	1	10	3	...	67	-1	1	C
24	F7	4	11	3	...	65	-2	1	A
25	F7	-4	12	3	...	R	R	1	R

Table 1: Dataframe from Blues Stay Away From Me by Wayne Raney et al. R represents a rest.

major triad two or more times, followed by one rest followed by the note C played one time. The conversion of a feature into color is achieved by mapping each feature dimension to RGB. Features with more than three dimensions undergo dimensionality reduction to a 3 dimensional space through non-negative matrix factorization (NMF) [4] . Figure 8 shows a plot of binary chroma and dimensionality reduced binary chroma overlaid with the patterns associated with each time step.

5.2 Song and Style Validation

Song and style validation describe to what extent a song or a style violates a specification. A violation occurs when a pattern does not satisfy a specification, i.e. the pattern does not exist in the pattern graph. Figure 9 provides histograms of violations obtained by validating $\mathfrak{D}^{test}$ on chord degree and interval specifications learned from $\mathfrak{D}^{train}$.

Given a total of 355 patterns learned from $\mathfrak{D}^{test}$, there were 35 chord degree violations and 35 melodic interval violations, producing an average violation ratio of ≈ 0.02 per song [5] . The dataset used for learning the specifications is small. A larger dataset will enable us to better investigate how they are not characteristic of the blues.

For the task of style validation, we build binary chroma specifications for each genre in the SAC dataset. The specifications are used separately to validate all genres in the SAC dataset. Validation is performed with the average validation ratio, which is computed as the ratio of violations given number of patterns in the song being validated. Figure 10 provides the respective violation ratio matrix. [6] These validations can be exploited in style recognition and we foresee that more complex validations are possible by using probabilistic metrics and describing pattern graph nodes as GMMs.

5.3 Machine Improvisation with formal specifications

Machine improvisation with formal specifications is based on the framework of Control Improvisation. Musically speaking, it describes a framework in which a *controller* regulates the events generated by an *improviser* such that all events generated by the *improviser* satisfy hard (non-probabilistic) and soft (probabilistic) specifications.

Using a 12-bar blues excerpt and its chord progression shown in Figure 12, we *navigated* the factor oracle [4] with 75% replication probability to generate improvisations with specifications generated from $\mathfrak{D}^{train}$. In this task we used duration, beat onset location, chord degree, interval class and melodic interval joint specifications.

We computed the average melodic similarity between $\mathfrak{D}^{train}$ and other sets of improvisation, including: 50 factor oracle improvisations generated without specifications, 50 factor oracle improvisations generated with specifications. The melodic similarity is computed using the algorithm described in [23]. As baselines, we also computed the similarity of $\mathfrak{D}^{train}$ to the 12 Bar Blues reference word and to 50 random improvisations. The results in Figure 11 show that the specifications are successful in controlling the events generated by the improviser, factor oracle, such that they are more similar to $\mathfrak{D}^{train}$ and satisfy the specifications learned from it.

Qualitatively, the improvisation without specifications violates several specifications related to expected harmonic and melodic behavior, as Figure 12 confirms. For example, measure 4 in the improvisation without specifications has chord degrees that violate harmonic specifications. This is possible because the events generated by the unsupervised improvisation disregard harmonic context, thus commonly producing unprepared and uncommon dissonant notes.

The improvisations with specifications are able to keep overall harmonic coherence despite the use of chromaticism. Their melodic contour is rather smooth and the improvisations include several occurrences of the 'Til and Surrounding patterns, as measures 5 and 1 respectively show.

6. CONCLUSIONS AND FUTURE WORK

This paper investigated the use of pattern graphs and specification mining for song and style summarization, validation, and machine improvisation with formal specifications. Our experimental results show that pattern graphs can be successfully used to graphically and algorithmically describe and compare characteristics of a music collection, and in guiding improvisations.

We are currently investigating smoothing strategies, including the use of a larger dataset, for pattern graph learning so that we can more robustly use probabilistic metrics for song and style validation.

[4] We use scikit-learn's NMF with default parameters

[5] $(35 + 35)/(355 * 10)$

[6] Note that this is not a confusion matrix and must not be symmetric.

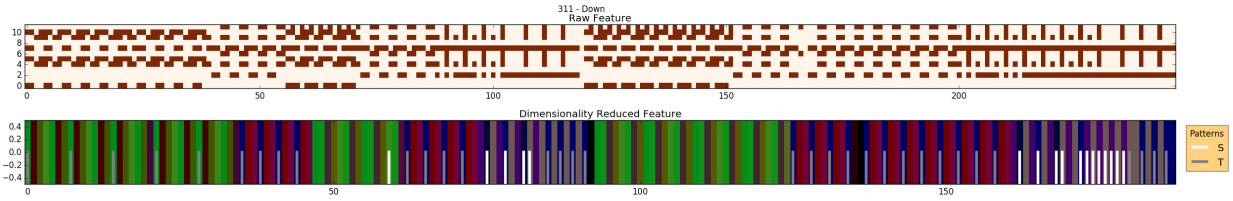


Figure 8: Down by the band 311 as found in SAC's dataset. The top plot shows the raw feature (binary chroma). The bottom plot shows the dimensionality reduced chroma (NMF with 3 components) with the components scaled and mapped to RGB. The patterns associated with each event are plotted as grayscale bars. The absence of a grayscale bar represents the Followed pattern.

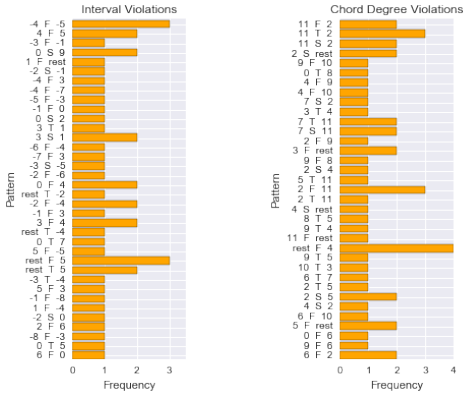


Figure 9: Histogram of melodic interval and chord degree violations. The y-axis represents the patterns that do not exist in the specification and the x-axis represents their frequency. F and T represent the patterns Followed and 'Till respectively.

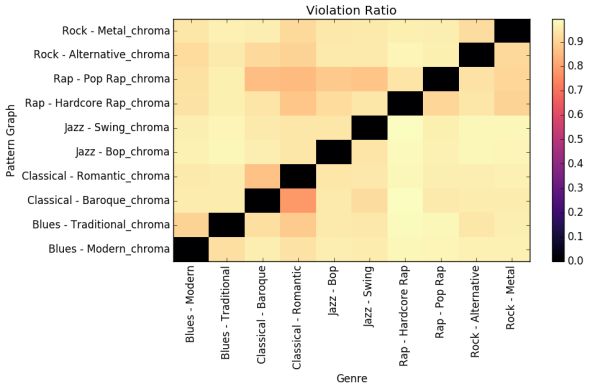


Figure 10: This violation ratio matrix shows that similar styles have lower violations ratios. Unexpectedly, Rap - Pop Rap specifications are more violated by Rock - Alternative than by Classical - Baroque or Classical - Romantic.

Although for this paper we hard-coded the pattern mining algorithm to avoid regular expression's long run time, we are researching sequential pattern mining algorithms that are fast and easy and flexible to use as *re*.

Last and most important, we are expanding specification mining to real-valued multidimensional features by expressing pattern graphs nodes as gaussian mixtures.

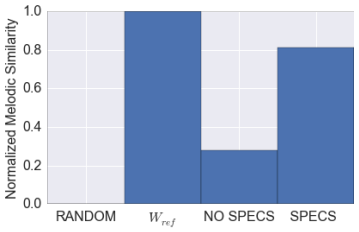
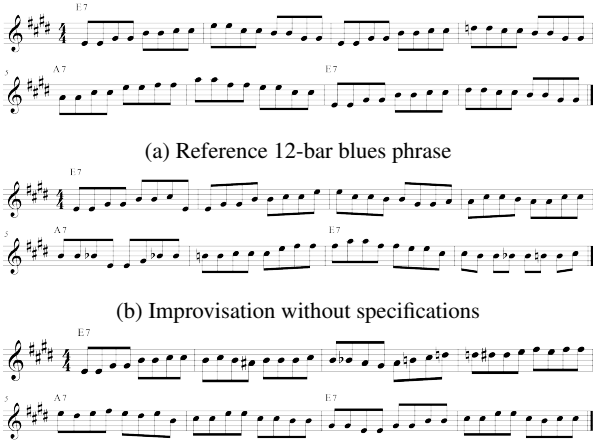


Figure 11: Normalized Melodic Similarity w.r.t $\mathfrak{D}^{train}$. W_{ref} is the 12-bar blues phrase used as improvisation input. NO SPECS and SPECS are improvisations generated with the factor oracle with 0.75 replication probability with and without specifications. The results show that specifications induce improvisations from that factor oracle that are closer to $\mathfrak{D}^{train}$.



(a) Reference 12-bar blues phrase

(b) Improvisation without specifications

(c) Improvisation with specifications

Figure 12: Factor Oracle improvisations with 0.75 replication probability on a traditional instrumental blues phrase.

7. ACKNOWLEDGEMENTS

This research was supported in part by the TerraSwarm Research Center, one of six centers supported by the STAR net phase of the Focus Center Research Program (FCRP) a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

8. REFERENCES

[1] Ilge Akkaya, Daniel J. Fremont, Rafael Valle, Alexandre Donze, Edward A. Lee, and Sanjit A. Seshia. Control improvisation with probabilistic temporal specifications. In *IEEE International Conference on Internet-of-Things Design and Implementation (IoTDI'16)*, 2015.

[2] Rajeev Alur, Pavol Cernỳ, Parthasarathy Madhusudan, and Wonhong Nam. Synthesis of interface specifications for java classes. *ACM SIGPLAN Notices*, 40(1):98–109, 2005.

[3] Glenn Ammons, Rastislav Bodík, and James R Larus. Mining specifications. *ACM Sigplan Notices*, 37(1):4–16, 2002.

[4] Gérard Assayag and Shlomo Dubnov. Using factor oracles for machine improvisation. *Soft Comput.*, 8(9):604–610, 2004.

[5] Michel Caplain. Finding invariant assertions for proving programs. In *ACM SIGPLAN Notices*, volume 10, pages 165–171. ACM, ACM, 1975.

[6] Darrell Conklin and Mathieu Bergeron. Feature set patterns in music. *Computer Music Journal*, 32(1):60–70, 2008.

[7] Darrell Conklin and Ian H Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.

[8] Alexandre Donzé, Sophie Libkind, Sanjit A. Seshia, and David Wessel. Control improvisation with application to music. Technical Report UCB/EECS-2013-183, EECS Department, University of California, Berkeley, Nov 2013.

[9] Alexandre Donzé, Rafael Valle, Ilge Akkaya, Sophie Libkind, Sanjit A. Seshia, and David Wessel. Machine improvisation with formal specifications. In *Proceedings of the 40th International Computer Music Conference (ICMC)*, 2014.

[10] Dawson Engler, David Yu Chen, Seth Hallem, Andy Chou, and Benjamin Chelf. *Bugs as deviant behavior: A general approach to inferring errors in systems code*, volume 35. ACM, 2001.

[11] Daniel J. Fremont, Alexandre Donzé, Sanjit A. Seshia, and David Wessel. Control improvisation. *CoRR*, abs/1411.0698, 2014.

[12] Mark Gabel and Zhendong Su. Javert: fully automatic mining of general temporal properties from dynamic traces. In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*, pages 339–349. ACM, 2008.

[13] Mark Gabel and Zhendong Su. Symbolic mining of temporal specifications. In *Proceedings of the 30th international conference on Software engineering*, pages 51–60. ACM, 2008.

[14] E. Mark Gold. Language identification in the limit. *Information and control*, 10(5):447–474, 1967.

[15] E. Mark Gold. Complexity of automaton identification from given data. *Information and control*, 37(3):302–320, 1978.

[16] Stefan Grossman, Stephen Calt, and Hal Grossman. *Country Blues Songbook*. Oak, 1973.

[17] Wenchao Li, Alessandro Forin, and Sanjit A. Seshia. Scalable specification mining for verification and diagnosis. In *Proceedings of the 47th design automation conference*, pages 755–760. ACM, 2010.

[18] Jack Long. The real book of blues. Wise, Hal Leonard, 1999.

[19] Cory McKay and Ichiro Fujinaga. Combining features extracted from audio, symbolic and cultural sources. In *ISMIR*, pages 597–602. Citeseer, 2008.

[20] David Meredith, Kjell Lemström, and Geraint A Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research*, 31(4):321–345, 2002.

[21] Marcus Pearce. *The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Composition*. PhD thesis, School of Informatics, City University, London, 2005.

[22] Colin Raffel and Daniel PW Ellis. Intuitive analysis, creation and manipulation of midi data with pretty_midi. In *15th International Society for Music Information Retrieval Conference Late Breaking and Demo Papers*, 2014.

[23] Rafael Valle and Adrian Freed. Symbolic music similarity using neuronal periodicity and dynamic programming. In *Mathematics and Computation in Music*, pages 199–204. Springer, 2015.

[24] Ben Wegbreit. The synthesis of loop predicates. *Communications of the ACM*, 17(2):102–113, 1974.

[25] Westley Weimer and George C Necula. Mining temporal specifications for error detection. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 461–476. Springer, 2005.

[26] Jinlin Yang, David Evans, Deepali Bhardwaj, Thirumalesh Bhat, and Manuvir Das. Perracotta: mining temporal api rules from imperfect traces. In *Proceedings of the 28th international conference on Software engineering*, pages 282–291. ACM, 2006.

LISTEN TO ME – DON'T LISTEN TO ME: WHAT CAN COMMUNITIES OF CRITICS TELL US ABOUT MUSIC

Ben Fields, Christophe Rhodes
Dept. of Computing
Goldsmiths University of London
New Cross
London SE14 6NW
United Kingdom
me@benfields.net, c.rhodes@gold.ac.uk

ABSTRACT

Social knowledge and data sharing on the Web takes many forms. So too do the ways people share ideas and opinions. In this paper we examine one such emerging form: the amateur critic. In particular, we examine genius.com, a website which allows its users to annotate and explain the meaning of segments of lyrics in music and other written works. We describe a novel dataset of approximately 700,000 users' activity on genius.com, their social connections, and song annotation activity. The dataset encompasses over 120,000 songs, with more than 3 million unique annotations. Using this dataset, we model overlap in interest or expertise through the proxy of co-annotation. This is the basis for a complex network model of the activity on genius.com, which is then used for community detection. We introduce a new measure of network community activity: community skew. Through this analysis we draw a comparison of between co-annotation and notions of genre and categorisation in music. We show a new view on the social constructs of genre in music.

1. INTRODUCTION

The near-ubiquitous availability and use of the Web has enabled many otherwise dispersed communities to coalesce. Many of these communities are concerned with the gathering and transfer of knowledge. Perhaps the best known of this kind of community is that of the editors and contributors at Wikipedia [1] [16, 22]. However, people coming together in a shared virtual space to exchange ideas is not limited to curation of encyclopedic facts. The Web is full of many communities; this paper focuses on an emerging one with a particular relevance to music: genius.com [2].

Genius.com brings users together through *annotation*. The stated purpose, and indeed, general use of the site is to explain portions of text through annotating them. These annotations can themselves be edited and modified, much as would take place on a website such as Wikipedia. Unlike on Wikipedia, however, the goal of allowing annotations is specifically to generate metadata: These annotations are both opinion and derivative works, criticism for the twitter age.

We have collected a significant sample of the user activity on Genius. This sample forms the core of a dataset that is ripe with potential. To show this, we construct a bipartite graph model of our Genius sample, connecting users and works via annotations made on those works. This graph model is then used to compare the communities formed around annotation with the genre prescribed to the annotated works. In doing this we seek to test the fitness and cultural relevance of the prescribed genre to these works.

The remainder of this paper is organized into the following sections. In Section 2 we discuss the relevant contexts: social network analytics in general, specific work in music, complex networks and community detection. From there, in Section 3 we describe the dataset – the collection techniques along with various statistics concerning the raw captured data. In Section 4 we then explore one possible avenue of use of our dataset, network modelling and community detection. We look at how detected communities align with prescribed genre labels for the works in these communities with a novel metric, *community skew*. Finally, we state our conclusions and consider what the next steps should be in Section 5.

2. BACKGROUND

When considering a social network of criticism such as Genius, we must consider what the landscape looks like to place this work in a more complete context.

[1] http://wikipedia.org

[2] The website and company began as rapgenius (http://rapgenius.com) with a strong focus on explaining the nuance, references, and in-jokes of rap and hip-hop lyrics. However they rebranded as 'Genius' as they widened their focus, which now includes lyrics from all genre of music as well as poetry, libretti, and factual texts such as news articles. See this announcement from 12 July 2014 http://genius.com/Genius-founders-introducing-geniuscom-annotated.

© Ben Fields, Christophe Rhodes. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Ben Fields, Christophe Rhodes. "Listen to Me – Don't Listen to Me: What Can Communities of Critics Tell Us About Music", 17th International Society for Music Information Retrieval Conference, 2016.

2.1 Music and Social Networks

While Genius has existed in some capacity since July of 2010 [3], it is one of many social networks with user-generated content (UGC) and an emphasis on music. One of the earliest of these networks was youtube [4]. While youtube is ostensibly a site for hosting and sharing video, it is also the single most prolific source of music on the Web [5]. Further, its social structure was one of the first on the modern Web to be extensively studied [4, 17]. It was shown that youtube, like many other Web-based social networks, has a power-law roll off in the distribution of its users' connections to other users and that the users congregate into clumps of tightly connected communities, showing 'small-world' characteristics.

Other Web-based communities brought together content creators with a greater explicit emphasis on social connections. In particular, myspace [6] has been looked at, both in terms of community structures [13] and as a proxy for understanding song and artist similarity [6, 7]. Further, these techniques have been used to drive recommenders and playlist generation [8]. In recent years, Soundcloud [7] has become the Web platform of choice for this combination of audio recordings and social network connectivity. It has broadly similar network characteristics [12, Chapter 3] with its own particular traits, reflecting interface and design decisions as well as the different user composition of the network. In addition to these networks around complete works, analysis has been done showing associations between properties of the contributor network for Freesound [8] (an open collection of audio clips) and creative outcomes among participants [19].

Analogous work has also been done on the listener or consumer side. In particular various aspects of listening and sharing behaviour on twitter [9] have been studied. The twitter microblogging platform has been successfully used to model artist similarity and descriptors, based on network ties and other attributes [20]. Extensions of this work then used twitter to show popularity trends across both time and space [21]. Going a step further, twitter network analysis can be used to create and order personalized playlists [14].

2.2 Information and Social Networks

While a significant volume of research has been done on information gathering social networks, it nearly exclusively uses Wikipedia as the source social network. As mentioned in Section 1, Wikipedia aims to be encyclopedic in both tone and scope, which colours the network significantly. Nevertheless, this work can offer useful insight and approaches for networks of this type.

Complex network techniques are effective in determining the most influential nodes across an information network [15]. This can be used to help understand how information flows through a social network. Wikipedia editors can broken down into different classes based on their behaviour within the network [11].

3. THE DATASET

In this section we describe the general structure of Genius, especially as it pertains to the dataset presented in this paper. We go into detail about the process of scraping and spidering the site to collect the data, highlighting sampling decisions and noting possible biases. Lastly, we present a statistical overview of the features of the dataset.

3.1 The Structure of Genius

At its core Genius is a collection of textual representations of works, most commonly but not exclusively lyrics. Each of these works are rendered such that an arbitrary sequence of words may be selected and a user may then write some commentary about the meaning of this section of the work (the *annotation*). An example of this display can be seen in Figure 1, in this case lyrics for *Hypnotize* by The Notorious B.I.G. with the line 'Timbs for hooligans in Brooklyn' highlighted with the annotation visible.

Once an annotation has been placed by a user, it can be edited and debated. This process can involve significant back and forth between users, as those interested within community voice their point of view as to the meaning of a line. The result is an annotation that reflects a collective process: the contributions that have led to the current state of an annotation are easily viewable, as can be seen in Figure 2 with the same annotation as the previous figure.

A user maintains a profile on Genius, as is the case on many social networks. Central to this profile is the history of the annotations made by the user. As such, a user's persona on Genius is effectively the collection of their annotations across the site. One such user profile is shown in Figure 3, that of the user 'OldJeezy', the lead contributor to the previously mentioned on annotation for the work *Hypnotize*.

3.2 Collecting the Data

Until recently Genius lacked any kind of machine-readable API [10], so our data collection effort restructured data drawn from the html as presented to a user. The data collection efforts on Genius are made up of two parts: a spider and a scraper. The spider, or mechanism to automatically move through the pages to be collected, sets out to evenly sample across the space of user IDs, without preference for or against how active a particular users is on the site. This algorithm is reasonably straight-forward and relies on the

[3] The beginning of their current site can be seen dating back to 22 July 2010 according to `http://web.archive.org/web/20100615000000*/http://rapgenius.com`

[4] `http://youtube.com`

[5] `http://www.nielsen.com/us/en/press-room/2012/music-discovery-still-dominated-by-radio--says-nielsen-music-360.html`

[6] `http://myspace.com`, though it has decayed a great deal from its peak of activity circa 2006-2008

[7] `http://soundcloud.com`

[8] urlhttp://www.freesound.org/

[9] `http://twitter.com`

[10] The recently announced API (`https://docs.genius.com/`) mitigates the most of the need for further scraping via html, though the spidering and sampling techniques detailed here are unchanged.

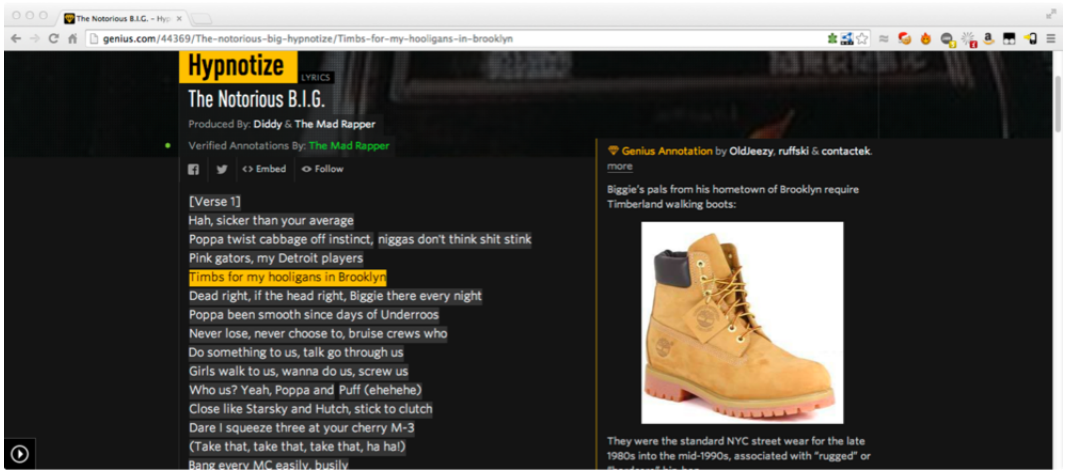


Figure 1. The lyrics to Hypnotize by The Notorious B.I.G., with an annotation shown for the line 'Timbs for hooligans in Brooklyn'. Taken from `http://genius.com/44369/The-notorious-big-hypnotize/Timbs-for-my-hooligans-in-brooklyn` on 10 March 2015.

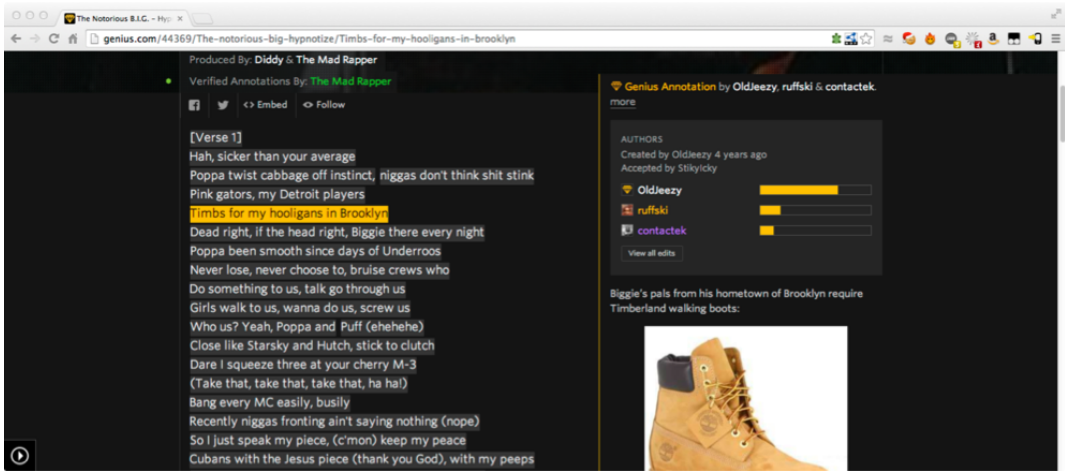


Figure 2. The same lyrics annotation as in Figure 1, but showing the total contribution of the three users who have edited the annotation for the highlighted text.

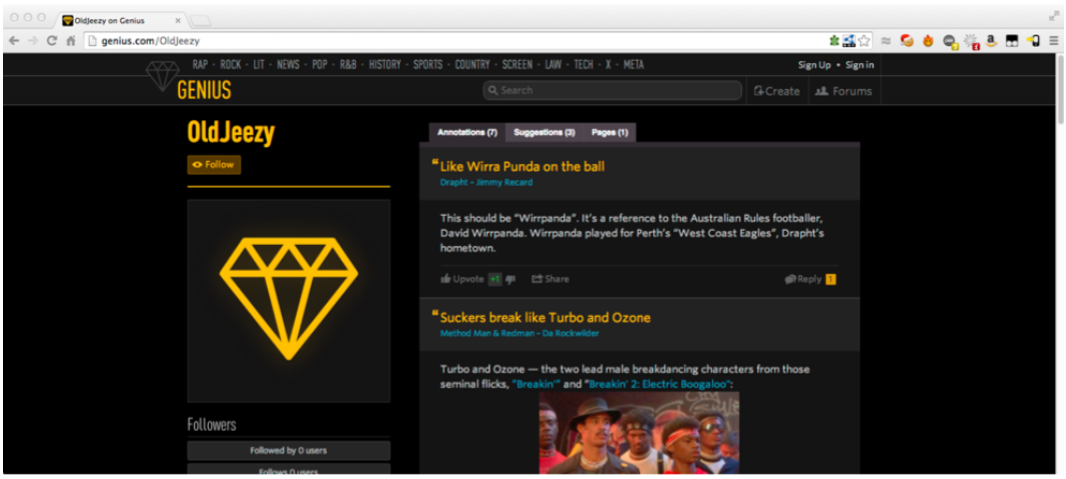


Figure 3. The recent annotation history for the user 'OldJeezy', the top contributor to the lyrics annotation shown in Figure 1. Taken from `http://genius.com/OldJeezy` on 10 March 2015

fact that Genius has sequential integer user IDs. As these ID are very nearly continuous from 0 to the most recent ID assigned, it is trivial to approximate a fair draw random generator to visit the user annotation history pages. Because of the flat and random mechanism in this spider, a partial sample is far less likely to introduce a bias toward a more densely connected graph than spidering methods that move from one user to another via a common edge (in this case a mutually annotated work). This implies that a partial capture will be reasonably representative of the whole userbase. The corresponding drawback is that any particular work may not have its entire annotation record collected, so its relative position in the topography of the graph (e.g. in terms of degree) may not be accurate, though this problem will decrease as more of the graph is captured.

To gather the data for each individual page, we created a screen scraper using Python and the BeautifulSoup [11] toolkit. This scraper is released with an open source license and is available from github [12] .

The spider and scraper were run during December 2014 collecting user metadata, annotation, works, and works metadata from the contributions of 704,438 users. This sample covers 41.1% of the 1,713,700 users [13] .

This dataset is available for download and reuse, as both CSV and SQL dump from the Transforming Musicology dataset repository [14] .

3.3 Statistical Overview

A variety of statistics describing the Genius data set can be seen in Table 1. As previously mentioned, the dataset covers the contributions of 704,438 users: 1,256,912 annotations on 146,186 unique works. Genius, as is common among many social networks [2], appears to have a steep drop off from users who sign up to users who do anything. This can be seen in the disparity between the total captured users and the contributing users (704,438 versus 71,129): 10.1% of users have written an annotation.

description	count
total users	704,438
total annotations	1,256,912
total works	146,186
contributing users	71,129
annotation edits	2,196,522
annotations with multiple contributors	194,795

Table 1. High-level statistics for Genius dataset.

Our dataset covers some 146,186 unique works and 1,256,912 annotations, giving a mean average of 8.6 annotations per work. Further, the dataset contains a total of 2,196,522 distinct edits of annotations, giving the mean annotation 1.75 edits, including its first.

Genius has 15 top-level categories for works on the site. Each work is assigned exactly one category, which can be taken as the work's genre. While that is not quite right for the non-musical categories, it is a helpful approximation. The breakdown of the works per category (genre) are seen in Table 2. The first thing that pops out is that while the company behind Genius may have decided to drop 'rap' from their name, it still dominates their collection of works, making up almost three-quarters of our dataset. While the meanings of most of these genre names are fairly typical, it is worth commenting on the few that are particular to Genius: 'x' is used as a catch-all or miscellaneous; 'screen' is for screenplays and teleplays; 'history' is for both scholarly and lay texts of a historical nature; 'unbranded' means our scraper was unable to capture the assigned genre; 'tech' covers prose about technology and the tech industry; finally 'meta' is where contributors to Genius discuss rules and community standards.

category	works count	percentage
rap	107270	73.3%
rock	16393	11.2%
lit	9386	6.2%
news	3720	2.5%
pop	3715	2.5%
sports	1140	0.7%
x	1014	0.6%
country	744	0.5%
screen	697	0.4%
r-b	655	0.4%
history	502	0.3%
unbranded	370	0.2%
law	250	0.1%
tech	159	0.1%
meta	151	0.1%

Table 2. Genre breakdown for Genius dataset.

In addition to the top-level categories, Genius supports work-level social tags. The tags have also been captured in our data set for all the works. As is typical for tags any number may be used per work, though the top-level genre category is repeated as a tag mechanically, so each work has at least one. Including these top-level categories, our dataset contains 802 unique tags. The top 10 tags (not including the categories), along with the count of the works they've been applied to, appears in Table 3.

4. NETWORK ANALYTICS

In an effort to understand what the community of annotators on Genius can tell us about that material they're annotating, we model our dataset as a graph. We use this graph, and a transform of it, to observe the community structure of works based on *co-annotation* and *user-overlap* patterns. Here co-annotation is when a common user annotates a pair of works. Similarly, user-overlap is when any pair of

[11] `http://www.crummy.com/software/BeautifulSoup/bs4/doc/`

[12] `http://dx.doi.org/10.5281/zenodo.17515`

[13] The total user count is an approximation based on the highest successfully resolving user ID as on 29 April 2015.

[14] Specifically `http://genius-annotations.data.t-mus.org/`, note that this dataset does not contain the source lyrics, only the network structure around the lyrics and their annontations. This is done for reasons of copyright compliance

category	works count
Rap Genius France	9135
Genius France	6009
Deutscher Rap	5725
Polski Rap	3298
West Coast	1384
Brasil	841
Bay Area	839
Indie Rock	716
Chicago	710
Genius Britian	540

Table 3. Top tags in Genius dataset.

users contribute to any annotation on the same work (not necessarily the same annotation).

4.1 The Graph Model

We initially model the dataset as a bipartite graph [9]. That is a graph where each node represents one of two distinct classes: a work or a user. The edges is this graph are formed whenever a user has contributed at least one annotation to a work. No edges join two nodes of the same class.

Given this graph we can discuss its topological features [1]. The graph has 216,943 nodes across both class – 71,129 of those nodes represent all the users that have contributed an annotation, 145,814 represent the works [15] . The graph contains 439,835 edges, representing the number of unique user-work pairs with annotations. While nearly half a million in number, this is quite sparse representing only 4.24×10^{-5} of the more than 10 billion possible pairs. Therefore the graph has a average degree of 2.02. This bipartite graph, serialised as graphml, is also included as part of the dataset and is available for download as mentioned in Section 3.

The remained of this section concerns the detection of communities of works. In order to do this, we project our bipartite graph to a songs-as-nodes single class graph with weighted edges representing the users that co-annotated linked works. We also only consider the largest connected component, i.e. the largest number of works for which there is a path between each pair of works included. This reduces the number of nodes to 125,044.

4.2 Examining Community

We have performed community detection with three different algorithms: fast greedy [5], leading eigenvector [18], and multilevel [3]. In order to assess the suitability of each of these inferred community structures, we take the *modularity* of each. Here modularity is a measure of the ratio of connections within communities against connections among communities. The optimum modularity resulting from each of these communities detection methods, along with the number of detected communities that give said modularity, can be seen in Table 4. Based on modularity, the multilevel community detection measure gives the best grouping, resulting in 169 distinct detected communities.

method	modularity	communities
Fast greedy	0.529	498
Leading eigenvector	0.003	11488
Multilevel	**0.582**	169

Table 4. The optimum modularity scores of each of the three community detection methods used on the works graph. The highest modularity, achieved the multilevel method, is shown in bold.

category	community count	community skew
meta	16	90.0
law	12	70.0
tech	12	70.0
history	19	36.6
screen	24	35.5
r-b	23	35.0
x	24	23.3
country	19	22.0
sports	19	15.7
pop	38	8.8
news	35	8.4
unbranded	22	6.5
lit	59	5.6
rock	50	2.7
rap	143	1.2

Table 5. The spread of each genre, across detected communities.

While the higher modularity of the multilevel method is inline with previous research on other small-world graphs, the low score and high number of communities generated by the leading eigenvector method is notable and merits further investigation.

Given the 169 detected communities of works, we can compare these communities to the prescribed genre labels to see how (and if) they align. To do this, we generate a confusion matrix, analogous to what might be used to evaluate a automatic classification task. However, unlike in a common classification task, our confusion matrix is not square, having dimension so of 15 x 169 (the number of categories by the number of communities). Given the size of the confusion matrix, it is not practical to visualize the entire thing, rather we will consider it in the following reduction. [16] Since there are more than 10 times the detected communities as there are genre categories, we can see how widespread each genre category is across communities. That is, how many communities have more than zero works from a given genre. This can be seen in Table 5, which shows that spread seems to correspond with popularity of the genre label.

[15] The careful reader may notice that this is 372 works fewer than the 146,186 works reference in Table 1. This is due to those works URLs not resolving at the time of the crawl, most likely due to deletion of the work from the collection after the annotation was made.

[16] The raw confusion matrix is available for download as a csv at `http://genius-annotations.data.t-mus.org/`

Beyond the raw counts, we can examine the *community skew* of a category S_c which we define as

$$S_c = \frac{W_c}{W} * \frac{C_c}{C} \quad (1)$$

where W_c is the number of works in category c, W is the total number of works in the corpus, C_c is the number communities in the split with at least one work category c amongst its members, and C is the total number of communities in across the network. Community skew therefore gives a measure of how widely distributed a given category label is across communities, normalised to how popular that label is in the corpus. A community skew of 1 means that the number of communities covering a genre exactly mirrors its overall representation in the corpus. Further as the skew increase away from 1 it show a disproportionate capture of the communities across the network. Looking at the community skew in Table 5 this is especially the case for the meta, law, and tech categories. With a few exceptions, the more well represented in the dataset a genre is the less its skew. This relationship implies that with more works in a genre community annotators become more distinct.

5. CONCLUSIONS AND FUTURE WORK

We have introduced the Web community Genius, a collection of (mostly music related) textual works and criticism in the form of annotations. We described and data gathering methodology, and using that methodology, collect the annotation and works metadata for the activity of over 700,000 users, with just over 10% of them active contributors. We then modelled this dataset as a bipartite graph of works and users. This graph was then projected into a single class for community detection. When performing community detection, the multilevel method was found to perform best, with a modularity score of 0.582 finding 169 communities. Using these communities we examined the community skew of each genre across these communities of works. In these community measures, and skew in particular, we see that a genre's definition is clearer as it is more popular.

While there are many further avenues of research to take this dataset and these foundations in the future, one in particular stands out: hybrid-methods using content. Performing content analysis on the lyrics, such as reading comprehension or rhyme structure analysis [10], and then using the result in tandem with cultural structures as captured by this work's network models present many possible further insights to the organisation of music.

6. ACKNOWLEDGEMENTS

This work is supported in part via the Transforming Musicology UK AHRC grant, under the Digital Transformations in the Arts and Humanities scheme, number AH/L006820/1. We would also like to acknowledge and thank the anonymous peer reviewers for their help in refining this paper.

7. REFERENCES

[1] Yong-Yeol Ahn, Seungyeop Han, Haewoon Kwak, Sue Moon, and Hawoong Jeong. Analysis of topological characteristics of huge online social networking services. In *The World Wide Web Conference (WWW)*, Alberta, Canada, May 2007.

[2] Fabrício Benevenuto, Tiago Rodrigues, Meeyoung Cha, and Virgílio Almeida. Characterizing user behavior in online social networks. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 49–62. ACM, 2009.

[3] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[4] Xu Cheng, Cameron Dale, and Jiangchuan Liu. Statistics and social network of youtube videos. In *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on*, pages 229–238. IEEE, 2008.

[5] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111, Dec 2004.

[6] Ben Fields, Kurt Jacobson, Michael Casey, and Mark Sandler. Do you sound like your friends? exploring artist similarity via artist social network relationships and audio signal processing. In *International Computer Music Conference (ICMC)*, August 2008.

[7] Ben Fields, Kurt Jacobson, Christophe Rhodes, and Michael Casey. Social playlists and bottleneck measurements : Exploiting musician social graphs using content-based dissimilarity and pairwise maximum flow values. In *International Conference on Music Information Retrieval (ISMIR)*, September 2008.

[8] Ben Fields, Kurt Jacobson, Christophe Rhodes, Mark Sandler, Mark d'Inverno, and Michael Casey. Analysis and exploitation of musician social networks for recommendation and discovery. *IEEE Transactions on Multimedia*, PP(99):1, 2011.

[9] Jean-Loup Guillaume and Matthieu Latapy. Bipartite graphs as models of complex networks. *Physica A: Statistical Mechanics and its Applications*, 371(2):795–813, 2006.

[10] Hussein Hirjee and Daniel G Brown. Automatic detection of internal and imperfect rhymes in rap lyrics. In *International Society on Music Information Retrieval Conference (ISMIR)*, pages 711–716, 2009.

[11] Takashi Iba, Keiichi Nemoto, Bernd Peters, and Peter A Gloor. Analyzing the creative editing behavior of wikipedia editors: Through dynamic social network analysis. *Procedia-Social and Behavioral Sciences*, 2(4):6441–6456, 2010.

[12] Kurt Jacobson. *Connections in Music*. PhD thesis, Centre for Digital Music, Queen Mary University of London, 2011.

[13] Kurt Jacobson, Ben Fields, and Mark Sandler. Using audio analysis and network structure to identify communities of on-line social networks of artists. In *International Conference on Music Information Retrieval (ISMIR)*, Philadelphia, PA, USA, October 2008.

[14] Sanghoon Jun, Daehoon Kim, Mina Jeon, Seungmin Rho, and Eenjun Hwang. Social mix: automatic music recommendation and mixing scheme based on social network analysis. *The Journal of Supercomputing*, pages 1–22, 2014.

[15] Masahiro Kimura, Kazumi Saito, and Ryohei Nakano. Extracting influential nodes for information diffusion on a social network. In *AAAI*, volume 7, pages 1371–1376, 2007.

[16] Andrea Lancichinetti, Mikko Kivelä, Jari Saramäki, and Santo Fortunato. Characterizing the community structure of complex networks. *PloS one*, 5(8):e11976, 2010.

[17] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42. ACM, 2007.

[18] Mark EJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[19] Gerard Roma, Perfecto Herrera, Massimiliano Zanin, Sergio L Toral, Frederic Font, and Xavier Serra. Small world networks and creativity in audio clip sharing. *International Journal of Social Network Mining*, 1(1):112–127, 2012.

[20] Markus Schedl. On the use of microblogging posts for similarity estimation and artist labeling. In *International Conference on Music Information Retrieval (ISMIR)*, pages 447–452, 2010.

[21] Markus Schedl. Analyzing the potential of microblogs for spatio-temporal popularity estimation of music artists. In *In Proceedings of the IJCAI 2011: International workshop on social web mining*, 2011.

[22] Karsten Steinhaeuser and Nitesh V Chawla. Identifying and evaluating community structure in complex networks. *Pattern Recognition Letters*, 31(5):413–421, 2010.

LONG-TERM REVERBERATION MODELING FOR UNDER-DETERMINED AUDIO SOURCE SEPARATION WITH APPLICATION TO VOCAL MELODY EXTRACTION.

Romain Hennequin
Deezer R&D
10 rue d'Athènes, 75009 Paris, France
rhennequin@deezer.com

François Rigaud
Audionamix R&D
171 quai de Valmy, 75010 Paris, France
francois.rigaud@audionamix.com

ABSTRACT

In this paper, we present a way to model long-term reverberation effects in under-determined source separation algorithms based on a non-negative decomposition framework. A general model for the sources affected by reverberation is introduced and update rules for the estimation of the parameters are presented. Combined with a well-known source-filter model for singing voice, an application to the extraction of reverberated vocal tracks from polyphonic music signals is proposed. Finally, an objective evaluation of this application is described. Performance improvements are obtained compared to the same model without reverberation modeling, in particular by significantly reducing the amount of interference between sources.

1. INTRODUCTION

Under-determined audio source separation has been a key topic in audio signal processing for the last two decades. It consists in isolating different meaningful 'parts' of the sound, such as for instance the lead vocal from the accompaniment in a song, or the dialog from the background music and effects in a movie soundtrack. Non-negative decompositions such as Non-negative Matrix Factorization [5] and its derivative have been very popular in this research area for the last decade and have achieved state-of-the art performances [3, 9, 12].

In music recordings, the vocal track generally contains reverberation that is either naturally present due to the recording environment or artificially added during the mixing process. For source separation algorithms, the effects of reverberation are usually not explicitly modeled and thus not properly extracted with the corresponding sources. Some studies [1, 2] introduce a model for the effect of spatial diffusion caused by the reverberation for a multichannel source separation application. In [7] a model for the dereverberation of spectrograms is presented for the case of long reverberations, *i.e.* when the reverberation time is longer than the length of the analysis window.

We propose in this paper to extend the model of reverberation proposed in [7] to a source separation application that allows extracting the reverberation of a specific source together with its dry signal. The reverberation model is introduced first in a general framework for which no assumption is made about the spectrogram of the dry sources. At this state, and as often in demixing application, the estimation problem is ill-posed (optimization of a non-convex cost function with local minima, result highly dependent on the initialization, ...) and requires the incorporation of some knowledge about the source signals. In [7], this issue is dealt with a sparsity prior on the unreverberated spectrogram model. Alternatively, the spectrogram sources can be structured by using models of non-negative decompositions with constraints (*e.g.* harmonic structure of the source's tones, sparsity of the activations) and/or by guiding the estimation process with prior information (*e.g.* source activation, multi-pitch transcription). Thus in this paper we propose to combine the generic reverberation model with a well-known source/filter model of singing voice [3]. A modified version of the original voice extraction algorithm is described and evaluated on an application to the extraction of reverberated vocal melodies from polyphonic music signals.

Note that unlike usual application of reverberation modeling, we do not aim at extracting dereverbated sources but we try to extract accurately both the dry signal and the reverberation within the same track. Thus, the designation source separation is not completely in accordance with our application which targets more precisely *stem* separation.

The rest of the paper is organized as follows: Section 2 presents the general model for a reverberated source and Section 3 introduces the update rule used for its estimation. In Section 4, a practical implementation for which the reverberation model is combined with a source/filter model is presented. Then, Section 5 presents experimental results that demonstrate the ability of our algorithm to extract properly vocals affected by reverberation. Finally, conclusions are drawn in Section 6.

© Romain Hennequin, François Rigaud. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).
Attribution: Romain Hennequin, François Rigaud. "Long-term reverberation modeling for under-determined audio source separation with application to vocal melody extraction.", 17th International Society for Music Information Retrieval Conference, 2016. This work has been done when the first author was working for Audionamix.

Notation

- Matrices are denoted by bold capital letters: $\mathbf{M}$. The coefficients at row f and column t of matrix $\mathbf{M}$ is denoted by $\mathbf{M}_{f,t}$.
- Vectors are denoted by bold lower case letters: $\mathbf{v}$.
- Matrix or vector sizes are denoted by capital letters: T, whereas indexes are denoted with lower case letters: t.
- Scalars are denoted by italic lower case letters: s.
- $\odot$ stands for element-wise matrix multiplication (Hadamard product) and $\mathbf{M}^{\odot\lambda}$ stands for element-wise exponentiation of matrix $\mathbf{M}$ with exponent λ.

2. GENERAL MODEL

For the sake of clarity, we will present the signal model for mono signals only although it can be easily generalized to multichannel signals as in [6]. In the experimental Section 5, a stereo signal model is actually used.

2.1 Non-negative Decomposition

Most source separation algorithms based on a non-negative decomposition assume that the non-negative mixture spectrogram $\mathbf{V}$ (usually the modulus or the squared modulus of a time-frequency representation such as the Short Time Fourier Transform (STFT)) which is a $F \times T$ non-negative matrix, can be approximated as the sum of K source model spectrograms $\hat{\mathbf{V}}^k$, which are also non-negative:

$$\mathbf{V} \approx \hat{\mathbf{V}} = \sum_{k=1}^{K} \hat{\mathbf{V}}^k \tag{1}$$

Various structured matrix decomposition have been proposed for the source models $\hat{\mathbf{V}}^k$, such as, to name a few, standard NMF [8], source/filter modeling [3] or harmonic source modeling [10].

2.2 Reverberation Model

In the time-domain, a time-invariant reverberation can be accurately modeled using a convolution with a filter and thus be written as:

$$\mathbf{y} = \mathbf{h} * \mathbf{x}, \tag{2}$$

where $\mathbf{x}$ is the dry signal, $\mathbf{h}$ is the impulse response of the reverberation filter and $\mathbf{y}$ is the reverberated signal.

For short-term convolution, this expression can be approximated by a multiplication in the frequency domain such as proposed in [6] :

$$\overline{\mathbf{y}}_t = \overline{\mathbf{h}} \odot \overline{\mathbf{x}}_t, \tag{3}$$

where $\overline{\mathbf{x}}_t$ (respectively $\overline{\mathbf{y}}_t$) is the modulus of the t-th frame of the STFT of $\mathbf{x}$ (respectively $\mathbf{y}$) and $\overline{\mathbf{h}}$ is the modulus of the Fourier transform of $\mathbf{h}$.

For long-term convolution, this approximation does not hold. The support of typical reverberation filters are generally greater than half a second which is way too long for a STFT analysis window in this kind of application. In this case, as suggested in [7], we can use an other approximation which is a convolution in each frequency channel :

$$\overline{\mathbf{y}}^f = \overline{\mathbf{h}}^f * \overline{\mathbf{x}}^f, \tag{4}$$

Where $\overline{\mathbf{y}}^f$, $\overline{\mathbf{h}}^f$ and $\overline{\mathbf{x}}^f$ are the f-th frequency channel of the STFT of respectively $\mathbf{y}$, $\mathbf{h}$ and $\mathbf{x}$.

Then, starting from a dry spectrogram model $\hat{\mathbf{V}}^{\text{dry},k}$ of a source with index k, the reverberated model of the same source is obtained using the following non-negative approximation:

$$\hat{\mathbf{V}}^{\text{rev},k}_{f,t} = \sum_{\tau=1}^{T_k} \hat{\mathbf{V}}^{\text{dry},k}_{f,t-\tau+1} \mathbf{R}^k_{f,\tau} \tag{5}$$

where $\mathbf{R}^k$ is the $F \times T_k$ non-negative reverberation matrix of model k to be estimated.

The model of Equation (5) makes it possible to take long-term effects of reverberation into account and generalizes short-term convolution models as proposed in [6] since when $T_k = 1$, the model corresponds to the short-term convolution approximation.

3. ALGORITHM

3.1 Non-negative decomposition algorithms

The approximation of Equation (1) is generally quantified using a divergence (a measure of dissimilarity) between $\mathbf{V}$ and $\hat{\mathbf{V}}$ to be minimized with respect to the set of parameters $\mathbf{\Lambda}$ of all the models:

$$\mathcal{C}(\mathbf{\Lambda}) = D(\mathbf{V}|\hat{\mathbf{V}}(\mathbf{\Lambda})) \tag{6}$$

A commonly used class of divergence is the element-wise β-divergence which encompasses the Itakura-Saito divergence ($\beta = 0$), the Kullback-Leibler divergence ($\beta = 1$) and the squared Frobenius distance ($\beta = 2$) [4]. The global cost then writes:

$$\mathcal{C}(\mathbf{\Lambda}) = \sum_{f,t} d_\beta(\mathbf{V}_{f,t}|\hat{\mathbf{V}}_{f,t}(\mathbf{\Lambda})). \tag{7}$$

The problem being not convex, the minimization is generally done using alternating update rules on each parameters of Λ. The update rule for a parameter $\mathbf{\Theta}$ is commonly obtained using an heuristic consisting in decomposing the gradient of the cost-function with respect to this parameter as a difference of two positive terms, such as

$$\nabla_{\mathbf{\Theta}}\mathcal{C} = P_{\mathbf{\Theta}} - M_{\mathbf{\Theta}}, \quad P_{\mathbf{\Theta}} \geq 0,\ M_{\mathbf{\Theta}} \geq 0, \tag{8}$$

and then by updating the parameter according to:

$$\mathbf{\Theta} \leftarrow \mathbf{\Theta} \odot \frac{M_{\mathbf{\Theta}}}{P_{\mathbf{\Theta}}}. \tag{9}$$

This kind of update rule ensures that the parameter remains non-negative. Moreover the parameter is updated in a direction descent or remains constant if the partial derivative is zero. In some cases (including the update rules

we will present), it is possible to prove using a Majorize-Minimization (MM) approach [4] that the multiplicative update rules actually lead to a decrease of the cost function.

Using such an approach, the update rules for a standard NMF model $\hat{\mathbf{V}} = \mathbf{WH}$ can be expressed as:

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T \left(\hat{\mathbf{V}}^{\odot\beta-2} \odot \mathbf{V}\right)}{\mathbf{W}^T \hat{\mathbf{V}}^{\odot\beta-1}}, \quad (10)$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\left(\hat{\mathbf{V}}^{\odot\beta-2} \odot \mathbf{V}\right) \mathbf{H}^T}{\hat{\mathbf{V}}^{\odot\beta-1} \mathbf{H}^T}. \quad (11)$$

3.2 Estimation of the reverberation matrix

When dry models $\hat{\mathbf{V}}^{\text{dry},k}$ are fixed, the reverberation matrix can be estimated using the following update rule applied successively on each reverberation matrix:

$$\mathbf{R}^k \leftarrow \mathbf{R}^k \odot \frac{\left(\hat{\mathbf{V}}^{\odot\beta-2} \odot \mathbf{V}\right) *_t \hat{\mathbf{V}}^{\text{dry},k}}{\hat{\mathbf{V}}^{\odot\beta-1} *_t \hat{\mathbf{V}}^{\text{dry},k}} \quad (12)$$

where $*_t$ stands for time-convolution:

$$\left[\hat{\mathbf{V}}^{\odot\beta-1} *_t \hat{\mathbf{V}}^{\text{dry},k}\right]_{f,\tau} = \sum_{\tau=t}^{T} \hat{\mathbf{V}}_{f,\tau}^{\odot\beta-1} \hat{\mathbf{V}}_{f,\tau-t+1}^{\text{dry},k}. \quad (13)$$

The update rule (12) obtained using the procedure described in Section 3.1 ensures the non-negativity of $\mathbf{R}^k$. This update can be obtained using the MM approach which ensures that the cost-function will not increase.

3.3 Estimation with other free model

A general drawback of source separation models that do not explicitly account for reverberation effects is that the reverberation affecting a given source is usually spread among all separated sources. However, this issue can still arise with a proper model of reverberation if the dry model of the reverberated source is not constrained enough. Indeed using the generic reverberation model of Equation (5), the reverberation of a source can still be incorrectly modeled during the optimization by other source models having more degrees of freedom. A possible solution for enforcing a correct optimization of the reverberated model is to further constrain the structure of the dry model spectrogram, *e.g.* through the inclusion of sparsity or pitch activation constraints, and potentially to adopt a sequential estimation scheme. For instance, first discarding the reverberation model, a first rough estimate of the dry source may be produced. Second, considering the reverberation model, the dry model previously estimated can be refined while estimating at the same time the reverberation matrix. Such an approach is described in the following section for a practical implementation of the algorithm to the problem of lead vocal extraction from polyphonic music signals.

4. APPLICATION TO VOICE EXTRACTION

In this section we propose an implementation of our reverberation model in a practical case: we use Durrieu's algorithm [3] for lead vocal isolation and add the reverberation model over the voice model.

4.1 Base voice extraction algorithm

Durrieu's algorithm for lead vocal isolation in a song is based on a source/filter model for the voice.

4.1.1 Model

The non-negative mixture spectrogram model consists in the sum of a voice spectrogram model based on a source/filter model and a music spectrogram model based on a standard NMF:

$$\mathbf{V} \approx \hat{\mathbf{V}} = \hat{\mathbf{V}}^{\text{voice}} + \hat{\mathbf{V}}^{\text{music}}. \quad (14)$$

The voice model is based on a source/filter speech production model:

$$\hat{\mathbf{V}}^{\text{voice}} = (\mathbf{W}_{\text{F0}}\mathbf{H}_{\text{F0}}) \odot (\mathbf{W}_{\text{K}}\mathbf{H}_{\text{K}}). \quad (15)$$

The first factor $(\mathbf{W}_{\text{F0}}\mathbf{H}_{F0})$ is the source part corresponding to the excitation of the vocal folds: $\mathbf{W}_{\text{F0}}$ is a matrix of fixed harmonic atoms and $\mathbf{H}_{\text{F0}}$ is the activation of these atoms over time. The second factor $(\mathbf{W}_{\text{K}}\mathbf{H}_{\text{K}})$ is the filter part corresponding to the resonance of the vocal tract: $\mathbf{W}_{\text{K}}$ is a matrix of smooth filter atoms and $\mathbf{H}_{\text{K}}$ is the activation of these atoms over time.

The background music model is a generic NMF:

$$\hat{\mathbf{V}}^{\text{music}} = \mathbf{W}_{\text{R}}\mathbf{H}_{\mathbf{R}}. \quad (16)$$

4.1.2 Algorithm

Matrices $\mathbf{H}_{\text{F0}}$, $\mathbf{W}_{\text{K}}$, $\mathbf{H}_{\text{K}}$, $\mathbf{W}_{\text{R}}$ and $\mathbf{H}_{\text{R}}$ are estimated minimizing the element-wise Itakura-Saito divergence between the original mixture power spectrogram and the mixture model:

$$\mathcal{C}(\mathbf{H}_{\text{F0}}, \mathbf{W}_{\text{K}}, \mathbf{H}_{\text{K}}, \mathbf{W}_{\text{R}}, \mathbf{H}_{\text{R}}) = \sum_{f,t} d_{\text{IS}}(\mathbf{V}_{f,t} | \hat{\mathbf{V}}_{f,t}), \quad (17)$$

where $d_{\text{IS}}(x, y) = \frac{x}{y} - \log(\frac{x}{y}) - 1$. The minimization is achieved using multiplicative update rules.

The estimation is done in three steps:

1. A first step of parameter estimation is done using iteratively the multiplicative update rules.

2. The matrix $\mathbf{H}_{\text{F0}}$ is processed using a Viterbi decoding for tracking the main melody and is then thresholded so that coefficients too far from the melody are set to zero.

3. Parameters are re-estimated as in the first step but using the thresholded version of $\mathbf{H}_{\text{F0}}$ for the initialization.

4.2 Inclusion of the reverberation model

As stated in Section 3, the dry spectrogram model (*i.e.* the spectrogram model for the source without reverberation) has to be sufficiently constrained in order to accurately estimate the reverberation part. This constraint is here obtained through the use of a fixed harmonic dictionary $\mathbf{W}_{\text{F0}}$ and mostly, by the thresholding of the matrix $\mathbf{H}_{\text{F0}}$ that enforces the sparsity of the activations.

We thus introduce the reverberation model after the step of thresholding of the matrix $\mathbf{H}_{\text{F0}}$. The two first steps then remains the same as presented in Section 4.1.2. In the third step, the dry voice model of Equation (15) is replaced by a reverberated voice model following Equation (5):

$$\hat{\mathbf{V}}_{f,t}^{\text{rev. voice}} = \sum_{\tau=1}^{T} \hat{\mathbf{V}}_{f,t-\tau+1}^{\text{voice}} \mathbf{R}_{f,\tau}. \tag{18}$$

For the parameter re-estimation of step 3, the multiplicative update rule of $\mathbf{R}$ is given by Equation (12). For the other parameters of the voice model, the update rules from [3] are modified to take the reverberation model into account:

$$\mathbf{H}_{\text{F0}} \leftarrow \mathbf{H}_{\text{F0}} \odot \frac{\mathbf{W}_{\text{F0}}^T \left((\mathbf{W}_{\text{K}}\mathbf{H}_{\text{K}}) \odot \left(\mathbf{R} *_t (\hat{\mathbf{V}}^{\odot\beta-2} \odot \mathbf{V}) \right) \right)}{\mathbf{W}_{\text{F0}}^T \left((\mathbf{W}_{\text{K}}\mathbf{H}_{\text{K}}) \odot \left(\mathbf{R} *_t \hat{\mathbf{V}}^{\odot\beta-1} \right) \right)} \tag{19}$$

$$\mathbf{H}_{\text{K}} \leftarrow \mathbf{H}_{\text{K}} \odot \frac{\mathbf{W}_{\text{K}}^T \left((\mathbf{W}_{\text{F0}}\mathbf{H}_{\text{F0}}) \odot \left(\mathbf{R} *_t (\hat{\mathbf{V}}^{\odot\beta-2} \odot \mathbf{V}) \right) \right)}{\mathbf{W}_{\text{K}}^T \left((\mathbf{W}_{\text{F0}}\mathbf{H}_{\text{F0}}) \odot \left(\mathbf{R} *_t \hat{\mathbf{V}}^{\odot\beta-1} \right) \right)} \tag{20}$$

$$\mathbf{W}_{\text{K}} \leftarrow \mathbf{W}_{\text{K}} \odot \frac{\left((\mathbf{W}_{\text{F0}}\mathbf{H}_{\text{F0}}) \odot \left(\mathbf{R} *_t (\hat{\mathbf{V}}^{\odot\beta-2} \odot \mathbf{V}) \right) \right) \mathbf{H}_{\text{K}}^T}{\left((\mathbf{W}_{\text{F0}}\mathbf{H}_{\text{F0}}) \odot \left(\mathbf{R} *_t \hat{\mathbf{V}}^{\odot\beta-1} \right) \right) \mathbf{H}_{\text{K}}^T} \tag{21}$$

The update rules for the parameters of the music model ($\mathbf{H}_{\text{R}}$ and $\mathbf{W}_{\text{R}}$), are unchanged and thus identical to those given in Equations (10) and (11).

5. EXPERIMENTAL RESULTS

5.1 Experimental setup

We tested the reverberation model that we proposed with the algorithm presented in Section 4 on a task of lead vocal extraction in a song. In order to assess the improvement of our model over the existing one, we ran the separation with and without reverberation modeling.

We used a database composed of 9 song excerpts of professionally produced music. The total duration of all excerpts was about 10 minutes. As the use of reverberation modeling only makes sense if there is a significant amount of it, all the selected excerpts contains a fair amount of reverberation. This reverberation was already present in the separated tracks and was not added artificially by ourselves. On some excerpts, the reverberation is time-variant: active on some parts and inactive on other, ducking echo effect ... Some short excerpts, as well as the separation results, can be played on the companion website [1] .

Spectrograms were computed as the squared modulus of the STFT of the signal sampled at 44100 Hz, with 4096-sample (92.9 ms) long Hamming window with 75% overlap. The length T of the reverberation matrix was arbitrarily fixed to 52 frames (which corresponds to about 1.2 s) in order to be sufficient for long reverberations.

5.2 Results

In order to quantify the results we use standard metrics of source separation as described in [11]: Signal to Distorsion Ratio (SDR), Signal to Artefact Ratio (SAR) and Signal to Interference Ratio (SIR).

The results are presented in Figure 1 for the evaluation of the extracted voice signals and in Figure 2 for the extracted music signals. The oracle performance, obtained using the actual spectrograms of the sources to compute the separation masks, are also reported. As we can see, adding the reverberation modeling increases all these metrics. The SIR is particularly increased in Figure 1 (more than 5dB): this is mainly because without the reverberation model, a large part of the reverberation of the voice leaks in the music model. This is a phenomenon which is also clearly audible in excerpts with strong reverberation: using the reverberation model, the long reverberation tail is mainly heard within the separated voice and is almost not audible within the separated music. In return, extracted vocals with the reverberation model tend to have more audible interferences. This result is in part due to the fact that the pre-estimation of the dry model (step 1 and 2 of the base algorithm) is not interference-free, so that applying the reverberation model increases the energy of these interferences.

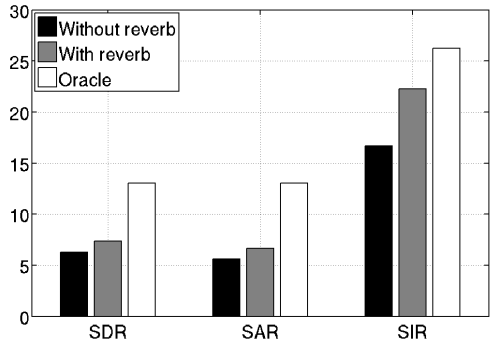


Figure 1. Experimental separation results for the voice stem.

[1] http://romain-hennequin.fr/En/demo/reverb_separation/reverb.html

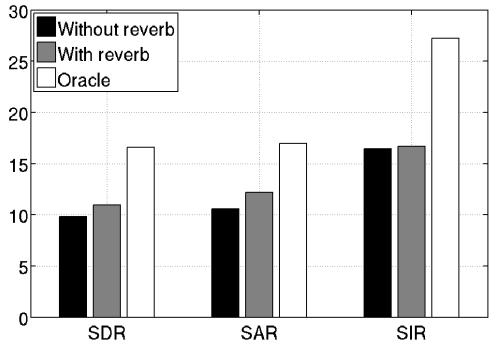


Figure 2. Experimental separation results for the music stem.

6. CONCLUSION

In this paper we proposed a method to model long-term effects of reverberation in a source separation application for which a constrained model of the dry source is available. Future work should focus on speeding up the algorithm since, multiple convolutions at each iteration can be time-consuming. Developing methods to estimate the reverberation duration (of a specific source within a mix) would also make it possible to automate the whole process. It could also be interesting to add spatial modeling for multichannel processing using full rank spatial variance matrix and multichannel reverberation matrices.

7. REFERENCES

[1] Simon Arberet, Alexey Ozerov, Ngoc Q. K. Duong, Emmanuel Vincent, Frédéric Bimbot, and Pierre Vandergheynst. Nonnegative matrix factorization and spatial covariance model for under-determined reverberant audio source separation. In *International Conference on Information Sciences, Signal Processing and their applications*, pages 1–4, May 2010.

[2] Ngoc Q. K. Duong, Emmanuel Vincent, and Rémi Gribonval. Under-determined reverberant audio source separation using a full-rank spatial covariance model. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(7):1830 – 1840, Sept 2010.

[3] Jean-Louis Durrieu, Gaël Richard, and Bertrand David. An iterative approach to monaural musical mixture de-soloing. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 105–108, Taipei, Taiwan, April 2009.

[4] Cédric Févotte and Jérôme Idier. Algorithms for nonnegative matrix factorization with the beta-divergence. *Neural Computation*, 23(9):2421–2456, September 2011.

[5] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, October 1999.

[6] Alexey Ozerov and Cédric Févotte. Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 18(3):550–563, 2010.

[7] Rita Singh, Bhiksha Raj, and Paris Smaragdis. Latent-variable decomposition based dereverberation of monaural and multi-channel signals. In *IEEE International Conference on Audio and Speech Signal Processing*, Dallas, Texas, USA, March 2010.

[8] Paris Smaragdis and Judith C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177 – 180, New Paltz, NY, USA, October 2003.

[9] Paris Smaragdis, Bhiksha Raj, and Madhusudana Shashanka. Supervised and semi-supervised separation of sounds from single-channel mixtures. In *7th International Conference on Independent Component Analysis and Signal Separation*, London, UK, September 2007.

[10] Emmanuel Vincent, Nancy Bertin, and Roland Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech and Language Processing*, 18(3):528–537, March 2010.

[11] Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 14(4):1462–1469, July 2006.

[12] Tuomas Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1066–1074, March 2007.

NONNEGATIVE TENSOR FACTORIZATION WITH FREQUENCY MODULATION CUES FOR BLIND AUDIO SOURCE SEPARATION

Elliot Creager[1,3] **Noah D. Stein**[1] **Roland Badeau**[2,3] **Philippe Depalle**[3]
[1] Analog Devices Lyric Labs, Cambridge, MA, USA
[2] LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay, Paris, France
[3] CIRMMT, McGill University, Montréal, Canada
elliot.creager@analog.com, noah.stein@analog.com,
roland.badeau@telecom-paristech.fr, depalle@music.mcgill.ca

ABSTRACT

We present Vibrato Nonnegative Tensor Factorization, an algorithm for single-channel unsupervised audio source separation with an application to separating instrumental or vocal sources with nonstationary pitch from music recordings. Our approach extends Nonnegative Matrix Factorization for audio modeling by including local estimates of frequency modulation as cues in the separation. This permits the modeling and unsupervised separation of vibrato or glissando musical sources, which is not possible with the basic matrix factorization formulation.

The algorithm factorizes a sparse nonnegative tensor comprising the audio spectrogram and local frequency-slope-to-frequency ratios, which are estimated at each time-frequency bin using the Distributed Derivative Method. The use of local frequency modulations as separation cues is motivated by the principle of common fate partial grouping from Auditory Scene Analysis, which hypothesizes that each latent source in a mixture is characterized perceptually by coherent frequency and amplitude modulations shared by its component partials. We derive multiplicative factor updates by Minorization-Maximization, which guarantees convergence to a local optimum by iteration. We then compare our method to the baseline on two separation tasks: one considers synthetic vibrato notes, while the other considers vibrato string instrument recordings.

1. INTRODUCTION

Nonnegative matrix factorization (NMF) [11] is a popular method for the analysis of audio spectrograms [16], especially for audio source separation [17]. NMF models the observed spectrogram as a weighted sum of rank-1 latent components, each of which factorizes as the outer product of a pair of vectors representing the constituent frequencies and onset regions for some significant component in the mixture, e.g. a musical note. Equivalently, the entire spectrogram matrix approximately factorizes as a matrix of spectral templates times a matrix of temporal activations, typically such that the approximate factors have many fewer elements than the full observation. While NMF can be used for supervised source separation tasks with a straightforward extension of the signal model [19], this necessitates pre-training NMF representations for each source of interest.

The use of modulation cues in source separation is popular in the Computational Auditory Scene Analysis (CASA) [26] literature, which, unlike NMF, typically relies on partial tracking. E.g., [25] isolates individual partials by frequency warping and filtering, while [12] groups partials via correlations in amplitude modulations. [2], which more closely resembles our work in the sense of being data-driven, factorizes a tensor encoding amplitude modulations for speech separation.

Our approach is inspired by [20] and [21], which present a Nonnegative Tensor Factorization (NTF) incorporating direction-of-arrival (DOA) estimates in an unsupervised speech source separation task. Whereas use of DOA information in that work necessitates multi-microphone data, we address the single-channel case by incorporating the local frequency modulation (FM) cues at each time-frequency bin. These cues are combined with the spectrogram as a sparse observation tensor, which we factorize in a probabilistic framework. The modulation cues are adopted structurally by way of an NTF where each source in the mixture is modeled via an NMF factor and a time-varying FM factor.

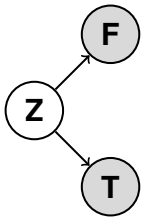
2. BACKGROUND

2.1 Nonnegative matrix factorization

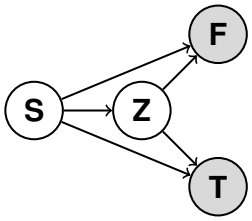
We now summarize NMF within a probabilistic framework. We consider the normalized Short-Time Fourier Transform (STFT) magnitudes (i.e., spectrogram) of the input signal as an observed discrete probability distribution of energy over the time-frequency plane, i.e.,

$$p^{\text{obs}}(f,t) \triangleq \frac{|X(f,t)|}{\sum_{\nu,\tau}|X(\nu,\tau)|}, \tag{1}$$

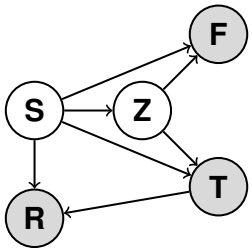
© Elliot Creager, Noah D. Stein, Roland Badeau, Philippe Depalle. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Elliot Creager, Noah D. Stein, Roland Badeau, Philippe Depalle. "Nonnegative tensor factorization with frequency modulation cues for blind audio source separation", 17th International Society for Music Information Retrieval Conference, 2016.



(a) Basic NMF (Section 2.1)



(b) NMF for source separation (Section 2.2)



(c) Vibrato NTF (Section 3.3)

Figure 1: Graphical models for the factorizations in this paper. In each case the input data are a distribution over the observed (shaded) variables, while the model approximates the observation by a joint distribution over observed and latent (unshaded) variables that factorizes as specified. F, T, Z, S, and R respectively represent the discrete frequencies, hops, components, sources, and frequency modulations over which the data is distributed.

$\forall f \in \{1, ..., F\}, t \in \{1, ..., T\}$, where X is the input STFT and (f, t) indexes the time-frequency plane. NMF seeks an approximation q to observed distribution p^{obs} that is a valid distribution over the time-frequency plane and factorizes as

$$q(f,t) = \sum_z q(f|z)q(t|z)q(z) = \sum_z q(f|z)q(z,t). \quad (2)$$

Figure 1(a) shows the graphical model for a joint distribution with this factorization.

We have introduced $z \in \{1, ..., Z\}$ as a latent variable that indexes components in the mixture, typically with Z chosen to yield an overall data reduction, i.e., $FZ + ZT \ll FT$. For a fixed z_0, $q(f|z_0)$ is a vector interpreted as the spectral template of the z_0-th component, i.e., the distribution over frequency bins of energy belonging to that component. Likewise, $q(z_0, t)$ is interpreted as a vector of temporal activations of the z_0-th component, i.e., it specifies at what time indices the z_0-th component is prominent in the observed mixture. Indeed, (2) can be implemented as a matrix multiplication, with the usual nonnegativity constraint on the factors satisfied implicitly, since q is a valid probability distribution.

The optimization problem is typically formalized as minimizing the Kullback-Leibler (KL) divergence between the observation and approximation, or equivalently as maximizing the cross entropy between the two distributions:

$$\begin{aligned} \underset{q}{\text{maximize}} \quad & \sum_{f,t} p^{\text{obs}}(f,t) \log q(f,t) \\ \text{subject to} \quad & q(f,t) = \sum_z q(f|z)q(z,t). \end{aligned} \quad (3)$$

While the non-convexity of this problem prohibits a globally optimal solution in reasonable time, a locally optimal solution can be found by multiplicative updates to the factors, which were first presented in [10]. We refer to this algorithm as KL-NMF, but note its equivalence to Probabilistic Latent Component Analysis (PLCA) [18], as well as a strong connection to topic modeling of counts data.

2.2 NMF for source separation

NMF can be leveraged as a source model within a source separation task, such that the observed mixture is modeled as a sum of sources, each of which is modeled by NMF. Whereas the latent variable z in NMF indexes latent components belonging to a source, we now introduce an additional latent variable $s \in \{1, .., S\}$, which indexes latent sources within the mixture. The resulting joint distribution over observed and latent variables is expressed as

$$q(f,t,s,z) = q(s)q(f|s,z)q(z,t|s). \quad (4)$$

Thus the approximation to $p^{obs}(f,t)$ is the marginal distribution

$$\begin{aligned} q(f,t) &= \sum_s q(s)q(f,t|s) \\ &= \sum_s q(s) \sum_z q(f|s,z)q(z,t|s), \end{aligned} \quad (5)$$

where $q(s_0)$ and $q(f,t|s_0)$ represent the mixing coefficient and NMF source model for the s_0-th source in the mixture, respectively. Figure 1(b) shows the graphical model.

Given a suitable approximation q, we estimate the latent sources in the mixture via Wiener filtering, i.e.,

$$X_s(f,t) = X(f,t)q(s|f,t), \quad (6)$$

where the Wiener gains $q(s|f,t)$ are given by the conditional probabilities [1] of the latent sources given the approximating joint distribution

$$q(s|f,t) = \frac{q(f,t,s)}{q(f,t)} = \frac{\sum_z q(s)q(f|s,z)q(z,t|s)}{\sum_{z,s'} q(s')q(f|s',z)q(z,t|s')}. \quad (7)$$

The estimated sources can then be reconstructed in the time-domain via the inverse STFT.

We seek a q that both approximates p^{obs} and yields source estimates $q(f,t|s)$ close to the true sources. In a supervised setting, the spectral templates for each source model can be fixed by using basic NMF on some characteristic training examples in isolation. When the appropriate training data is unavailable, the basic NMF can

[1] A convenient result of the Wiener filter gains being conditional distributions over sources is that the mixture energy is conserved by the source estimates in the sense that $\sum_s X_s(f,t) = X(f,t) \; \forall \, f, t$.

be extended by introducing priors on the factors or otherwise adding structure to the observation model to encourage, e.g., smoothness in the activations [24] or harmonicity in the spectral templates [3], which hopefully in turn improves the source estimates. By contrast, our approach exploits local FM cues directly in the factorization, yielding an observation model for latent sources consistent with the sorts of pitch modulations expected in musical sounds.

2.3 Coherent frequency modulation

We now introduce frequency-slope-to-frequency ratios (FSFR) as local signal parameters under an additive sinusoidal model that are useful as grouping cues for the separation of sources with coherent FM, e.g. in the vibrato or glissando effects. In continuous time, the additive sinusoidal model expresses the s-th source as a sum of component partials, [2] each parameterized by an instantaneous frequency and amplitude, i.e.,

$$x_s(\tau) = \sum_{p=1}^{P} A_p(\tau) \cos\left(\theta_p(\tau_0) + \int_{\tau_0}^{\tau} \omega_p(u)du\right) \quad (8)$$

where p is the partial index, and $\theta_p(\tau_0)$, $A_p(\tau)$ and $\omega_p(\tau)$ specify the initial phase, instantaneous amplitude, and instantaneous frequency of the p-th partial.

We now consider a source under coherent FM, i.e.,

$$\omega_p(\tau) \triangleq (1 + \kappa_s(\tau))\omega_p(\tau_0) \,\forall\, p \quad (9)$$

for some modulation function κ_s with $\kappa_s(\tau_0) = 0$. E.g., κ_s resembles a slowly-varying sinusoid during frequency vibrato, or a gradual ramp function during glissando. The FSFR are then expressed as

$$\upsilon_p(\tau) \triangleq \frac{\omega_p'(\tau)}{\omega_p(\tau)} = \frac{\kappa_s'(\tau)}{1 + \kappa_s(\tau)}. \quad (10)$$

Note that $\{\upsilon_p(\tau)\}$ are time-varying but independent of the partial index p for a given source index s. In other words, the instantaneous FSFR is common to all partials belonging to the same source and can be used as a grouping cue in unsupervised source separation [7].

2.4 Distributed Derivative Method

We now summarize the Distributed Derivative Method (DDM) [4, 8] for signal parameter estimation, which we use to estimate the FSFR at each time-frequency bin. DDM estimates the parameters of a monochrome analytic signal under a Q-th order generalized sinusoid model, [3] which is expressed as

$$x(\tau) = \exp\left(\sum_{q=0}^{Q} \eta_q \tau^q\right), \quad (11)$$

where $\boldsymbol{\eta} \in \mathbb{C}^{Q+1}$ is the vector of signal parameters, whose real and imaginary parts specify the log amplitude law and phase law, [4] respectively. In this work, we specify (11) as a constant amplitude signal with linear frequency modulation, i.e., $\boldsymbol{\eta} \in \mathbb{C}^3$ with $\Re(\eta_i) = 0 \;\; \forall \;\; i$. The signal parameters $\Im(\eta_1)$ and $\Im(\eta_2)$ then specify (within multiplicative constants) the instantaneous frequency and frequency slope, respectively.

The parameters of interest can be estimated by considering the inner product of the signal with a family of differentiable analysis atoms of finite time-frequency support. In particular, the continuous-time STFT can be expressed by inner product as

$$\mathcal{X}(f,t) \triangleq \langle x(\tau), \phi(\tau; f, t)\rangle = \int_{\tau=-\infty}^{+\infty} x(\tau)\phi(\tau; f, t)^* d\tau, \quad (12)$$

where $\mathcal{X}(f,t)$ is the STFT, $x(\tau)$ is the input signal, and $\phi(\tau; f, t)$ is a heterodyned window function from some differentiable family (e.g. Hann), parameterized by its localization (f,t) in the time-frequency plane. The signal parameters are solutions to equations of the form

$$\langle x(\tau), \phi'(\tau; f, t)\rangle = -\sum_{q=1}^{Q} \eta_q \langle q\tau^{q-1} x(\tau), \phi(\tau; f, t)\rangle, \quad (13)$$

which is linear in $\{\eta_q\}$ for $q > 0$, and permits an STFT-like computation of both inner products. The right-hand side of (13) is derived from the left-hand side using integration by parts, exploiting the finite support of $\phi(\tau; f, t)$, and substituting in the signal derivative $x'(\tau)$ from (11). To estimate the signal parameters at a particular (f_0, t_0), we construct a system of linear equations by evaluating (13) for each $\phi(\tau; f, t)$ in a set of nearby atoms Φ, then solve for $\boldsymbol{\eta}$ in a least-squares sense. We typically use atoms in neighboring frequency bins at the same time step, i.e., $\Phi = \{\phi(\tau; t_0, f_0 - \frac{L-1}{2}), \; ..., \; \phi(\tau; t_0, f_0 + \frac{L-1}{2})\}$ for some odd L.

While DDM is an unbiased estimator of the signal parameters in continuous time, we must implement a discrete-time approximation on a computer. This introduces a small bias that can be ignored in practice since the STFT window is typically longer than a few samples [4].

3. PROPOSED METHOD

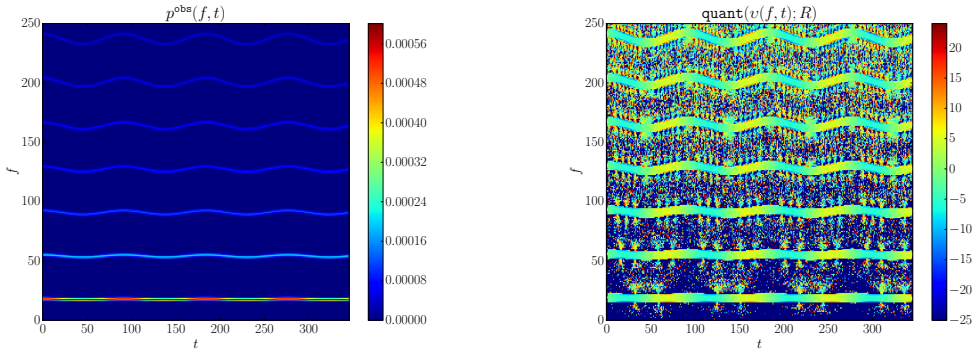
3.1 Motivation

The NMF signal model is not sufficiently expressive to compactly represent a large class of musical sounds, namely those characterized by slow frequency modulations, e.g., in the vibrato effect. In particular, it specifies a single fixed spectral template per latent component

[2] We do not assume any special structure in the partial frequencies, e.g., harmonicity.

[3] It is natural to specify the signal locally (near some time-frequency bin) as a generalized sinusoid even while the global model remains additive sinusoidal. In particular, the notion of a time-frequency-localized signal follows from the filterbank summation interpretation of the STFT, and corresponds to the heterodyned and shifted input, prior to low-pass filtering by the window and downsampling in time [1]. In a slight abuse of notation, we later absorb the time-frequency indices as parameters in the analysis atom, i.e., we switch to the overlap-add interpretation of the STFT without warning.

[4] The frequency law is trivially computed from the phase law.



(a) Spectrogram (b) Discretized FSFR

Figure 2: Unfolding the nonzero elements in the observation tensor for a synthetic vibrato square wave note (G5). The hop index t spans 2 seconds of the input audio, while the bin index f spans half the sampling rate, 0–22.05 kHz.

and thus requires a large number of components to model sounds with nonstationary pitch. From a separation perspective, as the number of latent components grows, so grows the need for a comprehensive model that can correctly group components belonging to the same source. To this end, we appeal to the perceptual theory of Auditory Scene Analysis [5], which postulates the importance of shared frequency or amplitude modulations among partials as a perceptual cue in their grouping [6, 14]. In this work we focus on FM, although in principle our approach could be extended to include amplitude modulations. [5] We now propose an extension to KL-NMF that leverages this so-called common fate principle and is suitable for the analysis of vibrato signals.

3.2 Compiling the observations as a tensor

DDM yields the local estimates of frequency and frequency slope for each time-frequency bin, from which the FSFR are trivially computed. We define the (sparse) observation tensor $p^{\text{obs}}(f,t,r) \in \mathbb{R}_{\geq 0}^{F\times T\times R}$ as an assignment of the normalized spectrogram into one of R discrete bins for each (f,t) according the local FSFR estimate, i.e.,

$$p^{\text{obs}}(f,t,r) \triangleq \begin{cases} p^{\text{obs}}(f,t) & \text{if} \quad \texttt{quant}(\upsilon(f,t);R) = r \\ 0 & \text{else,} \end{cases} \tag{14}$$

where $p^{\text{obs}}(f,t)$ is the normalized spectrogram as in (1) and υ are the FSFR as in (10), which are quantized by $\texttt{quant}(\cdot;R)$, possibly after clipping to some reasonable range of values. Figure 2 shows the spectrogram and FSFR for a synthetic vibrato square wave.

3.3 Vibrato NTF

As with NMF, we seek a joint distribution q with a particular factorized form, whose marginal maximizes cross entropy against the observed data. We propose an observation model of the form

$$q(f,t,r) = \sum_s q(s)q(r|t,s)\sum_z q(f|s,z)q(z,t|s) \tag{15}$$

where $q(s)$ represents the mixing, $q(r|t,s)$ represents the common time-varying FSFR per source, and $\sum_z q(f|s,z)q(z,t|s)$ represents the NMF source model. Figure 1(c) shows the graphical model of the joint distribution. Thus, given p^{obs}, we seek an approximation q that factorizes as in (15) and maximizes

$$\begin{aligned} \alpha(q) &\triangleq \sum_{f,t,r} p^{\text{obs}}(f,t,r)\log q(f,t,r) \\ &= \sum_{f,t,r} p^{\text{obs}}(f,t,r)\log\sum_{z,s} q(f,t,r,z,s). \end{aligned} \tag{16}$$

The sum in the argument to the log makes this difficult to solve outright, so we find a local optimum by iterative Minorization-Maximization (MM) [9] instead. That is, given $q^{(i)}$, our model at the current (i-th) iteration, we pick a better $q^{(i+1)}$ by (a) finding a concave minorizing function $\beta(q;q^{(i)})$ such that $\beta(q;q^{(i)}) \leq \alpha(q)\ \forall\ q$ and $\beta(q^{(i)};q^{(i)}) = \alpha(q^{(i)})$, and (b) maximizing $\beta(q;q^{(i)})$ with respect to q.

In particular, $\beta(q;q^{(i)})$ is derived [6] by applying Jensen's inequality to (16), and is expressed as

$$\beta(q;q^{(i)}) \triangleq \sum_{f,t,r,z,s} p^{\text{obs}}(f,t,r)q^{(i)}(z,s|f,t,r)\log\frac{q(f,t,r,z,s)}{q^{(i)}(z,s|f,t,r)}, \tag{17}$$

where $q^{(i)}(z,s|f,t,r)$ is the approximate posterior over latent variables given the model at the i-th iteration [7], computed as

$$q^{(i)}(z,s|f,t,r) = \frac{q^{(i)}(z,s,f,t,r)}{\sum_{z',s'} q^{(i)}(z',s',f,t,r)}. \tag{18}$$

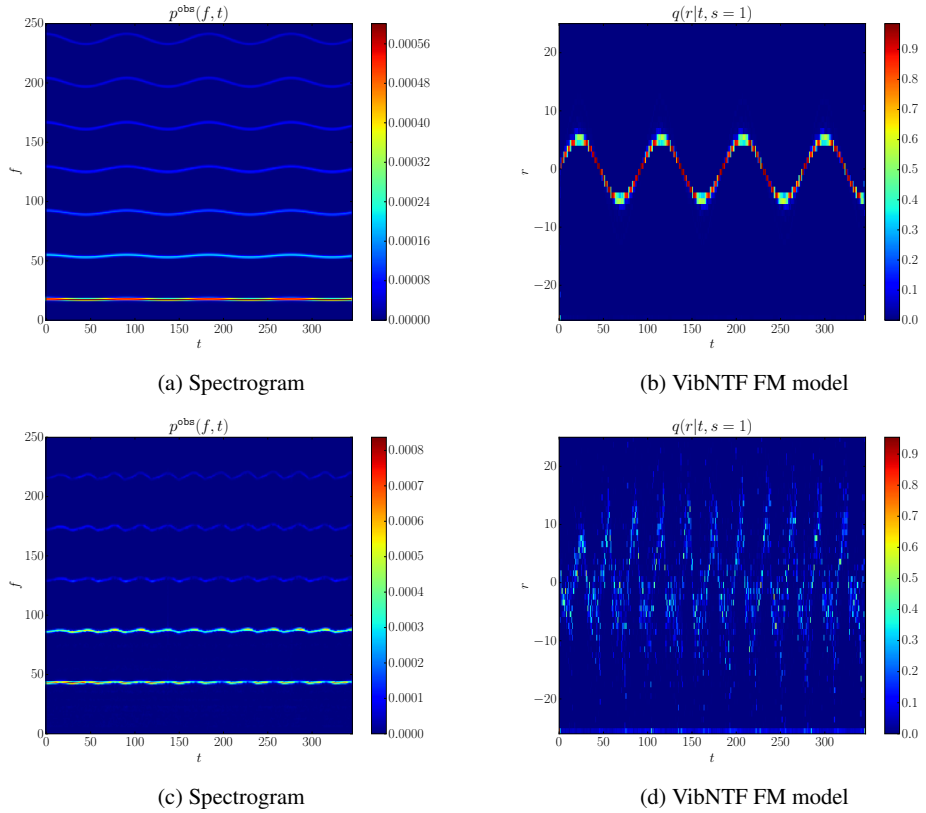
For notational convenience we define $\rho(f,t,r,z,s) \triangleq p^{\text{obs}}(f,t,r)q^{(i)}(z,s|f,t,r)$ and discarding the denominator in the log of (17) (constant w.r.t. q), equivalently write the optimization over the minorizing function as

$$\max_q \sum_{f,t,r,z,s} \rho(f,t,r,z,s)\log q(s)q(f|z,s)q(z,t|s)q(r|t,s). \tag{19}$$

[5] In turn, this would increase the dimensionality of the data.

[6] Cf. [20] for a more thorough treatment.

[7] Note that the MM iteration specifies an expectation-maximization.



(a) Spectrogram

(b) VibNTF FM model

(c) Spectrogram

(d) VibNTF FM model

Figure 3: For single-note analyses, VibNTF encodes the time-varying pitch modulation. The top row shows a synthetic vibrato square wave note (G5), while the bottom row shows a real recording of a violin vibrato note (B♭6). We plot r in the range $[-\frac{R}{2}, \frac{R}{2}]$ in figures 3(b) and 3(d) to clarify that the index r represents a zero-mean quantity (the FSFR).

We now alternatively update each factor by separating the argument in the log in (19) as a sum of logs, each term of which can be optimized by applying Gibb's inequality [13]. That is, given the current model, the optimal choice for some factor of $q^{(i+1)}$ is the marginal of ρ over the corresponding variables. E.g.,

$$q^{(i+1)}(s) \leftarrow \frac{\sum_{f,t,r,z} \rho(f,t,r,z,s)}{\sum_{f,t,r,z,s'} \rho(f,t,r,z,s')}. \tag{20a}$$

Likewise, the remaining factor updates are expressed as

$$q^{(i+1)}(f|z,s) \leftarrow \frac{\sum_{t,r} \rho(f,t,r,z,s)}{\sum_{f',t,r} \rho(f',t,r,z,s)}; \tag{20b}$$

$$q^{(i+1)}(z,t|s) \leftarrow \frac{\sum_{f,r} \rho(f,t,r,z,s)}{\sum_{f,t',r,z'} \rho(f,t',r,z',s)}; \tag{20c}$$

$$q^{(i+1)}(r|t,s) \leftarrow \frac{\sum_{f,z} \rho(f,t,r,z,s)}{\sum_{f,r',z} \rho(f,t,r',z,s)}. \tag{20d}$$

Since ρ is expressed as a product of the current factors and observed data, the factor updates can be implemented efficiently by using matrix multiplications to sum across inner dimensions as necessary. The theory guarantees convergence [8] to a local minimum [9], although in practice we stop the algorithm after some fixed number of iterations. The algorithm is initialized by choosing factors of $q^{(0)}$ as random valid conditional probabilities.

Figure 3 visualizes the FM factor $q(r|t,s)$ estimated by the proposed algorithm for single note analyses ($S = 1$) of both synthetic and real data.

4. EVALUATION

We present a comparison of our proposed method with the baseline KL-NMF (which our method extends) in a blind source separation task examining mixtures of two single-note recordings. We use the `BSS_EVAL` criteria [23] to evaluate separation performance, which necessitates the use of artificial mixtures. We report the source-to-distortion ratio (SDR), source-to-interference ratio (SIR), and source-to-artifact ratio (SAR), each in dB. Each experiment comprises 500 separations, with the sources in each trial chosen as specified below and mixed at 0 dB with a total mixture duration of 2 seconds at 44.1 kHz sampling rate. We report the average metrics across all sources and trials.

To use KL-NMF for blind source separation, we must specify $Z = 2$, i.e., each mixture component considered as a source. This baseline should be relatively easy to beat, since empirically KL-NMF does a poor job of modeling vibrato signals when Z is small.

[8] For guaranteed convergence, ρ must be recomputed after each factor update, rather than once per iteration as the notation suggests. However, in practice we observe convergence without the recomputation.

	BSS_EVAL in dB		
Algorithm	SDR	SIR	SAR
(A) Synthetic data			
2-part KL-NMF	-1.5 $\pm$ 0.1	0.1 $\pm$ 0.2	6.9 $\pm$ 0.2
Vibrato NTF	14.6 $\pm$ 1.0	17.0 $\pm$ 1.2	23.6 $\pm$ 0.7
(B) Real data			
2-part KL-NMF	2.8 $\pm$ 0.4	8.0 $\pm$ 2.1	9.2 $\pm$ 0.2
Vibrato NTF	5.8 $\pm$ 0.5	9.7 $\pm$ 2.2	17.7 $\pm$ 0.5

Table 1: Mean and 95% confidence intervals of the BSS_EVAL metrics for 500 unsupervised separations of two-source mixtures. Experiment A considers synthetic vibrato square waves, while experiment B considers single-note vibrato string instrument recordings.

For Vibrato NTF, we specify $S = 2$ and $Z = 3$, i.e., for each of the two sources we learn spectral templates and temporal activations for three components. E.g., considering a sinusoidal vibrato, the components could model the source during the crest, midpoint, and trough of the pitch modulation. We estimate the signal parameters at a particular (f_0, t_0) using DDM with a family of $L = 5$ analysis atoms (heterodyned Hann functions) in the same hop index and nearby frequency bins. In order to avoid the influence of noisy FSFR estimates in the factorization, we apply some mild post-processing prior to quantization. Specifically, we implicitly discard FSFR at (f, t) with $p^{\text{obs}}(f, t)$ below the 10^{th} percentile, or outside a reasonable range of ± 4 times the sampling rate by setting them to the data median. The FSFR are then quantized evenly across their range into $R = 50$ discrete values.

For both algorithms, the STFT in (1) is specified by a 1024-length (23 msec) Discrete Fourier Transform using a Hann window with 75% overlap between successive frames. Thus, $F = 513$, corresponding to the non-redundant frequency bins, and $T = 346$, the number of hops required to cover the mixture duration. Both algorithms are initialized randomly and run for 100 iterations.

Experiment A examines synthetic data, where the sources are square waves with frequency vibrato, whose signal parameters are generated at random. The fundamental frequency corresponds to a note value selected uniformly at random from the three-octave range [A3, G$\sharp$5]. The number of partials is chosen uniformly at random from the range [10, 30], and subsequently reduced as necessary to avoid aliasing. The vibrato modulation function, i.e., κ_s in (9), is a sinusoid with depth chosen uniformly at random in the range of [5%, 20%] of the fundamental and rate chosen log-uniformly at random from the range [0.5, 10] Hz.

Experiment B examines real data, where the sources are single-note recordings from the McGill University Master Samples (MUMS) [15], which contains over 6000 single-note and single-phrase recordings of classical and popular instruments. We focus our evaluation on string instruments, which exhibit strong frequency modulation in their vibrato effect [22]. The MUMS subset of string instrument notes with vibrato comprises a total of 250 unique recordings of violin, viola, cello, and double bass. The sources are chosen randomly from this subset and trimmed or padded to 2 seconds as necessary.

Results for both experiments are provided in table 1. Experiment A shows a dramatic win for Vibrato NTF over the baseline. We see some variability in the results, which reflects an optimization over a cost surface with many local optima. With random initialization, Vibrato NTF works either very well or very poorly, so robustness could be improved by a more careful initialization, or alternatively by regularizing the factorization in such a way as to avoid sub-optimal solutions.

In experiment B, we see that moving from synthetic to real data degrades the performance of our proposed method, although we still beat the baseline by a modest margin. Interestingly, the baseline performs better on real data than synthetic, likely because the pitch variations are less pronounced so KL-NMF fails less frequently. Moreover, the pitch modulations in real data are more complex than in the synthetic case (compare figures 3(b) and 3(d)), and may require more components (larger Z) to be properly modeled. Vibrato NTF as proposed tends to decrease in performance as Z increases, so additional work is required to improve robustness for the analysis of real data. We hypothesize that an extension enforcing temporal continuity in the FM factor, which should be smooth and monotonic per-source, would enhance the grouping of components, permitting a larger Z in practice.

5. CONCLUSION

We proposed Vibrato NTF, a novel blind source separation algorithm that extends NMF by leveraging local estimates of frequency modulation as grouping cues directly in the factorization. Experimental results using synthetic data showed a substantial improvement over the baseline, and validated the FSFR as useful grouping cues in a source separation task. In the experiment with real recordings, our method provided a more modest improvement. With regards to the analysis of real data, we believe the incorporation of sensible priors on the factors would improve the separation performance, while careful initalization would improve the robustness. Further work could include tailoring the proposed method to the analysis of polyphonic sounds, or sounds with mild or no frequency modulation. Additionally, an extension including coherent amplitude modulations as a grouping cue is possible within the proposed tensor factorization framework.

6. ACKNOWLEDGEMENTS

The research leading to this paper was partially supported by the French National Research Agency (ANR) as a part of the EDISON 3D project (ANR- 13-CORD-0008-02), and by the Canadian National Science and Engineering Research Council (NSERC). Additional support was provided by the Analog Garage, the emerging business accelerator at Analog Devices, Inc.

7. REFERENCES

[1] J. Allen and L. Rabiner. A unified approach to short-time Fourier analysis and synthesis. *Proceedings of the IEEE*, 65:1558–64, 1977.

[2] T. Barker and T. Virtanen. Non-negative tensor factorization of modulation spectrograms for monaural sound separation. In *Proceedings of the 2013 Interspeech Conference*, pages 827–31, Lyon, France, 2013.

[3] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in Bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):538–49, 2010.

[4] M. Betser. Sinusoidal polyphonic parameter estimation using the distribution derivative. *IEEE Transactions on Signal Processing*, 57(12):4633–45, 2009.

[5] A. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. The MIT Press, Cambridge, MA, 1990.

[6] J. M. Chowning. Computer synthesis of the singing voice. In *Sound Generation in Winds, Strings, Computers*, pages 4–13. Kungl. Musikaliska Akademien, Stokholm, Sweden, 1980.

[7] E. Creager. Musical source separation by coherent frequency modulation cues. Master's thesis, McGill University, 2015.

[8] B. Hamilton and P. Depalle. A unified view of non-stationary sinusoidal parameter estimation methods using signal derivatives. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 369–72, Kyoto, Japan, 2012.

[9] D. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 58(1):30–7, 2004.

[10] D. Lee, M. Hill, and H. Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*, 13:556–62, 2001.

[11] D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–91, 1999.

[12] Y. Li, J. Woodruff, and D.Wang. Monaural musical sound separation based on pitch and common amplitude modulation. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(7):1361–71, 2009.

[13] D. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2005.

[14] S. McAdams. Segregation of concurrent sounds I: Effects of frequency modulation coherence. *Journal of the Acoustic Society of America*, 86(6):2148–59, 1989.

[15] F. Opolko and J. Wapnick. McGill University master samples [Compact Disks], 1987.

[16] P. Smaragdis and J. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–80, New Paltz, NY, 2003.

[17] P. Smaragdis, C. Févotte, G. Mysore, N. Mohammadia, and M. Hoffman. Static and dynamic source separation using nonnegative factorizations: A unified view. *IEEE Signal Processing Magazine*, 31(3):66–74, 2014.

[18] P. Smaragdis, B. Raj, and M. Shashanka. A probabilistic latent variable model for acoustic modeling. In *Proceedings of the NIPS Workshop of Advances in Models for Acoustic Processing*, Vancouver, Canada, 2006.

[19] P. Smaragdis, B. Raj, and M. Shashanka. Supervised and semi-supervised separation of sounds single-channel mixtures. *Independent Component Analysis and Signal Separation*, (Lecture Notes in Computer Science, 4666):414–21, 2007.

[20] N. Stein. Nonnegative tensor factorization for directional unsupervised audio source separation. *arXiv preprint*, http://arxiv.org/abs/1411.5010, 2015.

[21] J. Traa, P. Smaragdis, N. Stein, and D. Wingate. Directional NMF for joint source localization and separation. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 2015.

[22] V. Verfaille, C. Guastavino, and P. Depalle. Perceptual evaluation of vibrato models. In *Proceedings of the Conference on Interdisciplinary Musicology*, Montreal, Canada, 2005.

[23] E. Vincent, R. Gribonval, and C. Févotte. Performance measurements in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462–9, 2006.

[24] T. Virtanen. Monaural sound source separation by non-negative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3):1066–74, 2007.

[25] A. Wang. Instantaneous and frequency-warped techniques for source separation and signal parameterization. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 47–50, New Paltz, NY, 1995.

[26] D. Wang and G. Brown. *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. Wiley Interscience, Hoboken, NJ, 2006.

ON DRUM PLAYING TECHNIQUE DETECTION IN POLYPHONIC MIXTURES

Chih-Wei Wu, Alexander Lerch
Georgia Institute of Technology, Center for Music Technology
{cwu307, alexander.lerch}@gatech.edu

ABSTRACT

In this paper, the problem of drum playing technique detection in polyphonic mixtures of music is addressed. We focus on the identification of 4 rudimentary techniques: strike, buzz roll, flam, and drag. The specifics and the challenges of this task are being discussed, and different sets of features are compared, including various features extracted from NMF-based activation functions, as well as baseline spectral features. We investigate the capabilities and limitations of the presented system in the case of real-world recordings and polyphonic mixtures. To design and evaluate the system, two datasets are introduced: a training dataset generated from individual drum hits, and additional annotations of the well-known ENST drum dataset minus one subset as test dataset. The results demonstrate issues with the traditionally used spectral features, and indicate the potential of using NMF activation functions for playing technique detection, however, the performance of polyphonic music still leaves room for future improvement.

1. INTRODUCTION

Automatic Music Transcription (AMT), one of the most popular research topics in the Music Information Retrieval (MIR) community, is the process of transcribing the musical events in the audio signal into a notation such as MIDI or sheet music. In spite of being intensively studied, there still remain many unsolved problems and challenges in AMT [1]. One of the challenges is the extraction of additional information, such as dynamics, expressive notation and articulation, in order to produce a more complete description of the music performance.

For pitched instruments, most of the work in AMT mainly focuses on tasks such as melody extraction [3], chord estimation [10], and instrument recognition [8]. Few studies try to expand the scope to playing technique and expression detection for instruments such as electric guitar [5, 17] and violin [12]. Similarly, the main focus of AMT systems for percussive instruments has been put on recognizing the instrument types (e.g., HiHat (HH), Snare Drum (SD), Bass Drum (BD)) and their corresponding onset times [2, 6, 14, 18, 20]. Studies on retrieving the playing techniques and expressions are relatively sparse.

Since playing technique is an important layer of a musical performance for its deep connection to the timbre and subtle expressions of an instrument, an automatic system that transcribes such techniques may provide insights into the performance and facilitate other research in MIR. In this paper, we present a system that aims to detect the drum playing techniques within polyphonic mixtures of music. The contributions of this paper can be summarized as follows: first, to the best of our knowledge, this is the first study to investigate the automatic detection of drum playing techniques in polyphonic mixtures of music. The results may support the future development of a complete drum transcription system. Second, a comparison between the commonly used timbre features and features based on activation functions of a Non-Negative Matrix Factorization (NMF) system are presented and discussed. The results reveal problems with using established timbre features. Third, two datasets for training and testing are introduced. The release of these datasets is intended to encourage future research in this field. The data may also be seen as a core compilation to be extended in the future.

The remainder of the paper is structured as follows: in Sect. 2, related work in drum playing technique detection is introduced. The details of the proposed system and the extracted features are described in Sect. 3, and the evaluation process, metrics, and the experiment results are shown in Sect. 4. Finally, the conclusion and future research directions are addressed in Sect. 5.

2. RELATED WORK

Percussive instruments, generating sounds through vibrations induced by strikes and other excitations, are among the oldest musical instruments [15]. While the basic gesture is generally simple, the generated sounds can be complex depending on where and how the instrument is being excited. In western popular music, a drum set, which contains multiple instruments such as SD, BD, HH, is one of the most commonly used percussion instruments. In general, every instrument in a drum set is excited using drum sticks. With good control of the drum sticks, variations in timbre can be created through different excitation methods and gestures [16]. These gestures, referred to as rudiments, are the foundations of many drum playing techniques. These

© Chih-Wei Wu, Alexander Lerch. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Chih-Wei Wu, Alexander Lerch. "On drum playing technique detection in polyphonic mixtures", 17th International Society for Music Information Retrieval Conference, 2016.

rudiments can be categorized into four types: [1]

1. Roll Rudiments: drum rolls created by single or multiple bounce strokes (Buzz Roll).
2. Paradiddle Rudiments: a mixture of alternative single and double strokes.
3. Flam Rudiments: drum hits with one preceding grace note.
4. Drag Rudiments: drum hits with two preceding grace notes created by double stroke.

There are also other playing techniques that are commonly used to create timbral variations in a drum set, such as *Brush*, *Cross Stick*, *Rim Shot*, etc. Most drum transcription systems, however, focus on single strikes instead of these playing techniques [2, 6, 14, 18, 20].

In an early attempt to retrieve percussion gestures from the audio signal, Tindale et al. investigated the timbral variations of the snare drum sounds induced by different excitations [19]. Three expert players were asked to play on different locations on the snare drums (center, halfway, edge, etc.) with different excitations (strike, rim shot, and brush), resulting in a dataset with 1260 individual samples. The classification results for this dataset based on standard spectral and temporal features (e.g., centroid, flux, MFCCs, etc.) and classifiers (e.g., k-Nearest Neighbors (k-NN) and Support Vector Machine (SVM)) were reported, and an overall accuracy of around 90% was achieved. Since the dataset is relatively small, howver, it is difficult to generalize the results to different scenarios.

Following the same direction, Prockup et al. further explored the discrepancy between more expressive gestures with a larger dataset that covers multiple drums of a standard drum set [13]. A dataset was created with combinations of different drums, stick heights, stroke intensities, strike positions and articulations. Using a machine learning based approach similar to [19], various features were extracted from the samples, and a SVM was trained to classify the sounds. An accuracy of over 95% was reported on multiple drums with a 4-class SVM and features such as MFCCs, Spectral features, and the proposed custom-designed features.

Both of the above mentioned studies showed promising results in classifying the isolated sounds, however, they were not evaluated with real-world drum recordings, and the applicability of these approaches for transcribing real-world drum recordings still needs to be tested. Additionally, the potential impact of polyphonic background music could be another concern with respect to these approaches.

Another way to retrieve more information from the drum performance is through the use of multi-modal data [9]. Hochenbaum and Kapur investigated the inclusion of drum hand recognition in the data by capturing microphone and accelerometer data simultaneously. Two performers were asked to play the snare drum with four different rudiments (namely single stroke roll, double stroke open roll, single paradiddle and double paradiddle). Standard spectral and temporal features (e.g., centroid, skewness, zero-crossing rate, etc.) were extracted from the audio and accelerometer data, and different classifiers were applied and compared. With a Multi-Layer Perceptron (MLP), an accuracy of around 84% was achieved for a 2-class drum hand classification task. It cannot be ruled out that the extra requirement of attaching the sensors to the performers' hands might alter the playing experience and result in deviations from the real playing gestures. Furthermore, this method does not allow the analysis of existing audio recordings.

[1] http://vicfirth.com/40-essential-rudiments/ Last Access: 2016/3/16

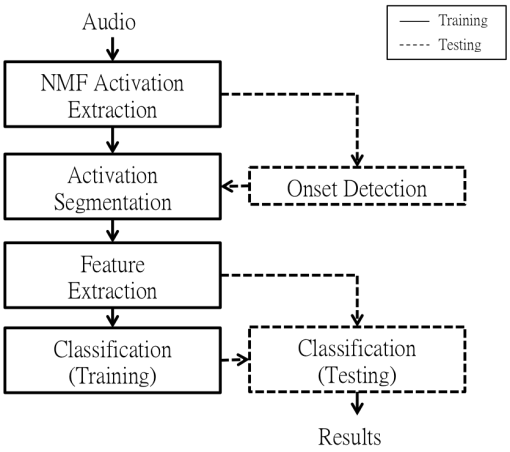


Figure 1. Block diagram of the proposed system (onset detection is bypassed in the current experiments)

In general, the above mentioned studies mainly focus on evaluating the discriminability of isolated samples. The evaluation on real-world drum recordings, i.e., recordings of a drummer continuously playing, is usually unavailable due to the lack of annotated datasets. In Table 1, different datasets for drum transcription are presented. It can be found that most of the datasets only contain annotations of playing techniques that are easily distinguishable from the normal strike (e.g., Cross Stick, Brush, Rim Shot). For playing techniques such as Flam, Drag and Buzz Roll, there are no datasets and annotations available.

3. METHOD

3.1 System Overview

The block diagram of the proposed system is shown in Figure 1. The system consists of two stages: training and testing. During the training stage, NMF activation functions (see Sect. 3.2.1) will first be extracted from the training data. Here, the training data only consists of audio clips with one-shot samples of different playing techniques. Next, features will be extracted from a short segment around the salient peak in the activation function (see Sect. 3.2.1). Finally, all of the features and their corresponding labels will be used to train a classifier. The classes we focus on in this paper are: Strike, Buzz Roll, Drag, Flam.

For the testing, a similar procedure is performed. When a longer drum recording is used as the testing data, an

Dataset	Annotated Techniques	Description	Total
Data in [15]	Strike, Rim Shot, Brush	1 drum (snare), 5 strike positions (from center to edge)	1264 clips
MDLib2.2 [16]	Strike, Rim Shot, Buzz Roll, Cross Stick	9 drums, 4 stick heights, 3 stroke intensities, 3 strike positions	10624 clips
IDMT-Drum [9]	Strike	3 drums (snare, bass and hihat), 3 drum kits (real, waveDrum, technoDrum)	560 clips
ENST Drum Minus One Subset [18]	Strike, Rim Shot, Brush, Cross Stick	13 drums, 3 drum kits played by 3 drummers	64 tracks

Table 1. An overview of publicly available datasets for drum transcription tasks

additional onset detection step is taken to narrow down the area of interest. Since the focus of this paper is on playing technique detection, the onset detection step is bypassed by adopting the annotated ground truth in order to simulate the best case scenario. Once the features have been extracted from the segments, the pre-trained classifier can be used to classify the playing technique in the recordings. More details will be given in the following sections.

3.2 Feature Extraction

3.2.1 Activation Functions (AF)

To detect drum playing technique in polyphonic music, a transcription method that is robust against the influence of background music is required. In this paper, we applied the drum transcription scheme as described in [20] for its adaptability to polyphonic mixtures of music. The flowchart of the process is shown in Fig. 2. This method decomposes the magnitude spectrogram of the complex mixtures with a fixed pre-trained drum dictionary and a randomly initialized dictionary for harmonic contents. Once the signal is decomposed, the activation function $h_i(n)$ of each individual drum can be extracted, in which n is the block index and $i = \{HH, SD, BD\}$ indicates the type of drum

All of the audio samples are mono with a sampling rate of 44.1 kHz. The Short Time Fourier Transform (STFT) of is computed with a block size of 512 and a hop size of 128, and a Hann window is applied to each block. The harmonic rank r_h for the partially-fixed NMF is 50, and the drum dictionary is trained from the ENST drum dataset [7] with a total number of three templates (one template per drum). The resulting $h_i(n)$ is scaled to a range between 0 and 1 and smoothed using a median filter with an order of $p = 5$ samples. Since a template in the dictionary is intended to capture the activity of the same type of drum, the drum sounds with slightly different timbres will still result in similar $h_i(n)$. Therefore, the extracted activation function $h_i(n)$ can be considered as a timbre invariant transformation and is desirable for detecting the underlying techniques. Segments of these activation functions can be used directly as features or as the intermediate representation for the extraction of other features.

3.2.2 Activation Derived Features (ADF)

Once the activation functions $h_i(n)$ have been extracted from the audio data, various features can be derived for subsequent classification. The steps can be summarized as

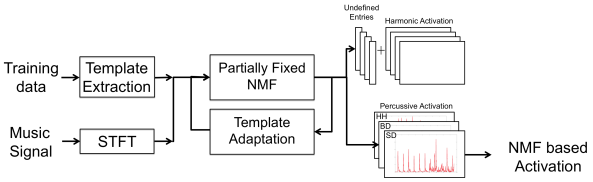


Figure 2. Flowchart of the activation extraction process, see [20]

follows: first, for every given onset at index n_o, a 400 ms segment centered around $h_i(n_o)$ will be selected. Next, the segment is shifted to ensure the maximum value is positioned at the center. From this segment, we extract the distribution features, the Inter-Onset Interval (IOI) features, the peak features, and the Dynamic Time Warping (DTW) features as described below:

1. **Distribution features,** $d = 5$: Spread, Skew, Crest, Centroid, and Flatness. These features are similar to the commonly used spectral features, which provide the general description of the pattern.

2. **IOI features,** $d = 2$: IOI mean, and IOI standard deviation. These features are simple statistics of the IOIs.

3. **Peak features,** $d = 8$: side peak to main peak ratio α_i, and side peak to main peak signed block index difference Δb_i $i = \{1, 2, 3, 4\}$. These features are designed to describe the details of the patterns. To compute the peak features, first we find the local maxima and sort them in descending order, then we calculate the ratio and index difference between the side peak and the main (largest) peak as features.

4. **DTW features,** $d = 4$: the cumulative cost of a DTW distance between the current and the 4 template activation functions. To compute the DTW features, a median activation template of each playing technique is trained from the training data, and the cumulative cost of every DTW template for the given segment can be calculated. The examples of the extracted DTW templates for each technique are shown in Fig. 3.

The resulting feature vector has a dimension $d = 19$.

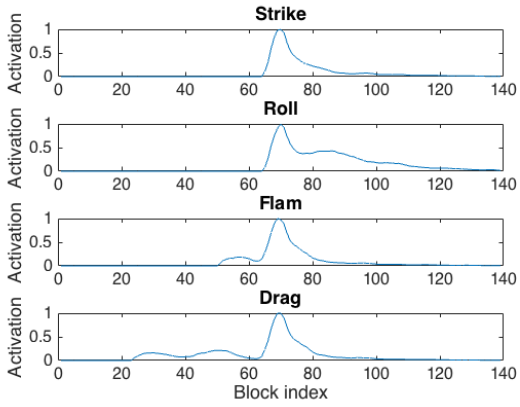


Figure 3. Examples of the extracted and normalized activation functions of (top to bottom): Strike, Buzz Roll, Flam, Drag

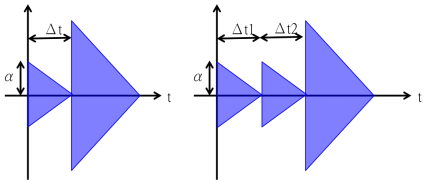


Figure 4. Illustration of the parametric forms of (Left) Flam and (Right) Drag

3.2.3 Timbre Features (TF)

To compare the effectiveness of the activation based features, a small set of the commonly used timbre features as described in [11] is extracted as well. The extraction process is similar to Sect. 3.2.2, however, instead of using activation functions, the waveform of a given segment is used to derive the features. The features are:

1. **Spectral features,** $d = 3$: Centroid, Rolloff, Flux
2. **Temporal features,** $d = 1$: Zero crossing rate
3. **MFCCs,** $d = 13$: the first 13 MFCC coefficients

These features are computed block by block using the same parameters as described in Sect. 3.2.1. The resulting feature vectors are further aggregated into one single vector per segment by computing the mean and standard deviation of all the blocks. The final feature vector has a dimension $d = 34$.

3.3 Dataset

3.3.1 Training Dataset

In this paper, we focus on four different playing techniques (Strike, Flam, Drag, Buzz Roll) played on the snare drum. As can be seen in Table 1, only Strike and Buzz Roll can be found in some of these datasets. Therefore, we generated a dataset through mixing existing recordings from MDLib 2.2 [13]. Since both Flam and Drag consist of preceding grace notes with different velocity and timing, they can be modeled with a limited set of parameters as shown in Fig. 4. The triangles in the figure represent the basic waveform excited by normal strikes, and the Δt is the time difference between neighboring excitations. All the waveforms have been normalized to a maximum amplitude of -1 to 1, and the α is the amplitude ratio between the grace note and the strong note.

Techniques	Description	Total (#clips)
Strike	Snare excerpts from MDLib 2.2 [16]	576
Buzz Roll	Snare excerpts from MDLib 2.2 [16]	576
Flam	144 mono snare excerpts α = {0.1:0.1:0.7} Δt = {30:10:60} (ms)	4032
Drag	144 mono snare excerpts α = {0.15:0.1:0.55} Δt_1 = {50:10:70} (ms) Δt_2 = {45:10:75} (ms)	8640

Table 2. An overview of the constructed dataset

In order to have realistic parameter settings for Δt and α, we annotated demo videos from Vic Firth's online lessons for both Flam [2] and Drag. [3] The final parameter settings and the details of the constructed dataset are shown in Table 2. The parameters are based on the mean and standard deviation estimated from the videos. The resulting data contains all possible combinations of the parameters with the 144 mono snare Strike in the MDLib 2.2. However, to ensure the classifier is trained with uniformly distributed classes, only 576 randomly selected clips are used for Flam and Drag during the training.

3.3.2 Test Dataset

To evaluate the system for detecting the playing techniques in polyphonic mixtures of music, the tracks from the ENST drum dataset minus one subset [7] have been annotated. The ENST drum dataset contains various drum recordings from 3 drummers with 3 different drum kits. The minus one subset, specifically, consists of 64 tracks of drum recordings with individual channel, mix, and accompaniments available. Since the playing technique is related to the playing style of the drummer, only 30 out of 64 tracks contain such techniques on snare drum. These techniques are annotated using the snare channel of the recordings, and each technique is labeled with the starting time, duration, and the technique index. As a result, a total number of 182 events (Roll: 109, Flam: 26, Drag: 47) have been annotated, and each event has a length of approximately 250 to 400 ms. All of the above mentioned annotations are available online. [4]

4. EVALUATION

4.1 Metrics

For evaluating the accuracy on the testing data, we calculate the *micro-averaged accuracy* and the *macro-averaged accu-*

[2] http://vicfirth.com/20-flam/ Last Access: 2016/03/16
[3] http://vicfirth.com/31-drag/ Last Access: 2016/03/16
[4] https://github.com/cwu307/DrumPtDataset

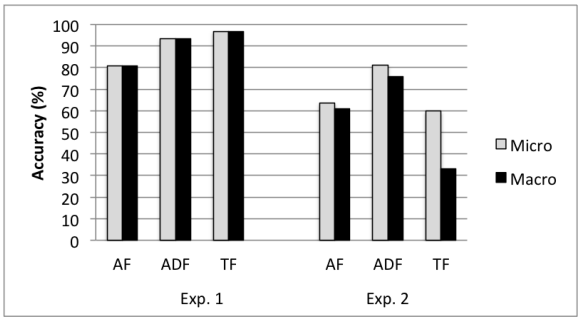


Figure 5. Results of experiment 1 (left) and experiment 2 (right)

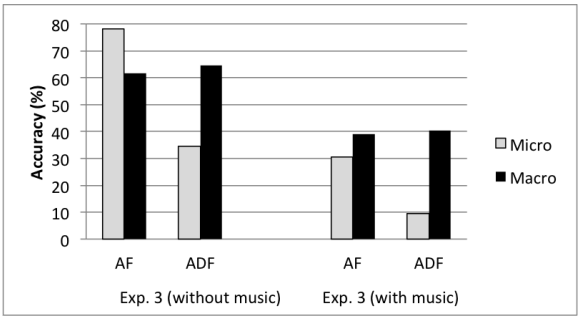


Figure 6. Results of experiment 3 without background music (left) and with background music (right)

racy [21] to account for the unevenly distributed and sparse classes. The metrics are defined in the following equations:

$$micro\ averaged = \frac{\sum_{k=1}^{K} C_k}{\sum_{k=1}^{K} N_k} \tag{1}$$

$$macro\ averaged = \frac{1}{K} \sum_{k=1}^{K} \left(\frac{C_k}{N_k} \right) \tag{2}$$

in which K is the total number of classes, N_k is the total number of samples in class k, and C_k is the total number of correct samples in class k. These two metrics have different meanings: while each sample is weighted equally for the micro-averaged accuracy, the macro-averaged accuracy applies equal weight to each class, which gives a better overview of the performance by emphasizing the minority classes.

4.2 Experiment Setup

In this paper, three sets of experiments are conducted. The first experiment consists of running a 10-fold cross-validation on the training data, in the second experiment the test data is classified with an annotation-informed segmentation, and the third experiment classifies the test data without the annotation-informed segmentation. Different feature sets as described in Sect. 3.2, namely AF, ADF, and TF, are tested using a multi-class C-SVM with Radial Basis Function (RBF) kernel. For the implementation, we used *libsvm* [4] in Matlab. All of the features are scaled to a range between 0 and 1 using the standard min-max scaling approach.

4.3 Results

4.3.1 Experiment 1: Cross-Validation on Training Data

In Experiment 1, a 10-fold cross validation on the training data using different sets of features is performed. The results are shown in Fig. 5 (left). This experimental setup is chosen for its similarity to the approaches described in previous work [13, 19]. As expected, the features allow to reliably separate the classes with accuracies between 80.9–96.8% for the different feature sets. Since the training data contains 576 samples for all classes, the micro-averaged and marco-averaged accuracy are the same.

4.3.2 Experiment 2: Annotation-Informed Testing

In Experiment 2, the same sets of features are extracted from the testing data for evaluation. Since the testing data is a completely different dataset with the real-world drum recordings, a verification of the feasibility of using the synthetic training data as well as the proposed feature representations is necessary. For this purpose, we simulate the best case scenario by using the snare channel as the input with an annotation-informed process for isolating the playing techniques. The resulting 182 segments are then classified using the trained SVM models from Experiment 1. With ADF, the best performance of 76.0 and 81.3% was achieved for macro and micro-averaged accuracy, respectively. This experiment serves as a sanity check to the presented scheme. Note that strikes are excluded in this experiment, therefore, the micro-averaged accuracy mainly reflects the accuracy of the majority class, which is Roll.

4.3.3 Experiment 3: Real-World Testing

Experiment 3 utilizes a more realistic setup and is our main experiment. Each onset is examined and classified without any prior knowledge about the segmentation. A fixed region around each onset is segmented and classified. As a result, a total number of 2943 onsets (including the previous mentioned 182 playing technique events and 2761 strikes) are evaluated. Since the timbre features do not show promising results in Experiment 2, they are excluded from this experiment. To investigate the influence of the background music, both the recordings of the snare channel and the complete polyphonic mixtures are tested. The results are shown in Fig. 6. Without the background music, the best macro-averaged accuracy is 64.6% using ADF, and the best micro-averaged accuracy is 78.0% using AF. With the background music, the best macro-averaged accuracy is 40.4% using ADF, and the best micro-averaged accuracy is 30.4% using AF.

4.4 Discussion

Based on the experiment results, the following observations can be made:

First, as can be seen in Fig. 5, the timbre features achieve the highest cross-validation accuracy in Experiment 1, which shows their effectiveness in differentiating

	Strike	Roll	Flam	Drag
Strike (2761)	28.9	38.8	5.7	26.6
Roll (109)	8.3	66.1	11.9	13.8
Flam (26)	3.8	53.8	19.2	23.1
Drag (47)	46.8	6.4	4.3	42.6

Table 3. Confusion matrix of Exp. 3 with music and AF (in %)

	Strike	Roll	Flam	Drag
Strike (2761)	5.8	62.9	3.8	27.6
Roll (109)	5.5	74.3	3.7	16.5
Flam (26)	0.0	61.5	11.5	26.9
Drag (47)	2.1	8.5	19.1	70.2

Table 4. Confusion matrix of Exp. 3 with music and ADF (in %)

our classes. This observation echos the results from the related work, which demonstrate the usefulness of timbre features for distinguishing the different sounds. However, when these features are applied to classify a completely different dataset, they are unable to recognize the same pattern played with different drum sounds. As a result, the timbre features achieve the lowest macro-averaged accuracy in Experiment 2, and the micro-averaged accuracy approximates the Zero-R accuracy by always predicting the majority class. This result shows that timbre features might not be directly applicable to detecting the playing techniques in unknown recordings. The activation functions and activation derived features, on the other hand, are relatively stable and consistent between the micro and macro-averaged accuracy. This indicates a better performance for detecting the proposed playing techniques in the unseen dataset.

Second, in comparison with the AF, the ADF tends to achieve a higher macro-averaged accuracy than the activation functions among all experiment results. Furthermore, the ADF is more sensitive to different playing techniques, whereas the AF is more sensitive to strikes. These results indicate that the ADF is more capable of detecting the playing techniques. This tendency can also be seen in the confusion matrices in Tables 3 and 4, where the ADF performs better than the AF in Roll and Drag, and slightly worse in Flam. The AF generally achieves higher micro-averaged accuracy than the ADF. Since the distribution of the classes is skewed towards Strike in the testing data, the micro-averaged accuracy of the AF is largely increased by a higher rate of detecting strikes.

Third, according to the confusion matrices (Tables 3 and 4), Strike and Flam can be easily confused with Roll for both features in the context of polyphonic mixtures of music. One possible explanation is that, whenever the signal is not properly segmented, the activation function will contain overlapping activities from the previous or the next onset, which might result in leakage to the original activation and make it resemble a Roll. The strong skewness towards the preceding grace notes in the case of Drag makes it relatively easy to distinguish from Roll for both features.

Fourth, for both activation functions and activation derived features, the detection performance drops drastically in Experiment 3 with the presence of background music. The reason could be that with the background music, the extracted activation function becomes noisier due to the imperfect decomposition. Since the classification models are trained on the clean signals, they might be susceptible to these disturbance. As a result, the classifier might be tricked into classifying Strike as other playing techniques, decreasing the micro-averaged accuracy.

Note that the proposed method does not take into account the onset detection at this moment. By adding the onset detection process, the detection accuracy will be further reduced, which decreases the reliability of the approach.

5. CONCLUSION

In this paper, a system for drum playing technique detection in polyphonic mixtures of music has been presented. To achieve this goal, two datasets have been generated for training and testing purposes. The experiment results indicate that the current method is able to detect the playing techniques from real-world drum recordings when the signal is relatively clean. However, low accuracy of the system in the presence of background music indicates that more sophisticated approaches should be applied in order to improve the detection of playing techniques in polyphonic mixtures of music.

Possible directions for the future work are: first, investigate different source separation algorithms as a preprocessing step in order to get a cleaner representation. The results of Experiments 2 and 3 show that a cleaner input representation improves both the micro and macro-averaged accuracy by more than 20%. Therefore, a good source separation method to isolate the snare drum sound could be beneficial. Common techniques such as HPSS and and other approaches for source separation should be investigated.

Second, since the results in Experiment 3 implies that the system is susceptible to the disturbance from background music, a classification model trained on the slightly noisier data could expose the system to more variations of the activation funcitons and possibly increase the robustness against the presence of unwanted sounds. The influence of adding different levels of random noise while training could be evaluated.

Third, the current dataset offers only a limited number of samples for the evaluation of playing technique detection in polyphonic mixtures of music. Due to the sparse nature of these playing techniques, their occurrence in existing datasets is rare, making the annotation difficult. However, to arrive at a statistically more meaningful conclusion, additional data would be necessary.

Last but not least, different state-of-the-art classification methods, such as deep neural networks, could also be applied to this task in searching for a better solution.

6. REFERENCES

[1] Emmanouil Benetos, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, jul 2013.

[2] Emmanouil Benetos, Sebastian Ewert, and Tillman Weyde. Automatic transcription of pitched and unpitched sounds from polyphonic music. In *Proceedings of the International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2014.

[3] Rachel M. Bittner, Justin Salamon, Slim Essid, and Juan P. Bello. Melody extraction by contour classification. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 500–506, 2015.

[4] Chih-Chung Chang and Chih-Jen Lin. LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011.

[5] Yuan-ping Chen, Li Su, and Yi-hsuan Yang. Electric Guitar Playing Technique Detection in Real-World Recordings Based on F0 Sequence Pattern Recognition. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 708–714, 2015.

[6] Christian Dittmar and Daniel Gärtner. Real-time Transcription and Separation of Drum Recording Based on NMF Decomposition. In *Proceedings of the International Conference on Digital Audio Effects (DAFX)*, pages 1–8, 2014.

[7] Olivier Gillet and Gaël Richard. Enst-drums: an extensive audio-visual database for drum signals processing. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2006.

[8] Philippe Hamel, Sean Wood, and Douglas Eck. Automatic Identification of Instrument Classes in Polyphonic and Poly-Instrument Audio. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 399–404, 2009.

[9] Jordan Hochenbaum and A Kapur. Drum Stroke Computing: Multimodal Signal Processing for Drum Stroke Identification and Performance Metrics. *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, 2011.

[10] Eric J Humphrey and Juan P Bello. Four timely insights on automatic chord estimation. *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2015.

[11] Alexander Lerch. *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*. John Wiley & Sons, 2012.

[12] Pei-Ching Li, Li Su, Yi-hsuan Yang, and Alvin W. Y. Su. Analysis of Expressive Musical Terms in Violin using Score-Informed and Expression-Based Audio Features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2015.

[13] Matthew Prockup, Erik M. Schmidt, Jeffrey Scott, and Youngmoo E. Kim. Toward Understanding Expressive Percussion Through Content Based Analysis. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2013.

[14] Axel Roebel, Jordi Pons, Marco Liuni, and Mathieu Lagrange. On Automatic Drum Transcription Using Non-Negative Matrix Deconvolution and Itakura Saito Divergence. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015.

[15] Thomas D Rossing. Acoustics of percussion instruments: Recent progress. *Acoustical Science and Technology*, 22(3):177–188, 2001.

[16] George Lawrence Stone. *Stick Control: for the Snare Drummer*. Alfred Music, 2009.

[17] Li Su, Li-Fan Yu, and Yi-Hsuan Yang. Sparse ceptral and phase codes for guitar playing technique classification. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, number Ismir, pages 9–14, 2014.

[18] Lucas Thompson, Matthias Mauch, and Simon Dixon. Drum Transcription via Classification of Bar-Level Rhythmic Patterns. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2014.

[19] Adam R. Tindale, Ajay Kapur, George Tzanetakis, and Ichiro Fujinaga. Retrieval of percussion gestures using timbre classification techniques. In *in Proceedings of the International Conference on Music Information Retrieval*, pages 541–544, 2004.

[20] Chih-Wei Wu and Alexander Lerch. Drum Transcription Using Partially Fixed Non-Negative Matrix Factorization with Template Adaptation. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 257–263, 2015.

[21] Yiming Yang. An Evaluation of Statistical Approaches to Text Categorization. *Journal of Information Retrieval*, 1:69–90, 1999.

PREDICTING MISSING MUSIC COMPONENTS WITH BIDIRECTIONAL LONG SHORT-TERM MEMORY NEURAL NETWORKS

I-Ting Liu
Carnegie Mellon University
School of Music
itingl@andrew.cmu.edu

Richard Randall
Carnegie Mellon University
School of Music
Center for the Neural Basis of Cognition
randall@cmu.edu

ABSTRACT

Successfully predicting missing components (entire parts or voices) from complex multipart musical textures has attracted researchers of music information retrieval and music theory. However, these applications were limited to either two-part melody and accompaniment (MA) textures or four-part Soprano-Alto-Tenor-Bass (SATB) textures. This paper proposes a robust framework applicable to both textures using a Bidirectional Long-Short Term Memory (BLSTM) recurrent neural network. The BLSTM system was evaluated using frame-wise accuracies on the Nottingham Folk Song dataset and J. S. Bach Chorales. Experimental results demonstrated that adding bidirectional links to the neural network improves prediction accuracy by 3% on average. Specifically, BLSTM outperforms other neural-network based methods by 4.6% on average for four-part SATB and two-part MA textures (employing a transition matrix). The high accuracies obtained with BLSTM on both two-part and four-part textures indicated that BLSTM is the most robust and applicable structure for predicting missing components from multi-part musical textures.

1. INTRODUCTION

This paper presents a method for predicting missing components from complex multipart musical textures. Specifically, we examine two-part melody and accompaniment (MA) and Soprano-Alto-Tenor-Bass (SATB) chorale textures. We treat each voice as a part (e.g. the melody of the MA texture or the Soprano of the SATB texture) and the problem we address is given an incomplete texture, how successfully can we generate the missing part. This project proposes a robust approach that is capable of handling both textures elegantly and has applications to any style of music. Predictions are made using a Bidirectional Long-Short Term Memory (BLSTM) recurrent neural network that is able to learn the relationship between components, and can thus be trained to predict missing components. This work demonstrates the capability of the BLSTM system by conducting experiments on the two tasks mentioned above with two distinct datasets.

Analyzing music with the aid of computer programs has attracted researchers of music information retrieval and music theory over the past twenty years. Music (especially western tonal music) has always been regarded as a kind of art with rigorous formalization. Various complex rules regulate how notes can be and cannot be played together in complex multipart textures. Such rules change over time and are subject to multiple factors. As artificial intelligence and machine-learning research advances, it is natural that computer scientists apply such technique to music analysis in order to elucidate these rules [2]. Two popular tasks investigated in this area are (1) generating chord accompaniments for a given melody in a two-part MA texture and (2) generating a missing voice for an incomplete four-part SATB texture. Successfully accomplishing either task manually is time-consuming and requires considerable style-specific knowledge and the applications discussed below are designed to automate and help non-professional musicians compose and analyze music.

Approaches that treat these problems can be categorized into two types according to the level of human engagement in discovering and applying music rules. Early works that handle incomplete four-part SATB textures were mostly knowledge-based models. Steels [28] proposed a representation system to encode musical information and exploited heuristic search, which takes the form of if-then musical rules that specifies solutions under different conditions to generate voices. Ebcioğlu built CHORAL, a knowledge-based system that includes over 350 rules modeling the style of Johann Sebastian Bach [8]. Due to the large number of rules involved, some studies modeled the problem as a constraint satisfaction problem, as was used by Pachet and Roy [22] on four-part textures and Ramirez, et al. [25] on two-part textures. Knowledge-based genetic algorithms were also used as an alternative method to represent the rules. Mcintyre [21] implemented a system that harmonizes user-defined melody in Baroque style, and Hall [17] presented a system that selects combination of attributes to model the harmonization of J. S. Bach's chorales. Freitas and Guimaraes also implemented a system based on genetic algorithms in [11]. The fitness function and genetic

© I-Ting Liu, Richard Randall. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** I-Ting Liu, Richard Randall. "Predicting Missing Music Components with Bidirectional Long Short-Term Memory Neural Networks", 17th International Society for Music Information Retrieval Conference, 2016.

operators rely on "music knowledges" to suggest chord progressions for given melodies.

While rules in knowledge-based systems have to be manually encoded into these systems, rules in probabilistic models and neural networks can be derived by training corpora without human intervention by the models. Hidden Markov Models (HMM) are one of the most common probabilistic models for the task of generating a chord sequence given melodies for two-part textures [27]. In HMM, a pre-selected dataset is used to train a transition probability matrix, which represents the probability of changing from one chord to another, and a melody observation matrix, the probability of encountering each note when different chords are being played. The optimal chord sequence is then generated using dynamic programming, or Viterbi Algorithm. HMM are also used by Allan [1] to harmonize four-part chorales in the style of J. S. Bach. In addition to HMM, Markov Model and Bayesian Networks are alternative models used for four-part textures by Biyikoglu [3] and Suzuki, et al. [29]. Raczyński, et al. [24] proposed a statistical model that combines multiple simple sub-models. Each sub-model captures different music aspects such as metric and pitch information, and all of them are then interpolated into a single model. Paiement, et al. [23] proposed a multi-level graphical model, which is proved to capture the long-term dependency among chord progression better than traditional HMM. One drawback of probabilistic models is that they cannot correctly handle data that are not seen in training data. Chuan and Chew [5] reduced this problem by using a hybrid system for style-specific chord sequence generation with statistical learning approach and music theory.

Neural networks have also been used by some researchers. Gang, et al. [13] were one of the earliest that used neural networks to produce chord harmonization for given melodies. Jordan's sequential neural network consisted of a sub-net that learned to identify chord notes for the melody in each measure, and the result was fed into the network to learn the relationship between melodies and chords. The network was later adopted real-time application [12, 14]. Consisting of 3 layers, the input layer takes pitch, metric information and the current chord context, and the output layer predicts the next chord. Cunha, et al. [6] also proposed a real-time chord harmonization system using multi-layer perceptron (MLP) neural network and a rule-based sequence tracker that analyzes the structure of the song in real-time, which provides additional information on the context of the notes being played.

Hoover, et al. [20] used two Artificial Neural Networks (ANN) to model the relationship between melodies and accompaniment as a function of time. The system was later extended to generate multi-voice accompaniment by increasing the size of the output layer in [19]. Bellgard and Tsand [2] trained an effective Bolzmann machine and incorporated external constraints so that harmonization follows the rules of a chorale. Fuelner developed a feed-forward neural network that harmonizes melodies in specific styles in [9]. De Prisco, et al. [7] proposed a neural network that finds appropriate chords to harmonize given bass lines in four-part SATB chorales by combining three base networks, each of which models context of different time lengths.

Although all these previous studies provide valuable insights, a number of constraints exist in their applications. Most rules encoded in knowledge-based systems are style-specific, making them hard to apply to other types of music efficiently. Probabilistic models and neural networks, on the other hand, provide a much more adaptable solution that can be applied to music of different styles by learning rules from different styles of training data. Nevertheless, many of the probabilistic models can only handle music pieces of fixed length. In addition, the transition matrix of probabilistic models has to be learned using specific music representation (e.g. chords) and cannot be generalized to other representations. Moreover, probabilistic models tend to ignore long-term dependency among music components as they mainly focus on local transitions between two consecutive components. Existing studies using neural networks captured long-term dependencies in music and also are capable of dealing with music pieces of arbitrary lengths. However, neural networks have been notoriously hard to train, and their ability to utilize long-term information was limited until the introduction of Long-Short Term Memory (LSTM) cells.

Although BRNNs have access to both past and future information, they have been notoriously hard to train because of "vanishing gradients," a problem commonly seen in RNNs when training with gradient based methods. Gradient methods, such as Back-Propagation Through Time (BPTT) [31], Real-Time Recurrent Learning (RTRL) [26] and their combinations, update the network by flowing errors "back in time." As the error propagates from layer to layer, it tends to either explode or shrink exponentially depending on the magnitude of the weights. Therefore, the network fails to learn long-term dependency between inputs and outputs. Tasks with time lags that are greater than 5-10 time steps are already difficult to learn, not to mention that dependency of music usually spans across tens to hundreds of notes in time, which contributes to music's unique phrase structures. Long short term memory (LSTM) [18] algorithm was designed to tackle the error-flow problem.

In an LSTM hidden layer, fully-connected memory blocks replace nonlinear units that are often used in feed-forward neural network. The core of a memory block is a linear cell that sums up the inputs, which has a self-recurrent connection of fixed weight 1.0, preserving all previous information and ensuring they would not vanish as they are propagated in time. A memory block also contains three sigmoid gating units: input gate, output gate, and forget gate. An input gate learns to control when inputs are allowed to pass into the cell in the memory block so that only relevant contents are remembered; an output gate learns to control when the cell's output should be passed out of the block, protecting other units from interference from current irrelevant memory contents; a forget gate learns to control when it is time to forget already

remembered value, i.e. to reset the memory cell. When gates are closed, irrelevant information does not enter the cell and the state of the cell is not altered. The outputs of all memory blocks are fed back recurrently to all memory blocks to remember past values. Finally, adding *bidirectional links* and LSTM cells improves a neural network's ability to employ additional timing information. All of the above contributes to the fact that the proposed BLSTM model is flexible and effective in generating the missing component in an incomplete multipart texture.

2. METHOD

2.1 Music Representation

MIDI files are used as input in both training and testing phases in this project. Multiple input and output neurons are used to represent different pitches. At each time, the value of the neuron associated with the particular pitch played at that time is 1.0. The values of the rest of the neurons are 0.0. We avoid distributed encodings and other dimension reduction techniques and represent the data in this simple form because this representation is common and assumes that neural networks can learn a more distributed representation within hidden layers.

The music is split into time frames and the length of each frame depends on the type of music. Finding missing music component can then be formulated as a supervised classification problem. For a song of length t_1, for every time t from t_0 to t_1, given input $\mathbf{x}(t)$, the notes played at time t, find the output $\mathbf{y}(t)$, which is the missing component we try to predict. In other words, for two-part MA textures, $\mathbf{y}(t)$ is the chord played at time t, while for four-part SATB textures, $\mathbf{y}(t)$ is the pitch of the missing part at time t.

2.2 Generating Accompaniment in Two-Part MA Texture

2.2.1 Input and Output

The MIDI files are split into eighth note fractions. The inputs at time t, $\mathbf{x}(t)$, are the notes of the melody played at time t. Instead of representing the notes by their MIDI number, which spans the whole range of 88 notes on a keyboard, we used pitch-class representation to encode note pitches into their corresponding pitch-class number. Pitch class, also known as "chroma," is the set of all pitches regardless of their octaves. That is, all C notes (C0, C1, ... etc.) are all classified as pitch-class C. All notes are represented with one of the 12 numbers corresponding to the 12 semitones in an octave. In addition to pitch-class information, two additional values are added as inputs: Note-Begin unit and Beat-On unit. In order to be able to tell when a note ends, a Note-Begin unit is used to differentiate two consecutive notes of the same pitch from one note that is held for two time frames as was done by [30]. If the note in the melody is beginning at the time, the value of the Note-Begin unit is 1.0; if the note is present but duplicates the previous note or is not played at all, the value of the unit is 0.0. The Beat-On unit, on the other hand, provides metric information to the network. If the time t is on a beat, the value of the Beat-On unit is 1.0, otherwise 0.0. If time t is a rest, the values of all input neurons are 0.0. The time signature information is obtained via meta-data in MIDI files.

The outputs at time t, $\mathbf{y}(t)$, is the chord played at time t. We limit chord selection to major, minor, diminished, suspended, and augmented triads as in [27], resulting in 52 chords in total [1]. The output units represent these 52 chords in a manner similar to the input neurons: the value of the neuron corresponding to the chord played at that time has a value of 1.0, and the values of the rest of the neurons are all 0.0.

2.2.2 Training the Network

The input layer has 14 input neurons: 12 neurons for each pitch in the pitch class, one neuron for note-begin and one for beat-on unit. The network consists of two hidden layers for both forward and backward states, resulting in four hidden layers in total. In every hidden layer are 20 LSTM blocks with one memory cell. The output layer uses the softmax activation function and cross entropy error function as in [15]. Softmax function is a standard function for multi-class classification that squashes a K-dimensional vector x in the range of $(0, 1)$, which takes the form

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}}, \quad \text{for } j = 1, ..., K \tag{1}$$

The softmax function ensures that all the output neurons sum to one at every time step, and thus can be regarded as the probability of the output chord given the inputs at that time.

Each music piece is presented to the network one at a time, frame-by-frame. The network is trained via standard gradient-descent Back-Prorogation. A split of data is used as the validation set for early-stopping in order to avoid over-fitting of the training data. If there is no improvement on the validation set for 30 epochs, training is finished and the network setting with the lowest classification error on the validation set is used for testing.

2.2.3 Markov Model as Post-Processing

The network trained in 2.2.2 can then be used to predict the chord associated with each melody note by choosing the output neuron that has the highest activation at each time. However, the predicted chord at each time is independent of the chord predicted in the previous and succeeding time. While there are forward and backward links in the hidden layers of the network, there is no recurrent connections from the final neuron output to the network. The chord might sound good with the melody, but the transition from one chord to another might not make sense at all. In fact,

[1] We represent the note of the chords with their pitches rather than pitch names. Therefore, A augmented chord would have the same representation as F augmented: the former consists of A, C#, and E#, whose pitches are the same as that of the component of the latter, F, A, and C#.

how one chord transits from and to the other typically follows specific chord-progression rules depending on different music styles. A bi-gram Markov Model is thus added to learn the probability of transitioning from each chord to possible successors independent of the melody, which will be referred to as the transition matrix. The transition matrix is smoothed using linear interpolation with a unigram model. The model also learns the statistics of the start chords.

Instead of selecting the output neuron with the highest activations, the first k neurons with the highest activations are chosen as candidates. Dynamic programming is then used to determine the optimal chord sequence among the candidates using the previously-learned transition matrix.

2.3 Generating the Missing Part in Four-Part SATB Textures

2.3.1 Input and Output

Without loss of generality, we sample the melody at every eighth note for similar reasons as explained by Prisco, et al. [7]. Notes that are shorter in length are considered as passing notes and are ignored here. The inputs at time t, $\mathbf{x}(t)$, are the pitches of the notes played at time t, spanning the whole range of 88 notes (A0, C8) on a keyboard, resulting a 88-dimensional vector. If a note i is played at time t, the value of the neuron associated with the particular pitch is 1.0, i.e. $\mathbf{x}_i(t) = 1.0$. The number of non-zero elements in $\mathbf{x}(t)$, which are the number notes played each time, ranges from one to three, depending on the number of voices present.

For the task of predicting the missing voice in a four-part texture where the other three voices are present, the input is polyphonic music. In this case, there are at most three non-zero elements in $\mathbf{x}_t$ for every time t, i.e. $\forall t \sum_{i=1}^{88} \mathbf{x}_i(t) \leq 3$. If the task is to predict one missing voice given only one of the three other voices, there is at most one non-zero element in $\mathbf{x}(t)$. The reason why we do not represent the notes with their pitch-class profile as we did when handling two-part MA texture is that the network depends on octave information to identify which voice the notes belong to. The outputs at time t, $\mathbf{y}(t)$, is the predicted missing note at time t, which falls in the pitch range of any of the four voices, depending on the task specified by our training data. Similarly, the value of the neuron associated with the particular pitch played at the time t is 1.0, otherwise 0.0.

2.3.2 Training the Network

The network structure is the same as the one used in Section 2.2.2 except that the number of input neurons and output neurons are 88, and that we use 20 LSTM blocks for the first hidden layer and 50 LSTM blocks for the second hidden layer. Similar to what we did for two-part MA textures, each music piece is presented to the network one at a time, frame-by-frame. If the task is to generate one missing voice given any of the three other voices, then the three present voices are given to the network individually as if they are independent melodies. In this case, each music piece is actually presented to the network three times and each time only one of the three voices is presented. This method gave the best results.

2.3.3 Predict Missing Voice with the Trained Network

The trained network is ready to predict the missing voice by doing an 88-class classification on the input voice. At each time frame, the neuron with the highest activations is selected, and the pitch it represents is considered as the pitch of the missing voice.

3. EVALUATION

3.1 Generating Missing Accompaniment in Two-Part MA Texture

3.1.1 Dataset

The system's performance on two-part MA textures is evaluated using the Nottingham Dataset [10] transcribed from ABC format, which is also used in [4] for composing polyphonic music. The dataset consists of 1024 double-track MIDI files, with melody on one track and accompaniment on the other. The length of the pieces ranges from 10 seconds to 7.5 minutes, the median being 1 minute and 4 seconds. Those without accompaniment and those whose accompaniment are more complicated than simple chord progressions are discarded, resulting in 962 MIDI files comprising more than 1000 minutes, in total. Songs not in the key of C major nor A minor (874 of them) were transposed to C major/A minor after probabilistically determining their original key using the Krumhansl-Schmuckler key-finding algorithm.

The chords were annotated at every beat or at every quarter note. Seventh chords were reduced to triads, and rests were replaced with previous chords. 60% of the dataset is selected randomly as training data, 20% as validation data, and 20% as testing data. Training finishes when validation accuracy does not improve for 30 epochs. All results for the training and testing sets were recorded at the time when the classification error on the validation set is lowest.

3.1.2 Effects of Including Metric Information in Input

Since the network learns the input melody as a sequence in time and has no access to information other than pitches, we added Beat-On flag to a frame when it is on a beat according to the time signature meta-data in MIDI files (Group iii and iv). We also added Note-Begin (Group ii and iv) to differentiate two consecutive notes of the same pitch from two distinctive notes, as mentioned in Section 2.2.1. All three groups were sampled every eighth note, and the MIDI note range (50, 95) was used as the input range. Table 3.1.2 shows the classification accuracy of the three groups as well as the one where neither flag is provided as a reference. Two groups where Beat-On flag is added, Group iii and iv, perform significantly better than the groups without the beat information (Group i), with a

	Training Set	Test Set
(i) Pitch Information only	72.88%	68.54
(ii) Note-Begin	72.11%	68.86
(iii) Beat-On	75.82%	70.34
(iv) Note-Begin and Beat-On	75.76%	**70.61**

Table 1. Classification accuracy of the dataset when a Note-Begin flag, Beat-On flag, and both flags are added to the inputs.

	Training Set	Test Set
(i) 8th Note + Range	75.76%	70.65 %
(ii) 8th Note + PC	73.13%	**72.05 %**
(iii) 16th Note + Range	73.10%	69.50 %
(iv) 16th Note + PC	74.02%	70.67 %

Table 2. Classification accuracy of the dataset when using various representations of pitches at various sampling rates.

95% confidence interval of 0.84%, 0.80% and 0.79% individually. This is consistent with the fact that chords always change on a beat or multiples of a beat. Therefore, such information is crucial to the timing of chord changes in the network. Note-Begin, on the other hand, does not seem to improve the accuracy, which is due to the fact that whether the note is held from the previous time or it is newly started does not affect chord choices.

3.1.3 Choice of Data Representations

To see how different resolutions of the melody affects the chord prediction result, we evaluated the performance of the system using different frame lengths. "8th Note" or "16th Note" indicates the melodies and accompaniments were sampled every eighth note or sixteenth note. We represented the input to the network using only the actual pitch range that melody notes are played in, which is MIDI note 50 (D3) to 95 (B6) (Groups i and iii, "Melody Range"), and using pitch class representation (Groups ii and iv, "Pitch Class").

Since the network learns the input melody as a sequence in time and have no access to information other than pitches, we added Beat-On flags to a frame when it is on a beat according to the time signature meta-data in MIDI files. We also added Note-Begin flags. Representing the melodies with their pitch-class number at every 8th note (Group ii) could correctly predict the missing chords approximately 72% of the time when both Note-Begin and Beat-On information are available. With a 95% confidence interval at 0.76%, it also significantly outperforms the other representation. Table 2 shows the result.

3.1.4 Comparison with Other Approaches

We compared the architecture used in this paper with four other neural network architectures: Unidirectional LSTM, Bidirectional recurrent neural network (BRNN), Unidirectional recurrent neural network (RNN), and Multi-layer perceptron network (MLP). Given the variety of different datasets and accessibility to code, our comparison is based on the BLSTM methods described above. Neurons in BRNN, RNN and MLP networks were sigmoid neurons. The size of the hidden layers were selected so that the number of weights are approximately the same (around 32,000) for all of the networks as in [15]

Network	Training Set	Test Set	Epochs
BLSTM	**75.76%**	**71.13 %**	103
LSTM	71.51%	67.57 %	130
BRNN	68.77%	68.86 %	136
RNN	68.33%	66.58 %	158
MLP	55.16%	54.66 %	120

Table 3. Classification accuracy of the dataset using different neural network architectures.

Table 3 shows the classification accuracy and the number of epochs required to converge. All groups were sampled at every eighth note, and were provided with both metric information, (Note On and Beat On), during training and testing. The 95% confidence interval for BLSTM and LSTM are 0.80% and 0.76%. Using approximately same number of weights, BLSTM performs significantly better than other neural networks and also converges the fastest.

3.2 Finding the Missing Part in Four-Part SATB Textures

3.2.1 Dataset

We evaluated our approach using 378 of J. S. Bach's four-part chorales acquired from [16]. MIDI files were all multi-tracked, one voice on each track. The average length of the pieces is approximate 45 seconds, the maximum and minimum being 6 minutes and 17 seconds. Among all chorales, 102 pieces are in minor mode. All of the chorales were transposed to C major/A minor using Krumhansl-Schmuckler key-finding algorithm. As in section 3.1, 60% of the files were used as training set, 20% as test set, and 20% as validation set, resulting in 226, 76, 76 pieces respectively.

3.2.2 Predicting Missing Voice Given the Other Three Voices

Table 3.2.2 shows the frame-wise classification accuracy of the predicted missing voices (Soprano, Alto, Tenor, or Bass) when the three other voices are given on training and test sets. The accuracy of predicting missing voices on the original non-transposed set is also listed for comparison. All songs were sampled at every eighth note. From the table, we can observe a few interesting phenomena. First, transposing the songs remarkably improves prediction accuracy in both training and test set. This is not surprising since transposing songs in advance reduces complexity. The same pre-processing is also used by [3] [4] [27]. Second, we see that the network could correctly predict Soprano, Alto, and Tenor approximately 70% of the time when the songs were transposed. Specifically, Alto seems

	Soprano		Alto	
	Training	Test	Training	Test
Not Transposed	69.15%	46.82%	63.61%	47.61%
Transposed	77.90%	71.52%	82.65%	**73.90%**
	Tenor		Bass	
	Training	Test	Training	Test
Not Transposed	47.25%	39.85%	45.40%	36.93%
Transposed	78.47%	69.76%	70.09%	61.22%

Table 4. Classification accuracy of the predicted missing voices, either Soprano, Alto, Tenor, or Bass, when the three other voices are given on training and testing sets.

	Soprano		Alto	
	Training	Test	Training	Test
BLSTM	84.88%	73.86 %	82.65%	73.90 %
BRNN	90.25%	**74.37%**	85.37%	**74.30 %**
LSTM	85.27%	70.39%	77.14%	70.45%
RNN	81.90%	72.29%	80.31%	71.73%
MLP	68.74%	66.54%	73.51%	70.03%
	Tenor		Bass	
	Training	Test	Training	Test
BLSTM	78.47%	69.76%	70.09%	61.22 %
BRNN	80.95%	**70.13%**	74.58%	**63.74%**
LSTM	73.84%	64.89%	65.86%	57.69%
RNN	75.48%	67.20%	69.68%	59.69%
MLP	68.85%	65.68%	58.58%	56.14%

Table 5. Classification accuracy of the predicted missing voices when three other voices are given using different network architecture.

to be the easiest to predict (in bold), while Bass is the most difficult.

3.2.3 Comparison with Other Approaches

Similar to our approach in Section 3.1.4, the size of the hidden layers were selected so that the number of weights are approximately the same (around 63,000) for all of the networks. Table 3.2.3 shows the classification accuracy of the missing voices (either Soprano, Alto, Tenor, or Bass) when all of the three other voices are present. From the result, we can see that BLSTM does not have a statistically significant performance from BRNN on Soprano, Alto, and Tenor parts (in bold) and outperforms other neural-network based methods on all parts. It also shows that including future information by using bidirectional connection effectively improves accuracy by 3% on average no matter using LSTM cells (in BLSTM and LSTM) or logistic cells (in BRNN and RNN). Note that LSTM, while powerful, is really hard to train since it requires parameter tuning and a large dataset. We will need to conduct more experiments in larger scale to explain what properties of LSTM and BRNN favor which tasks.

4. CONCLUSION

This paper has presented an approach to predicting missing music components for complex multipart musical textures using Bidirectional Long-Short Term Memory (BLSTM) neural networks. We demonstrated the flexibility and robustness of the system by applying the method to two distinctive but popular tasks in the computer-music field: generating chord accompaniment for given melodies in two-part MA textures and filling the missing voice in four-part SATB textures. The proposed approach is capable of handling music pieces of arbitrary length as well as various styles. In addition, the network could be used to generate missing music components of different forms, i.e. single notes for four-part SATB textures or chords for two-part MA textures, by simply altering the number of input and output neurons.

Two sets of experiments were conducted regarding the two tasks on two datasets of completely different styles, and issues that influence prediction accuracies were discussed. For the task of predicting chord accompaniment in two-part MA texture, the experimental results showed that BLSTM network could correctly generate chords for given melodies 72% of the time, which is significantly higher than 68%, the best accuracy achieved by using other neural network based approaches. We also discovered that representing the melodies using their pitch class profile yielded the best result.

As for the problem of finding the missing voice in four-part SATB texture, the experiment demonstrated that BLSTM network could correctly predict the missing voice approximately 70% of the time on average when three other voices are present. Putting the experimental results on two datasets together, the fact that BLSM outperforms other neural-network based networks for two-part MA textures and performs as well as BRNN for four-part SATB textures showed that the BLSTM network is the optimal structure for predicting missing components from multipart musical textures.

5. REFERENCES

[1] Moray Allan and Christopher KI Williams. Harmonising chorales by probabilistic inference. *Advances in neural information processing systems*, 17:25–32, 2005.

[2] Matthew I Bellgard and Chi-Ping Tsang. Harmonizing music the Boltzmann way. *Connection Science*, 6(2-3):281–297, 1994.

[3] Kaan M Biyikoglu. A Markov model for chorale harmonization. In *Preceedings of the 5 th Triennial ESCOM Conference*, pages 81–84, 2003.

[4] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

[5] Ching-Hua Chuan and Elaine Chew. A hybrid system for automatic generation of style-specific accompaniment. In *4th Intl Joint Workshop on Computational Creativity*, 2007.

[6] Uraquitan Sidney Cunha and Geber Ramalho. An intelligent hybrid model for chord prediction. *Organised Sound*, 4(02):115–119, 1999.

[7] Roberto De Prisco, Antonio Eletto, Antonio Torre, and Rocco Zaccagnino. A neural network for bass functional harmonization. In *Applications of Evolutionary Computation*, pages 351–360. Springer, 2010.

[8] Kemal Ebcioğlu. An expert system for harmonizing four-part chorales. *Computer Music Journal*, pages 43–51, 1988.

[9] Johannes Feulner. Neural networks that learn and reproduce various styles of harmonization. In *Proceedings of the International Computer Music Conference*, pages 236–236. International Computer Music Association, 1993.

[10] Eric Foxley. Nottingham dataset. http://ifdo.ca/ seymour/nottingham/nottingham.html, 2011. Accessed: 04-19-2015.

[11] Alan Freitas and Frederico Guimaraes. Melody harmonization in evolutionary music using multiobjective genetic algorithms. In *Proceedings of the Sound and Music Computing Conference*, 2011.

[12] Dan Gang, D Lehman, and Naftali Wagner. Tuning a neural network for harmonizing melodies in real-time. In *Proceedings of the International Computer Music Conference, Ann Arbor, Michigan*, 1998.

[13] Dan Gang and Daniel Lehmann. An artificial neural net for harmonizing melodies. *Proceedings of the International Computer Music Association*, 1995.

[14] Dan Gang, Daniel Lehmann, and Naftali Wagner. Harmonizing melodies in real-time: the connectionist approach. In *Proceedings of the International Computer Music Association, Thessaloniki, Greece*, 1997.

[15] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005.

[16] Margaret Greentree. www.jsbchorales.net/index.shtml, 1996. Accessed: 04-19-2015.

[17] Mark A Hall. Selection of attributes for modeling Bach chorales by a genetic algorithm. In *Artificial Neural Networks and Expert Systems, 1995. Proceedings., Second New Zealand International Two-Stream Conference on*, pages 182–185. IEEE, 1995.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[19] Amy K Hoover, Paul A Szerlip, Marie E Norton, Trevor A Brindle, Zachary Merritt, and Kenneth O Stanley. Generating a complete multipart musical composition from a single monophonic melody with functional scaffolding. In *International Conference on Computational Creativity*, page 111, 2012.

[20] Amy K Hoover, Paul A Szerlip, and Kenneth O Stanley. Generating musical accompaniment through functional scaffolding. In *Proceedings of the Eighth Sound and Music Computing Conference (SMC 2011)*, 2011.

[21] Ryan A McIntyre. Bach in a box: The evolution of four part baroque harmony using the genetic algorithm. In *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pages 852–857. IEEE, 1994.

[22] François Pachet and Pierre Roy. Mixing constraints and objects: A case study in automatic harmonization. In *Proceedings of TOOLS Europe*, volume 95, pages 119–126. Citeseer, 1995.

[23] Jean-François Paiement, Douglas Eck, and Samy Bengio. Probabilistic melodic harmonization. In *Advances in Artificial Intelligence*, pages 218–229. Springer, 2006.

[24] Stanisław A Raczyński, Satoru Fukayama, and Emmanuel Vincent. Melody harmonization with interpolated probabilistic models. *Journal of New Music Research*, 42(3):223–235, 2013.

[25] Rafael Ramirez and Julio Peralta. A constraint-based melody harmonizer. In *Proceedings of the Workshop on Constraints for Artistic Applications (ECAI'98)*, 1998.

[26] AJ Robinson and Frank Fallside. *The utility driven dynamic error propagation network*. University of Cambridge Department of Engineering, 1987.

[27] Ian Simon, Dan Morris, and Sumit Basu. Mysong: automatic accompaniment generation for vocal melodies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 725–734. ACM, 2008.

[28] Luc Steels. *Learning the craft of musical composition*. Ann Arbor, MI: MPublishing, University of Michigan Library, 1986.

[29] Syunpei Suzuki, Tetsuro Kitahara, and Nihon Univercity. Four-part harmonization using probabilistic models: Comparison of models with and without chord nodes. *Stockholm, Sweden*, pages 628–633, 2013.

[30] Peter M Todd. A connectionist approach to algorithmic composition. *Computer Music Journal*, pages 27–43, 1989.

[31] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

STRUCTURAL SEGMENTATION AND VISUALIZATION OF SITAR AND SAROD CONCERT AUDIO

Vinutha T.P. **Suryanarayana Sankagiri** **Kaustuv Kanti Ganguli** **Preeti Rao**
Department of Electrical Engineering, IIT Bombay, India
prao@ee.iitb.ac.in

ABSTRACT

Hindustani classical instrumental concerts follow an episodic development that, musicologically, is described via changes in the rhythmic structure. Uncovering this structure in a musically relevant form can provide for powerful visual representations of the concert audio that is of potential value in music appreciation and pedagogy. We investigate the structural analysis of the metered section (gat) of concerts of two plucked string instruments, the sitar and sarod. A prominent aspect of the gat is the interplay between the melody soloist and the accompanying drummer (tabla). The tempo as provided by the tabla together with the rhythmic density of the sitar/sarod plucks serve as the main dimensions that predict the transition between concert sections. We present methods to access the stream of tabla onsets separately from the sitar/sarod onsets, addressing challenges that arise in the instrument separation. Further, the robust detection of tempo and the estimation of rhythmic density of sitar/sarod plucks are discussed. A case study of a fully annotated concert is presented, and is followed by results of achieved segmentation accuracy on a database of sitar and sarod gats across artists.

1. INTRODUCTION

The repertoire of North Indian (Hindustani) classical music is characterized by a wide variety of solo instruments, playing styles and melodic material in the form of *ragas* and compositions. However, across all these, there is a striking universality in the concert structure, i.e., the way in which the music is organized in time. The temporal evolution of a concert can be described via changes in the rhythm of the music, with homogenous sections having identical rhythmic characteristics. The metric tempo and the surface rhythm, two important aspects of rhythm, characterize the individual sections. Obtaining these rhythm features as they vary with time gives us a rich transcription for music appreciation and pedagogy. It also allows rhythm-base segmentation with potential applications in concert summarization, music navigation. This provides a strong motivation for the rhythmic analysis of Hindustani classical concert audio.

Rhythmic analyses of audio has been widely used for music classification and tempo detection [1–3]. It has also been applied to music segmentation [4,5] although timbre- and harmony-based segmentation are more common. Recently, computational descriptions of rhythm were studied for Indian and Turkish music [6]. Beat detection and cycle length annotation were identified as musically relevant tasks that could benefit from the computational methods.

In this paper, we focus on the Hindustani classical instrumental concert which follows an established structure via a specified sequence of sections, viz. *alap-jod-jhala-gat* [7]. The first three are improvised sections where the melody instrumentalist (sarod/sitar) plays solo, and are often together called the "*alap*". The *gat* or composed section is marked by the entry of the tabla. The *gat* is further subdivided into episodes as discussed later. The structure originated in the ancient style of *dhrupad* singing where a *raga* performance is subdivided unequally into the mentioned temporally ordered sections.

In the present work, we consider concerts of two plucked string instruments, sitar and sarod, which are major components of Indian instrumental music. The two melodic instruments share common origins and represent the fretted and unfretted plucked monochords respectively. Verma et. al. [8] have worked on the segmentation of the unmetered section (*alap*) of such concerts into *alap-jod-jhala* based purely on the tempo and its salience. They use the fact that an increase in regularity and pluck density marked the beginning of *jod*. Higher pluck density was captured via increases in the energy and in the estimated tempo. The transition to *jhala* was marked by a further rise in tempo and additionally distinguished by the presence of the "*chikari*" strings.

In this paper, we focus on the rhythmic analysis and segmentation of the *gat*, or the tabla-accompaniment region, into its sections. Owing to differences in the rhythmic structure of the *alap* and the *gat*, the challenges involved in this task are different from those addressed in [8]. In the *gat*, the tabla provides a definite meter to the concert by playing a certain *tala*. The tempo, as set by the tabla, is also called the metric tempo. The tempo of the concert increases gradually with time, with occasional jumps. While the tabla provides the basic beats (*theka*), the melody instrumentalist plays the composition interspersed

© Vinutha T.P., Suryanarayana Sankagiri, Kaustuv Kanti Ganguli, Preeti Rao. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Vinutha T.P., Suryanarayana Sankagiri, Kaustuv Kanti Ganguli, Preeti Rao. "STRUCTURAL SEGMENTATION AND VISUALIZATION OF SITAR AND SAROD CONCERT AUDIO", 17th International Society for Music Information Retrieval Conference, 2016.

with *raga*-based improvisation ("*vistaar*"). A prominent aspect of instrumental concerts is that the *gat* is characterized by an interplay between the melody instrumentalist and the drummer, in which they alternate between the roles of soloist and timekeeper [7, 9]. The melody instrument can switch to fast rhythmic play ("*layakari*") over several cycles of the tabla. Then there are interludes where the tabla player is in the foreground ("tabla solo"), improvising at a fast rhythm, while the melody instrumentalist plays the role of the timekeeper by playing the melodic refrain of the composition cyclically. Although both these sections have high surface rhythm, the term "rhythmic density" refers to the stroke density of the sarod/sitar [10], and therefore is high only during the *layakari* sections. The values of the concert tempo and the rhythmic density as they evolve in time can thus provide an informative visual representation of the concert, as shown in [10].

In order to compute the rhythmic quantities of interest, we follow the general strategy of obtaining an onset detection function (ODF) and then computing the tempo from it [11]. To obtain the surface rhythm, we need an ODF sensitive to all onsets. However, to calculate the metric tempo, as well as to identify sections of high surface rhythm as originating from the tabla or sarod/sitar, we must discriminate the tabla and sitar/sarod stroke onsets. Both the sitar and the sarod are melodic instruments but share the percussive nature of the tabla near the pluck onset. The tabla itself is characterized by a wide variety of strokes, some of which are diffused in time and have decaying harmonic partials. This makes the discrimination of onsets particularly challenging.

Our new contributions are the (i) proposal of a tabla-specific onset detection method, (ii) computation of the metric tempo and rhythmic density of the *gat* over a concert to obtain a rhythmic description which matches with one provided by a musician, (iii) segmentation of the *gat* into episodes based on the rhythm analysis. These methods are demonstrated on a case study of a sarod *gat* by a famous artist, and are further tested for segmentation accuracy on a manually labeled set of sitar and sarod *gats*.

In section 2, we present the proposed tabla-sensitive ODF and test its effectiveness in selectively detecting tabla onsets from a dataset of labeled onsets drawn from a few sitar and sarod concerts. In section 3, we discuss the estimation of tempo and rhythmic density from the periodicity of the onset sequences and present the results on a manually annotated sarod *gat*. Finally, we present the results of segmentation on a test set of sitar and sarod *gats*.

2. ONSET DETECTION

A computationally simple and effective method of onset detection is the spectral flux which involves the time derivative of the short-time energy [12]. The onsets of both the percussive as well as the string instrument lead to a sudden increase in energy, and are therefore detected well by this method. A slight modification involves using a biphasic filter to compute the derivative [13]. This enhances the detection of sarod/sitar onsets, which have a slow decay in energy, and leads to a better ODF. Taking the logarithm of the energy before differencing enhances the sensitivity to weaker onsets. We hereafter refer to this ODF as the spectral flux-ODF (SF-ODF), and is given by Eq. 1. ($h[n]$ *denotes the biphasic filter as in [13]*)

$$SF\text{-}ODF[n] = h[n] * log(\sum_{k=0}^{N/2} |X[n,k]|) \quad (1)$$

Figure 1, which contains a sarod concert excerpt, illustrates the fact that SF-ODF is sensitive to both sarod and tabla onsets. In this example, and in all subsequent cases, we compute the spectrum by using a 40ms Hamming window on audio sampled at 16 kHz. The spectrum (and therefore the ODF) is computed at 5 ms intervals. Fig. 1(a) shows the audio waveform where onsets can be identified by peaks in the waveform envelope. Onsets can also be seen as vertical striations in the spectrogram (Fig. 1(b)). SF-ODF is shown in Fig. 1(c). Clearly, SF-ODF is not tabla-selective.

In order to obtain a tabla-sensitive ODF, we need to exploit some difference between tabla and sarod/sitar onsets. One salient difference is that in the case of a tabla onset, the energy decays very quickly (< 0.1 s). In contrast, the energy of a sitar/sarod pluck decays at a much slower rate (> 0.5 s). This difference is captured in the ODF that we propose, hereafter called as P-ODF. This ODF counts the number of bins in a spectral frame where the energy increases from the previous frame, and is given by Eq. 2. This method is similar in computation to the spectral flux method in [12]; we take the 0-norm of the half-wave rectified energy differences, instead of the 2-norm [12] or 1-norm [14]. However, the principle on which this ODF operates is different from the spectral flux ODF. P-ODF detects only those onsets that are characterised by a wide-band event, i.e., onsets that are pecussive in nature. Unlike the spectral flux ODF, it does not rely on the magnitude of energy change. In our work, this proves to be an advantage as it detects weak onsets of any instrument better, provided they are wide-band events.

$$P\text{-}ODF[n] = \sum_{k=0}^{N/2} \mathbb{1}\{|X[n,k]| > |X[n-1,k]|\} \quad (2)$$

From Fig. 1(d), we see that P-ODF peaks at the onset of a tabla stroke, as would be expected due to the wide-band nature of these onsets. It also peaks for sarod onsets, as these onsets have a percussive character. Thus, it is sensitive to all onsets of interest, and can be potentially used as generic ODF in place of SF-ODF, for sitar/sarod audio. What is of more interest is the fact that in the region immediately following a tabla onset, this count falls rapidly while such a pattern is not observed for sarod onsets (see Fig. 1(d)). This feature is seen because of the rapid decrease in energy after a tabla onset. In the absence of any activity, the value of the ODF is equal to half the number of bins as the energy changes from frame to frame in a bin due to small random perturbations.

The sharp downward lobe in P-ODF is a striking feature of tabla onsets, and can be used to obtain a tabla-sensitive

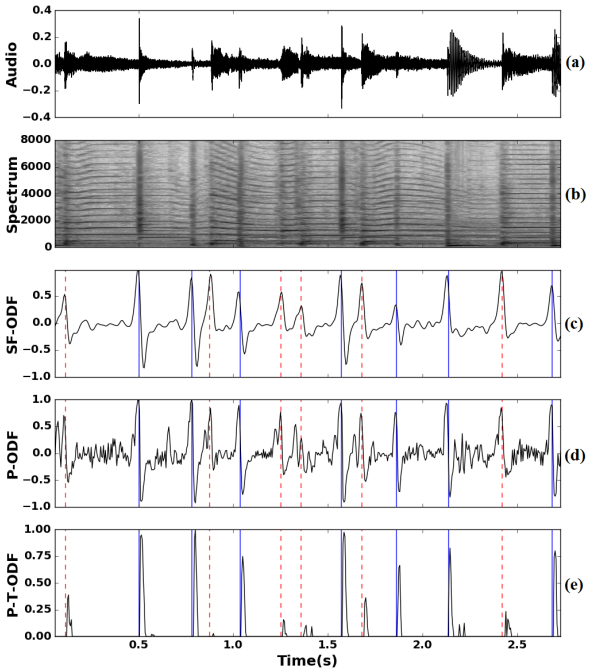


Figure 1: **(a)** Audio waveform, **(b)** Spectrogram, **(c)** SF-ODF, **(d)** P-ODF and **(e)** P-T-ODF of an excerpt of a sarod concert. All ODFs normalised. Tabla onsets marked in blue solid lines; sarod onsets marked in red dashed lines

ODF. We normalize the mean-removed function to [-1,1] and consider only the negative peaks of magnitude that exceed the empirically chosen threshold of 0.3. We call our proposed tabla-sensitive ODF as P-T-ODF. An example is shown in Fig. 1(e).

We wish to establish that the P-T-ODF performs better as a tabla-sensitive ODF than other existing methods. The spectral flux method is known to be sensitive to both onsets, and performs poorly as a tabla-sensitive ODF. However, one could hope to obtain better results by computing the ODF on a percussion-enhanced audio. Fitzgerald [15] proposes a median-filter based method for percussion enhancement that exploits the relatively high spectral variability of the melodic component of a music signal to suppress it relative to the more repetitive percussion. We used this method to preprocess our *gat* audio to obtain what we call the enhanced audio signal (tabla is enhanced), and test the SF-ODF on it. With this as the baseline, we compare our P-T-ODF applied to the original audio. In parallel, we wish to justify our claim that the P-ODF is a suitable ODF for detecting sarod/sitar as well as tabla onsets.

We evaluate our ODFs on a dataset of 930 labeled onsets comprising 158 sitar, 239 sarod and 533 tabla strokes drawn from different sections of 6 different concert gats. Onsets were marked by two of the authors, by carefully listening to the audio, and precisely locating the onset instant with the aid of the waveform and the spectrogram. We evaluate P-ODF and SF-ODF, derived from the original audio, for detection of all onsets, with SF-ODF serving as a baseline. The obtained ROC is shown in Fig. 2(a). We also evaluate P-T-ODF, derived from the original audio

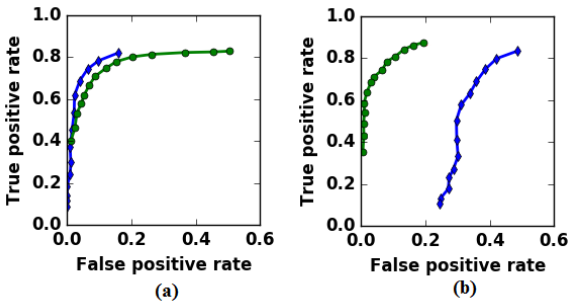


Figure 2: **(a)** All-onsets ROC for SF-ODF (blue diamonds) and P-ODF (green circles); **(b)** Tabla-onsets ROC for SF-ODF on enhanced audio (blue diamonds), and P-T-ODF on original audio (green circles)

and compare it with SF-ODF from enhanced audio, for detection of tabla onsets. The corresponding ROC is shown in Fig. 2(b).

We observe that the spectral flux and the P-ODF perform similarly in the all-onsets ROC of Fig. 2(a). A close examination of performance on the sitar and sarod gats separately revealed that the P-ODF performed marginally better than SF-ODF on sarod *gats*, while the performance of the spectral flux ODF was better than the P-ODF on the sitar strokes. In the following sections, we use the P-ODF to detect all onsets in sarod gats and the spectral flux-ODF on the sitar gats. We also note from Fig. 2(b) that the P-T-ODF fares significantly better than the SF-ODF applied on tabla-enhanced signal. The ineffectiveness of Fitzgerald's percussion enhancement is explained by the percussive nature of both instruments as well as the high variation (intended and unintended) of tabla strokes in performance. We observed that the median filtering did a good job of suppressing the sarod/sitar harmonics in but not their onsets. The P-T-ODF is established as an effective way to detect tabla onsets exclusively in both sarod and sitar *gats*.

3. RHYTHMOGRAMS AND TEMPO ESTIMATION: A CASE STUDY

A rhythm representation of a *gat* can be obtained from the onset detection function by periodicity analysis via the autocorrelation function (ACF) or the DFT. A rhythmogram uses the ACF to represent the rhythmic structure as it varies in time [16]. Abrupt changes in the rhythmic structure can be detected for concert section boundaries. The dominant periodicity at any time can serve as an estimate of the perceived tempo [5, 11]. Our goal is to meaningfully link the outcomes of such a computational analysis to the musicological description of the concert.

In this section, we present the musicological and corresponding computational analyses of a commercially recorded sarod *gat* (*Raga Bahar, Madhyalaya, Jhaptal*) by legendary sarodist Ustad Amjad Ali Khan. The musicological description was prepared by a trained musician on lines similar to the sitar *gat* case study by Clayton [17] and is presented next. The computational analysis involved applying the onset detection methods to obtain a rhythm rep-

resentation that facilitates the detection of the metric tempo and rhythmic density as well as the segmentation of the *gat*.

3.1 Annotation by a Trained Musician

A musician with over 15 years of training in Hindustani classical music made a few passes listening to the audio (duration 14 min) to annotate the *gat* at three levels. The first was to segment and label the sequence of distinct episodes as shown in Table 1. These labels reflect the performers' (i.e. the sarod and tabla players) intentions as perceived by a trained listener. The next two annotation levels involved marking the time-varying metric tempo and a measure of the sarod rhythmic density. The metric tempo was measured by tapping to the tabla strokes that define the *theka* (i.e. the 10 beats of the *Jhaptal* cycle) and computing the average BPM per cycle with the aid of the Sonic Visualizer interface [18]. The metric tempo is constant or slowly increasing across the concert with three observed instants of abrupt change.

The rhythmic density, on the other hand, was obtained by tapping to the sarod strokes and similarly obtaining a BPM per cycle over the duration of the *gat*. Figure 3 shows the obtained curves with the episode boundaries in the background. We note that the section boundaries coincide with abrupt changes in the rhythmic density. The metric tempo is constant or slowly increasing across the concert with three observed instants of abrupt change. The rhythmic density corresponds to the sarod strokes and switches between being once/twice the tempo in the *vistaar* to four times in the *layakari* (rhythmic improvisation by the melody soloist). Although the rhythmic density is high between cycles 20-40, this was due to fast melodic phrases occupying part of the rhythmic cycle during the *vistaar* improvisation. Since this is not a systematic change in the surface rhythm, it was not labeled *layakari* by our musician. In the tabla solo section, although the surface rhythm increases, it is not due to the sarod. Therefore, the tabla solo section does not appear distinctive in the musician's markings in Figure 3.

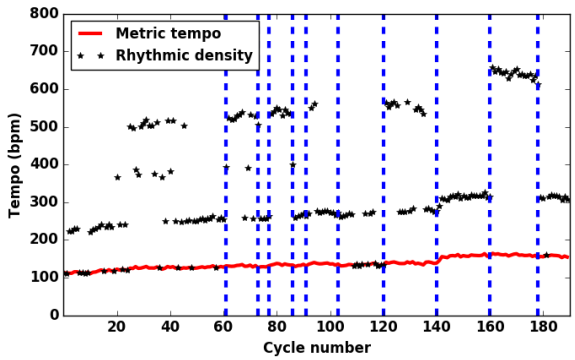


Figure 3: Musicians annotation of tempo and rhythmic density attributes across the *gat*. Dashed lines indicate section boundaries

Sec. No.	Cycles	Time (s)	Label
1	1-61	0-301	Vistaar *
2	62-73	302-356	Layakari
3	74-77	357-374	Vistaar
4	78-86	375-414	Layakari
5	87-91	415-441	Vistaar
6	92-103	442-490	Tabla solo
7	104-120	491-568	Vistaar
8	121-140	567-643	Layakari
9	141-160	644-728	Vistaar #
10	161-178	729-797	Layakari
11	179-190	798-839	Vistaar

Table 1: Labeled sections for the sarod case study. *Tempo increases at 67s & 127s; # also at 657s

3.2 Computational Analysis

3.2.1 Rhythmogram

The onset detection methods of Section 2 are applied over the duration of the concert. We confine our study to two ODFs based on insights obtained from the ROCs of Fig. 2. These are the P-ODF for all onsets and the P-T-ODF for tabla-onsets. Although the P-ODF was marginally worse than spectral flux in Fig. 2(a), it was found to detect weak sarod strokes better while the false alarms were irregularly distributed in time. This property is expected to help us track the sarod rhythmic density better.

The autocorrelation function of the ODFs is computed frame-wise, with a window length of 3 seconds and a hop of 0.5 seconds up to a lag of 1.5 seconds, and is normalized to have a maximum value of 1 in each frame. To improve the representation of peaks across the dynamic range in the rhythmogram, we perform a non-linear scaling of the amplitude of the ACF. For the tabla-centric rhythmogram (from P-T-ODF), we take the logarithm of the ACF between 0.1 and 1; for the generic rhythmogram (from P-ODF), the logarithm is taken between 0.01 and 1 due to its inherently wider dynamic range for peaks. The ACF values below this range are capped to a minimum of -10. This is followed by smoothing in the lag and time axes by moving average filters to length 3 and 10 respectively bringing in short-time continuity.

We thus obtain the two rhythmograms shown in Figures 4 and 5. We note that the P-ODF all-onsets rhythmogram (Figure 4) captures the homogenous rhythmic structure of each episode of *vistaar*, *layakari* and tabla solo, showing abrupt changes at the boundaries. Each section itself appears homogenous except for some spottiness in the sequence of low amplitude ACF peaks at submultiple lags (such as near 0.1s in the region until 300 s).

The tabla-centric rhythmogram (Figure 5), on the other hand, with its more prominent peaks appearing at lags near 0.5s and multiples, is indicative of a metric (base) tempo of around 120 BPM. We clearly distinguish from this rhythmogram, the tabla solo segment (where the tabla surface rhythm shoots up to 8 times the metric tempo). We observe, as expected, that the sarod *layakari* sections are

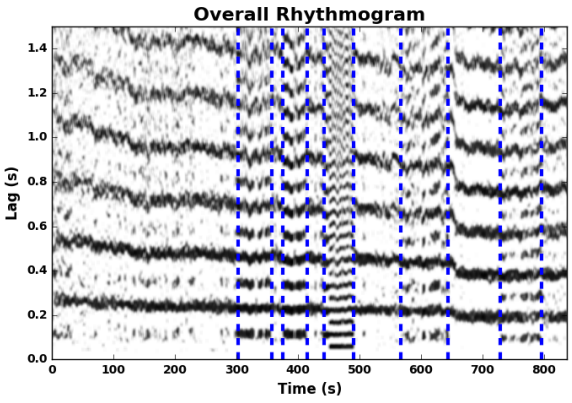


Figure 4: All-onsets rhythmogram from P-ODF

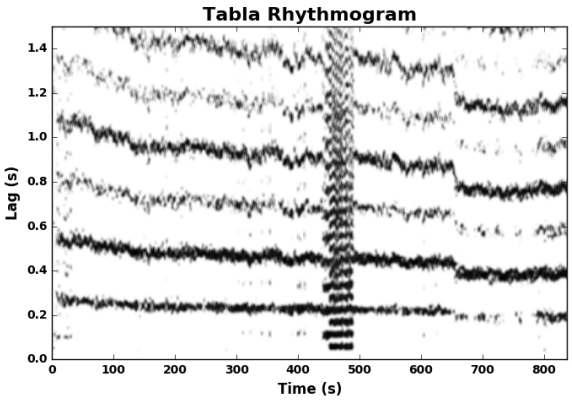


Figure 5: Tabla centric rhythmogram from P-T-ODF

completely absent from the tabla-centric rhythmogram.

3.2.2 Tempo and surface rhythm estimation

The rhythmograms provide interesting visual representations of the rhythmic structure. However a visual representation that is more amenable to immediate interpretation by musicians and listeners would have to parallel the musician's annotation of Fig. 3. We therefore must process the rhythmograms further to extract the relevant attributes of metric tempo and sarod rhythmic density. We present next the frame-wise estimation of these from the ACF vectors of the smoothened rhythmograms of Figs. 4 and 5.

The basic or metric tempo is obtained from the tabla rhythmogram (Fig. 5) by maximizing the mean of the peaks at candidate lags and corresponding lag multiples over the lag range of 50ms to 750ms (1200BPM to 80BPM). The estimated time-varying metric tempo is shown in Fig. 6(a) superposed on the ground-truth annotation (x-axis converted to time from cycles as in Fig. 3). We observe a near perfect match between the two with the exception of the tabla-solo region, where the surface rhythm was tracked. We use our knowledge that the surface rhythm would be a multiple of the metric tempo. Dividing each tempo value by that multiple that maintains continuity of the tempo gave us the detected contour of Fig. 6(a).

The rhythmic density of the sarod is the second musical attribute required to complete the visual representation. This is estimated from the generic (P-ODF) rhythmogram of Fig. 4 in a manner similar to that used on the table-centric version. The single difference is that we apply a bias favouring lower lags in the maximum likelihood tempo estimation. A weighting factor proportional to the inverse of the lag is applied. The biasing is motivated by our stated objective of uncovering the surface rhythmic density (equivalent to the smallest inter-onset interval).

The obtained rhythmic density estimates are shown in Fig. 6(b), again in comparison with the ground truth marked by the musician. The ground-truth markings have been converted to the time axis while smoothening lightly to remove the abrupt cycle-to-cycle variations in Fig. 3. We note that the correct tempo corresponding to the sarod surface rhythm is captured for the most part. The *layakari* sections are distinguished from the *vistaar* by the doubling of the rhythmic density. Obvious differences between the ground-truth and estimated rhythmic density appear in (i) the table solo region due to the high surface rhythm contributed by tabla strokes. Since P-ODF captures both the instrument onsets, this is expected. Another step based on the comparison of the two rhythmograms would easily enable us to correct this; (ii) intermittent regions in the 0-300s region of the *gat*. This is due to the low amplitude ACF peaks arising from the fast rhythmic phrases discussed in Sec. 3.1.

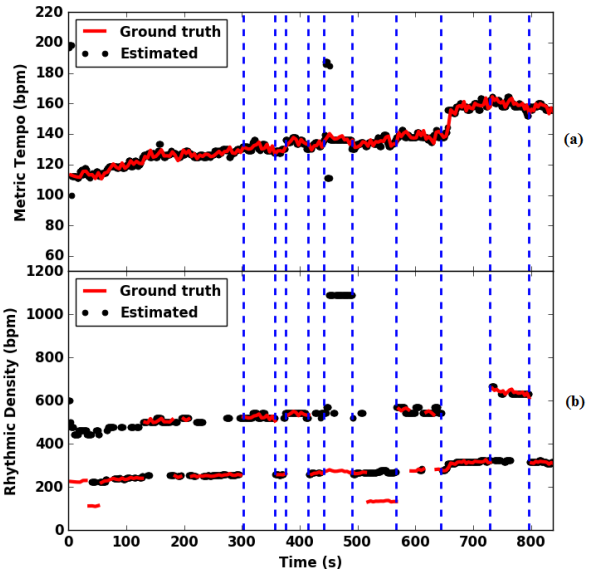


Figure 6: **(a)** Estimated metric tempo with musician's marked tempo. **(b)** Estimated rhythmic density with musicians marked rhythmic density

4. SEGMENTATION PERFORMANCE

The all-onsets rhythmogram provides a clear visual representation of abrupt rhythmic structure changes at the section boundaries specified by the ground-truth labels. In order to algorithmically detect the segment boundaries, we resort to the method of the similarity distance matrix (SDM) where peaks in the novelty function derived from diagonal kernel convolution can help identify instants of

change [19]. We treat the ACF at each time frame as a feature vector that contains the information of the local rhythmic structure. We compute the correlation distance between the ACF of every pair of frames across the concert to obtain the SDM. The diagonal of the SDM is then convolved with a checker-board kernel of 25s $\times$ 25s to compute the novelty function. Local maxima in the novelty function are suitably thresholded to locate instants of change in the rhythmic structure. Figure 7 shows the SDM and novelty function computed on the rhythmogram of Figure 5 corresponding to the case study sarod *gat*. We observe that all the known boundaries coincide with sharp peaks in the novelty function. The *layakari-vistaar* boundary at 644s is subsumed by the sudden tempo change at 657s due to the minimum time resolution imposed by the SDM kernel dimensions. We next present results for performance of our system on segment boundary detection across a small dataset of sitar and sarod *gats*.

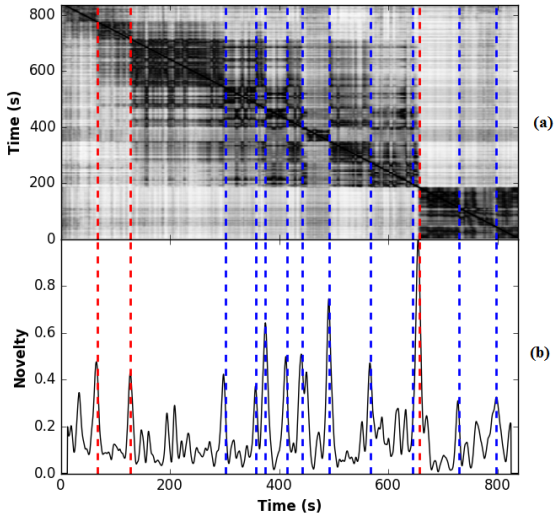


Figure 7: SDM and novelty curve for the case study sarod *gat* (whose rhythmogram appears in Figure 5). The blue dashed lines indicate ground-truth section boundaries as in Table 1. The red dashed lines indicate ground-truth instants of metric tempo jump.

4.1 Dataset

Our dataset for structural segmentation analysis consists of three sitar and three sarod *gats*, by four renowned artists. We have a total of 47 min of sarod audio (including the case study gat) and 64 min of sitar audio. Just like the case-study gat, each gat has multiple sections which have been labelled as *vistaar*, *layakari* and tabla solo. Overall we have 37 *vistaar* sections, 21 *layakari* sections and 25 tabla solo sections. Boundaries have been manually marked by noting rhythm changes upon listening to the audio. Minimum duration of any section is found to be 10s.

Gat. No.	Dur (min)	Method Used	Hit rate	False Alarms
1	14	P-ODF	13/13	0
2	24	P-ODF	14/14	1
3	9	P-ODF	20/20	2
4	16	SF-ODF	17/17	2
5	21	SF-ODF	11/12	1
6	27	SF-ODF	14/14	4

Table 2: Boundary detection results for 6 *gats*

4.2 Boundary Detection Performance

For each concert, the novelty function was normalised to [0,1] range and peaks above a threshold of 0.3 were taken to indicate boundary instants. We consider the detected boundary as a hit if it lies within 12.5 s of a marked boundary considering our kernel dimension of 25 s. We expect to detect instants where there is either a change in surface rhythm or an abrupt change in the metric tempo. Consistent with our onsets detection ROC study of Section 2, we observed that the P-ODF method gave better segmentation results than the spectral flux for sarod *gats*, while the reverse was true for sitar *gats*. Table 2 shows the corresponding segmentation performance for the sarod (1-3) and sitar (4-6) *gats*. We observe a nearly 100% boundary detection rate with a few false detections in each concert. The false alarms were found to be triggered by instances of tabla improvisation (change in stroke pattern) without a change in the metric tempo or basic *theka*.

5. CONCLUSION

Motivated by a compelling visual depiction of the rhythmic structure of a Hindustani classical sitar concert [10], we set about an effort to reproduce automatically, with MIR methods, the manual annotation created by expert musicians. A novel onset detection function that exploited the stroke characteristics of the melodic and percussive instrument, and additionally discriminated the two, proved effective in obtaining rhythm representations that separately captured the structural contributions of the tabla and the sitar/sarod. Tempo detection on the separate rhythm vectors provided estimates of the metric tempo and rhythmic density of the sitar/sarod. Segmentation using an SDM on the rhythm vectors provided section boundary estimates with high accuracy. The system now needs to be tested on a large and diverse database of sitar and sarod concerts. Further, given that the rhythmogram contains more information than we have exploited in the current work, we propose to develop methods for section labeling and other relevant musical descriptors.

Acknowledgement: This work received partial funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement 267583 (CompMusic). Also, part of the work was supported by Bharti Centre for Communication in IIT Bombay.

6. REFERENCES

[1] Geoffroy Peeters. Rhythm Classification Using Spectral Rhythm Patterns. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 644–647, 2005.

[2] Fabien Gouyon, Simon Dixon, Elias Pampalk, and Gerhard Widmer. Evaluating rhythmic descriptors for musical genre classification. In *Proceedings of the AES 25th International Conference*, pages 196–204, 2004.

[3] Klaus Seyerlehner, Gerhard Widmer, and Dominik Schnitzer. From Rhythm Patterns to Perceived Tempo. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 519–524, 2007.

[4] Kristoffer Jensen, Jieping Xu, and Martin Zachariasen. Rhythm-Based Segmentation of Popular Chinese Music. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 374–380, 2005.

[5] Peter Grosche, Meinard Müller, and Frank Kurth. Cyclic tempogram-a mid-level tempo representation for music signals. In *IEEE International Conference on Acoustics Speech and Signal Processing*, pages 5522–5525, 2010.

[6] Ajay Srinivasamurthy, André Holzapfel, and Xavier Serra. In search of automatic rhythm analysis methods for Turkish and Indian art music. *Journal of New Music Research*, 43(1):94–114, 2014.

[7] Bonnie C Wade. *Music in India: The classical traditions*, chapter 7: Performance Genres of Hindustani Music. Manohar Publishers, 2001.

[8] Prateek Verma, T. P. Vinutha, Parthe Pandit, and Preeti Rao. Structural segmentation of Hindustani concert audio with posterior features. In *IEEE International Conference on Acoustics Speech and Signal Processing*, pages 136–140, 2015.

[9] Sandeep Bagchee. *Nad: Understanding Raga Music*. Business Publications Inc., India, 1998.

[10] Martin Clayton. *Time in Indian Music: Rhythm, Metre, and Form in North Indian Rag Performance*, chapter 11: A case study in rhythmic analysis. Oxford University Press, UK, 2001.

[11] Geoffroy Peeters. Template-based estimation of time-varying tempo. *EURASIP Journal on Applied Signal Processing*, 2007(1):158–171, 2007.

[12] Juan Pablo Bello, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, and Mark B Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047, 2005.

[13] Dik J Hermes. Vowel-onset detection. *Journal of the Acoustical Society of America*, 87(2):866–873, 1990.

[14] Simon Dixon. Onset detection revisited. In *Proceedings of the 9th International Conference on Digital Audio Effects*, volume 120, pages 133–137, 2006.

[15] Derry FitzGerald. Vocal separation using nearest neighbours and median filtering. In *IET Irish Signals and Systems Conference (ISSC 2012)*, pages 1–5, 2012.

[16] Kristoffer Jensen. Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal on Advances in Signal Processing*, 2006(1):1–11, 2006.

[17] Martin Clayton. Two gat forms for the sitār: a case study in the rhythmic analysis of north indian music. *British Journal of Ethnomusicology*, 2(1):75–98, 1993.

[18] Chris Cannam, Christian Landone, and Mark Sandler. Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1467–1468, 2010.

[19] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *IEEE International Conference on Multimedia and Expo*, volume 1, pages 452–455, 2000.

TEMPLATE-BASED VIBRATO ANALYSIS OF MUSIC SIGNALS

Jonathan Driedger[1], **Stefan Balke**[1], **Sebastian Ewert**[2], **Meinard Müller**[1]
[1]International Audio Laboratories Erlangen, Germany
[2]Queen Mary University of London
{jonathan.driedger,stefan.balke,meinard.mueller}@audiolabs-erlangen.de

ABSTRACT

The automated analysis of vibrato in complex music signals is a highly challenging task. A common strategy is to proceed in a two-step fashion. First, a fundamental frequency (F0) trajectory for the musical voice that is likely to exhibit vibrato is estimated. In a second step, the trajectory is then analyzed with respect to periodic frequency modulations. As a major drawback, however, such a method cannot recover from errors made in the inherently difficult first step, which severely limits the performance during the second step. In this work, we present a novel vibrato analysis approach that avoids the first error-prone F0-estimation step. Our core idea is to perform the analysis directly on a signal's spectrogram representation where vibrato is evident in the form of characteristic spectro-temporal patterns. We detect and parameterize these patterns by locally comparing the spectrogram with a predefined set of vibrato templates. Our systematic experiments indicate that this approach is more robust than F0-based strategies.

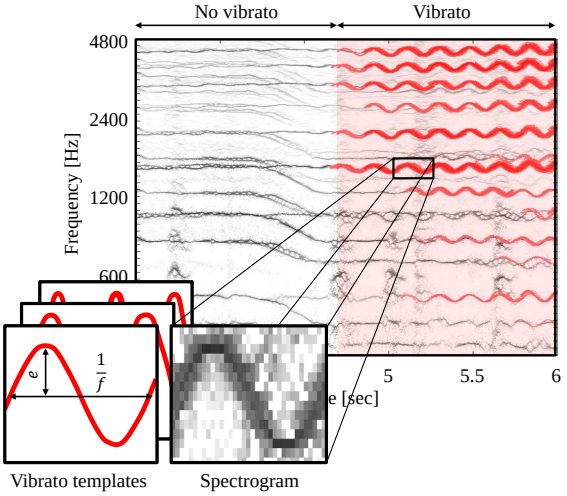


Figure 1. Template-based vibrato analysis. A matching vibrato template lets us infer the rate f and extent e of vibrato present in the music signal.

1. INTRODUCTION

The human voice and other instruments often reveal characteristic spectro-temporal patterns that are the result of specific articulation techniques. For example, vibrato is a musical effect that is frequently used by musicians to make their performance more expressive. Although a clear definition of vibrato does not exist [20], it can broadly be described as a musical voice's "periodic oscillation in pitch" [16]. It is commonly parameterized by its *rate* (the modulation frequency given in Hertz) and its *extent* (the modulation's amplitude given in cents [1]). These parameters have been studied extensively from musicological and psychological perspectives, often in a cumbersome process of manually annotating spectral representations of monophonic music signals, see for example [5, 10, 18, 20, 22].

To approach the topic from a computational perspective, the signal processing community has put considerable research efforts into developing automated vibrato analysis methods for monophonic, as well as for more complex music signals with multiple sound sources. While some applications implicitly exploit spectro-temporal characteristics of vibrato to approach higher-level tasks such as harmonic-percussive decomposition [9], singing voice detection [6], or singing voice separation [21], there also exist methods for explicitly detecting and parameterizing vibrato components in a given music signal. A common approach is to perform the vibrato analysis in two consecutive steps. In the first step, a fundamental frequency trajectory (F0-trajectory) is estimated for the musical voice that is most likely to exhibit vibrato. This trajectory is then analyzed in the second step to detect and parameterize periodic modulation patterns, see for example [4, 8, 12–14, 23]. However, computing F0-trajectories for complex signals with multiple instruments is a highly non-trivial and error-prone task by itself [15]. Therefore, a trajectory estimated in the first step may not appropriately reflect the relevant modulation patterns. This in turn renders the vibrato detection and parametrization in the second step problematic, if not impossible.

To avoid the error-prone F0-estimation step, in this work we propose a novel approach for automatically analyzing vibrato components in complex music signals. Our core idea is to detect spectro-temporal vibrato patterns di-

[1] A *cent* is a logarithmic frequency unit. A musical semitone is subdivided into 100 cents.

© Jonathan Driedger, Stefan Balke, Sebastian Ewert, Meinard Müller. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jonathan Driedger, Stefan Balke, Sebastian Ewert, Meinard Müller. "Template-Based Vibrato Analysis of Music Signals", 17th International Society for Music Information Retrieval Conference, 2016.

rectly in a music signal's spectrogram by locally comparing this representation with a set of predefined vibrato templates [2] that reflect different vibrato rates and extents. The measured similarity yields a novel mid-level feature representation—a *vibrato salience spectrogram*—in which spectro-temporal vibrato patterns are enhanced while other structures are suppressed. Figure 1 illustrates this idea, showing three different vibrato templates as well as a spectrogram representation of a choir with a lead singer who starts to sing with strong vibrato in the excerpt's second half. Time-frequency bins where one of the templates is locally similar to the spectrogram, thus yielding a high vibrato salience, are indicated in red. As we can see, these time-frequency bins temporally coincide with the annotated vibrato passage at the top of Figure 1. Additionally, a high vibrato salience does not only indicate the presence of vibrato in the music signal, but also reveals the vibrato's rate and extent encoded in the similarity maximizing template.

The remainder of this paper is structured as follows. In Section 2 we describe our template-based vibrato analysis approach in detail. In Section 3, we evaluate the performance of our proposed method, both by means of a quantitative evaluation on a novel dataset as well as by discussing illustrative examples. Finally, in Section 4, we conclude with an indication of possible future research directions. Note that this paper has an accompanying website at [2] where one can find all audio examples and annotations used in this paper.

2. TEMPLATE-BASED VIBRATO ANALYSIS

In this section, we describe our proposed template-based vibrato analysis approach. We discuss relevant spectrogram representations (Section 2.1) and describe how the vibrato templates are modeled (Section 2.2). Both our choice of spectrogram representation and the vibrato template's design are motivated by the correlation-like similarity measure that we use to locally compare the templates with the spectrogram. We then introduce the derivation of the vibrato salience spectrogram (Section 2.3) and comment on our approach's computational complexity (Section 2.4). As a running example, we use the choir signal from Figure 1.

2.1 Spectral Representation

Given a discrete music signal $x : \mathbb{Z} \to \mathbb{R}$, we first compute the *short-time Fourier transform* (STFT) $X : \mathbb{Z} \times \mathbb{Z} \to \mathbb{C}$ of x by

$$X(m,k) = \sum_{r \in \mathbb{Z}} w(r) \cdot x(r+mH) \cdot \exp(-2\pi i k r/N) \ , \quad (1)$$

where m is the frame index, k is the frequency index, N is the frame length, w is a window function, and H is the hopsize (w.l.o.g. we assume $m, k \in \mathbb{Z}$). Figure 2a shows an excerpt of our example signal's *magnitude spectrogram* $|X|$ where one can clearly see wave-like vibrato patterns in the lead singer's fundamental frequency and its overtones. However, due to the STFT's linear frequency sampling, the vibrato patterns' amplitudes increase with higher overtones.

[2] Note that this approach is conceptually similar to the Hough transform [3], a mathematical tool known from image processing for the detection of parameterized shapes in binary images. However, the Hough transform is known to be very sensitive to noise and therefore not suitable for detecting vibrato patterns in spectrograms that are commonly rather noisy.

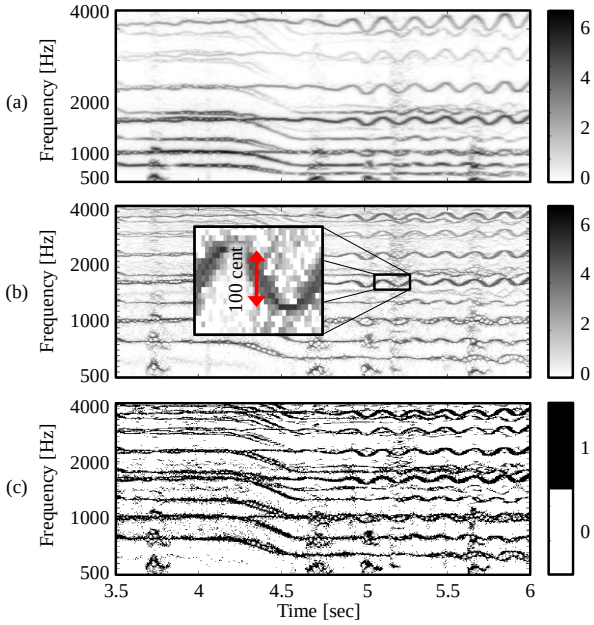


Figure 2. Spectrogram representations of the input signal x. **(a):** Magnitude spectrogram. **(b):** Log-frequency spectrogram. **(c):** Binarized log-frequency spectrogram Y.

In the context of our template-based analysis it is desirable that vibrato patterns stemming from the same frequency modulated tone have the same amplitude that directly reflects the vibrato's extent. We therefore compute a *log-frequency spectrogram* from the STFT X, using a phase vocoder-based reassignment approach as discussed in [7, Chapter 8] or [14]. In this representation, which can be seen in Figure 2b, frequency bands are spaced logarithmically and have a constant logarithmic bandwidth specified in cents. This ensures the desired property in this spectrogram representation.

In a last step, we normalize the spectrogram in order to achieve two goals. First, we aim to make the representation independent of the signal's volume such that we can also detect vibrato in quiet signal passages. Second, when locally comparing our vibrato templates with the representation, the resulting similarity measure should yield values in a fixed range. A method that showed to be simple and effective to achieve both goals is spectrogram binarization, where we set the ten percent highest values of each frame in the log-frequency spectrogram to one and all remaining values to zero. This yields a *binarized log-frequency spectrogram* $Y : \mathbb{Z} \times \mathbb{Z} \to \{0, 1\}$, see Figure 2c. In our experiments, we choose parameters such that Y has a time res-

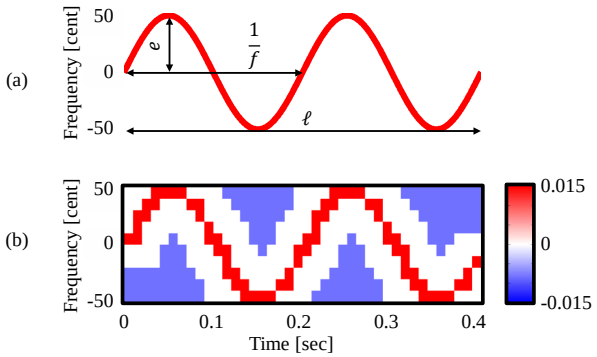


Figure 3. Generation of a vibrato template T with a vibrato rate $f = 5$ Hertz, extent $e = 50$ cent, and a duration of $\ell = 0.4$ seconds. **(a):** Sinusoidal vibrato trajectory s. **(b):** Vibrato template T.

olution of roughly 150 frames per second and a frequency resolution of ten bands per semitone.

2.2 Vibrato Templates

Next, we introduce a set $\mathcal{T}$ of templates that reflect spectro-temporal vibrato patterns as expected in Y. Let us model such a template $T \in \mathcal{T}$ for vibrato having a rate of f Hertz, an extent of e cents, and a duration of at least ℓ seconds. When locally comparing the template T with Y, one should obtain high similarity values when T is aligned with a matching spectro-temporal vibrato pattern in Y and low values otherwise. The idea is therefore to have a positive portion in T that reflects the spectro-temporal vibrato pattern as well as a negative portion that prevents the template from correlating well with regions in Y that are homogeneously equal to one.

Assuming a sinusoidal vibrato, we can describe the vibrato's trajectory (up to phase) by

$$s(t) = e \, \sin(2\pi f t) \, , \tag{2}$$

$t \in [0, \ell]$. Figure 3a shows such a trajectory for $f = 5$ Hertz, $e = 50$ cent, and $\ell = 0.4$ seconds. The trajectory is then discretized such that its time- and frequency resolution matches the binarized log-frequency spectrogram. Time-frequency bins that are close to s are assigned with positive values, while bins having a certain distance from s get negative values. To allow for some tolerance of the width of vibrato patterns in Y, the remaining time-frequency bins are defined to be zero. Finally, positive and negative entries in T are normalized to sum up to one and minus one, respectively, see Figure 3b.

2.3 Vibrato Salience

In order to locate and parameterize vibrato structures in the binarized log-frequency spectrogram Y, we aim to compute a *vibrato salience spectrogram* S—a kind of mid-level feature representation—in which vibrato structures are enhanced while other kinds of structures are suppressed. To this end, we define the vibrato salience spectrogram S_T for a single vibrato template $T : [0 : A-1] \times [0 : B-1] \to \mathbb{R}$, $A, B \in \mathbb{N}$. The computation process is illustrated in Figure 4a. Let $\mathcal{I}$ be the set of all index pairs $(a, b) \in [0 : A-1] \times [0 : B-1]$ such that $T(a, b)$ is positive (the indices of all red entries in Figure 3b). Furthermore, let

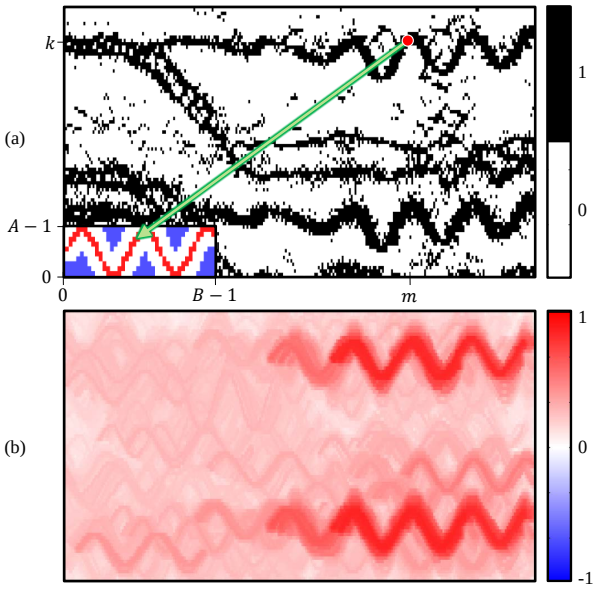


Figure 4. Vibrato salience spectrogram computation. **(a):** Process to compute S_T. The similarity-maximizing shift (μ, κ) that maps (m, k) onto an index pair in $\mathcal{I}$ is indicated by a green arrow. **(b):** Vibrato salience spectrogram S.

$$Y^{(\mu,\kappa)}(m, k) = Y(m - \mu, k - \kappa) \, , \tag{3}$$

$\mu, \kappa \in \mathbb{Z}$, denote a version of Y that is shifted by μ and κ indices in time- and frequency direction, respectively. Intuitively, the vibrato salience $S_T(m, k)$ should be high if $Y(m, k)$ is part of a spectro-temporal vibrato pattern as reflected by T. To this end, we verify if there is a shift (μ, κ) that aligns $Y(m, k)$ (red dot in Figure 4a) with one of the positive entries in the vibrato template T such that T and $Y^{(\mu,\kappa)}$ are similar (the optimal shift for our example in Figure 4a is indicated by a green arrow). To compute $S_T(m, k)$, we therefore maximize the correlation-like similarity measure

$$c(T, Y) = \sum_{a=0}^{A-1} \sum_{b=0}^{B-1} T(a, b) Y(a, b) \tag{4}$$

over all shifts (μ, κ) that map (m, k) onto one of the index pairs in $\mathcal{I}$:

$$S_T(m, k) = \max_{\{(\mu,\kappa):(m,k)-(\mu,\kappa)\in\mathcal{I}\}} c(T, Y^{(\mu,\kappa)}) \, . \tag{5}$$

The full vibrato salience spectrogram can then be computed by maximizing over all vibrato templates $T \in \mathcal{T}$:

$$S(m, k) = \max_{T\in\mathcal{T}} S_T(m, k) \, . \tag{6}$$

			-0 dB		-5 dB		-10 dB		
Item name	L_{x}	L_{vib}	TB-A	F0-M	TB-A	F0-M	TB-A	F0-M	BL
Sound On Sound Demo—Mystery	9.79	1.78	0.83	0.93	0.84	0.86	0.73	0.30	0.31
Giselle—You	5.12	2.99	0.91	0.94	0.91	0.88	0.86	0.53	0.73
Leaf—Full	5.36	1.64	0.84	0.86	0.74	0.29	0.82	0.00	0.46
Phre The Eon—Everybody is Falling Apart	2.47	0.47	0.98	0.97	0.96	0.97	0.95	0.00	0.32
Secretariat—Borderline	7.69	1.98	0.79	0.69	0.73	0.76	0.79	0.00	0.41
Sunshine Garcia Band—For I Am The Moon	12.54	3.36	0.63	0.73	0.67	0.62	0.74	0.44	0.42
Angela Thomas Wade—Milk Cow Blues	4.50	2.10	0.44	0.82	0.32	0.63	0.32	0.00	0.63
Triviul—Dorothy	5.22	0.85	0.77	0.88	0.73	0.85	0.65	0.00	0.28
Funny Valentines—Sleigh Ride	7.18	0.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00
∅	6.65	1.69	0.80	0.87	0.77	0.76	0.76	0.25	0.39

Table 1. Quantitative evaluation (F-measure), comparing our proposed template-based detection approach TB-A, F0-based vibrato detection F0-M (manual vibrato selection in F0-trajectories), and a baseline BL. Lengths of the signals (L_x) and accumulated lengths of ground truth vibrato passages (L_{vib}) are given in seconds.

Figure 4b shows the vibrato salience spectrogram S resulting from the binarized log-frequency spectrogram Y shown in Figure 4a. Note that by the vibrato template's design and $Y(m,k) \in \{0,1\}$, one obtains $S(m,k) \in [-1,1]$ for all $m,k \in \mathbb{Z}$. While the vibrato structures present in Y are also clearly visible in S, the horizontal structures as well as the glissando at the excerpt's beginning do not correlate well with the vibrato templates. They are therefore, as intended, suppressed in S.

2.4 Computational Complexity

The vibrato salience spectrogram's derivation as defined in the previous section is a computationally expensive process. When implemented naively, it is necessary to use a quadruply nested loop to iterate over all combinations of time-frequency bins (m,k) in Y, vibrato templates $T \in \mathcal{T}$, index shifts $\{(\mu,\kappa) : (m,k) - (\mu,\kappa) \in \mathcal{I}\}$, and index pairs (a,b) in T. However, note that many computations are redundant and that it is therefore possible to optimize the calculation process, for example by exploiting two-dimensional convolutions. Furthermore, one can speed up the derivation by considering only a limited frequency range in Y as well as by applying further heuristics such as only taking into account vibrato salience values above a threshold $\tau \in [-1,1]$. Although still being computationally demanding, the derivation therefore becomes feasible enough to be used in practice. For example, deriving S for a music signal with a duration of 60 seconds takes our MATLAB implementation roughly 40 seconds on a standard computer.

3. EXPERIMENTS

In this section, we present our experimental results. In Section 3.1, we quantitatively evaluate our proposed approach in the context of a vibrato detection task. Then, in Section 3.2 we demonstrate the method's potential for automatically analyzing vibrato rate and extent. Finally, in Section 3.3, we indicate open challenges and potential solutions.

3.1 Evaluation: Vibrato Detection

In a first experiment, we considered the task of temporally identifying vibrato passages in a music signal. We therefore compiled a dataset of nine items (see Table 1), which are excerpts of music signals from the "Mixing Secrets" multitrack dataset [17]. Each item consists of a monophonic vocal signal x_{voc} and a polyphonic accompaniment signal x_{acc}. Annotations of vibrato passages in the vocal signals were created manually to serve as ground truth for the subsequent evaluation (none of the accompaniment signals x_{acc} has vibrato). To vary the difficulty of the vibrato detection task, we created three different mixes for each of the items—one were x_{voc} and x_{acc} were mixed without modification (-0 dB), one were x_{voc} was attenuated by -5 dB prior to mixing the signals, and a third mix with x_{voc} being attenuated by -10 dB.

To construct an automated vibrato detection procedure based on our proposed template-based analysis approach, we first computed vibrato salience spectrograms S for all of the resulting 27 mix signals. Since only high vibrato salience values in S are likely to indicate the presence of spectro-temporal vibrato patterns, we then chose a threshold $\tau \in [-1,1]$. Time instances where the maximal vibrato salience in a frame exceeded τ were then labeled as having vibrato while all other time instances were labeled as having no vibrato. For this experiment we used a set $\mathcal{T}$ of 30 templates, reflecting vibrato rates from five to seven Hertz in steps of 0.5 Hertz, as well as extents from 50 to 100 cents in steps of 10 cents. These parameters were chosen particularly to detect the vibrato in singing voice as these are typical vibrato rates and extents for human singing, see [10, 11]. All templates had a length corresponding to $\ell = 0.4$ seconds. The threshold τ was experimentally set to $\tau = 0.55$, yielding good vibrato detection results for all items in the dataset.

One of this experiment's main objectives was to compare our template-based method's performance with F0-based strategies as discussed in Section 1. To emulate such an approach, we used MELODIA [14]—a state-of-the-art algorithm for estimating F0-trajectories of predominant musical voices in complex music signals—to esti-

mate trajectories for all mix signals. Instead of automatically analyzing the extracted trajectories in a second step, we then manually inspected them for passages that reflect vibrato. This was done to obtain an upper bound on the performance an automated procedure could achieve in this second step when detecting vibrato solely based on the estimated F0-trajectory.

We then computed precision (P), recall (R), and F-measure (F) for the detection results of our automated template-based procedure (`TB-A`), for the procedure based on the manually inspected F0-trajectory (`F0-M`), as well as for a baseline approach that simply labels every time instance as having vibrato (`BL`):

$$\mathrm{P} = \frac{\mathrm{TP}+\epsilon}{\mathrm{TP}+\mathrm{FP}+\epsilon}, \mathrm{R} = \frac{\mathrm{TP}+\epsilon}{\mathrm{TP}+\mathrm{FN}+\epsilon}, \mathrm{F} = \frac{2\mathrm{PR}}{\mathrm{P}+\mathrm{R}} . \quad (7)$$

Here, TP is the number of true positives, FP the number of false positives, FN the number of false negatives, and $\epsilon > 0 \in \mathbb{R}$ is some small number to prevent division by zero. Note that all music signals and annotations used in the experiment can be found at this paper's accompanying website [2].

The evaluation's results are summarized in Table 1 which shows for each item its name, the music signal's length, the accumulated duration of vibrato in this signal, as well as the F-measures of `TB-A` and `F0-M` for the three different mixes (-0 dB, -5 dB, and -10 dB). The F-measure for the baseline `BL` is indicated in the last column and the table's last row indicates mean values. Here we can observe a clear trend. For mixes where x_{voc} was not attenuated (-0 dB), both `TB-A` and `F0-M` yield average F-measures ($\mathrm{F} = 0.80$ and $\mathrm{F} = 0.87$) clearly above the baseline `BL` ($\mathrm{F} = 0.39$). For this mixing condition, `F0-M` outperforms our template-based approach. However, recall that `F0-M` constitutes an upper bound on the performance of F0-based vibrato detection approaches. Automating the vibrato detection step may therefore result in lower scores.

For mixes where x_{voc} was attenuated by -5 dB, the average F-measure of `TB-A` only slightly decreases to $\mathrm{F} = 0.77$, while the performance of `F0-M` drops to $\mathrm{F} = 0.76$. This tendency becomes even more extreme when considering vocal signals attenuated by -10 dB where `TB-A`'s performance stays almost constant ($\mathrm{F} = 0.76$) while `F0-M`'s average F-measure goes down to $\mathrm{F} = 0.25$, many of the individual items scoring F-measures of zero.

The reason for this trend becomes obvious when investigating individual items. Figure 5 depicts the vibrato detection results of both `TB-A` and `F0-M` in all mixing conditions for the item *Leaf—Full*. In the condition -0 dB, the results of `TB-A` (Figure 5a) and `F0-M` (Figure 5b) coincide well with the ground truth (Figure 5c), leading to high F-measures ($\mathrm{F} = 0.84$ and $\mathrm{F} = 0.86$). Here, our template-based analysis approach detects most of the spectro-temporal vibrato patterns in the signal's spectrogram (time-frequency bins where the vibrato salience exceeds the threshold τ are indicated in red in Figure 5a). `F0-M` also achieves a good result since the F0-trajectory extracted by MELODIA (indicated in blue in Figure 5b) captures the singing voice's fundamental frequency well

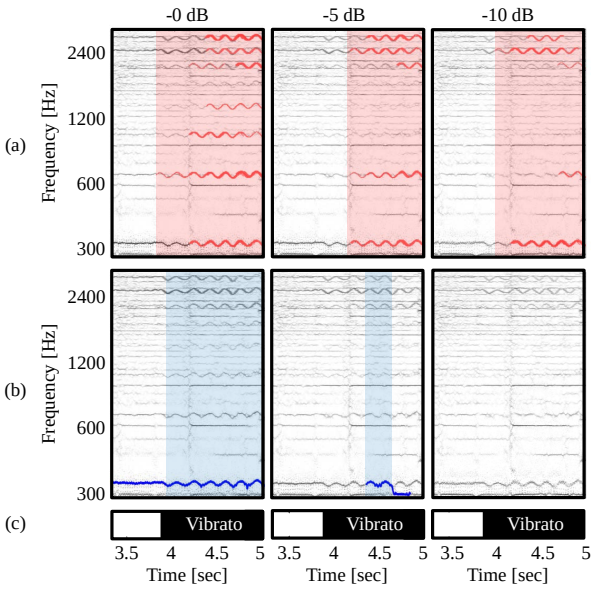


Figure 5. Comparison of `TB-A` and `F0-M` for the item *Leaf—Full*. **(a):** `TB-A`. Automatically derived vibrato passages are indicated in red. **(b):** `F0-M`. Manually annotated vibrato passages in the trajectory are indicated in blue. **(c):** Ground truth annotation.

in this mix. However, this changes when attenuating the vocal signal by -5 dB. While `TB-A` still identifies many vibrato patterns, therefore detecting the vibrato present in the mix ($\mathrm{F} = 0.74$), the F0-estimation becomes problematic and MELODIA retrieves only a small segment of the singing voice's F0-trajectory correctly, leading to a poor vibrato detection ($\mathrm{F} = 0.29$). When attenuating x_{voc} by -10 dB, the F0-trajectory's estimation fails completely ($\mathrm{F} = 0.00$) since MELODIA's assumption of a predominant melodic voice is violated. On the other hand, our proposed detection procedure is capable of detecting the vibrato in the mix.

As a final remark, note that our proposed approach also succeeds to recognize that the item *Funny Valentines—Sleigh Ride* does not contain any vibrato at all.

3.2 Evaluation: Vibrato Analysis

As we have seen in the previous section, the vibrato salience spectrogram S can be used to determine *when* vibrato is present in a music signal. Additionally, when computing S, we also implicitly obtain information about the vibrato's parameters. The rate and extent of vibrato present in the music signal are encoded by the similarity maximizing vibrato templates T in Equation (6). In Figure 6a, we see the log-frequency spectrogram of a mixture of piano music (no vibrato) and three consecutive artificial vibrato tones. The tones have vibrato rates of seven, five, and ten Hertz and extents of 40, 200, and 70 cents, respectively. Time-frequency bins where the vibrato salience exceeds τ are indicated in red. Note that for this experiment we used a much larger template set $\mathcal{T}$, consisting of 285 tem-

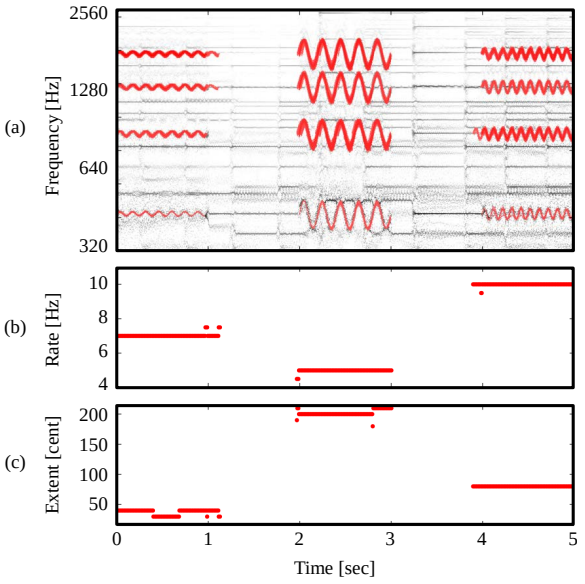


Figure 6. Vibrato rate and extent analysis. **(a):** Log-frequency spectrogram. Time-frequency bins (m, k) with $S(m, k) > \tau$ are indicated in red. **(b)/(c):** Vibrato rate and extent of the template T with the highest vibrato salience per frame.

plates that reflected vibrato rates from four to eleven Hertz in steps of 0.5 Hertz, as well as extents from 30 to 210 cents in steps of 10 cents. Figures 6b/c indicate the vibrato rate and extent of the vibrato template T that maximized the vibrato salience per frame. The two plots correctly reflect the tones' vibrato rates and extents, while showing only a few outliers. Note that values in the plots are quantized since our approach can only give estimates for rates and extents as they are reflected by one of the templates in $\mathcal{T}$. This kind of vibrato analysis could be helpful in scenarios like informed instrument identification when it is known that different instruments in a music signal perform with different vibrato rates or extents.

3.3 Challenges

In general, our proposed procedure yields useful analysis results for the music examples discussed in the previous sections. We now want to discuss a few difficult examples.

One potential source for incorrect analysis results are false positives as visualized in Figure 7a, which shows a log-frequency spectrogram excerpt of *Sunshine Garcia Band—For I Am The Moon* from our dataset. In this excerpt, one of our vibrato templates T is similar enough (with respect to our similarity measure) to a non-vibrato spectro-temporal pattern to yield vibrato salience values above the threshold τ. This could cause incorrect vibrato detection results or meaningless vibrato parametrizations. However, we experienced such spurious template matches to often occur in an isolated fashion. Here, one could exploit additional cues such as multiple template matches at the same time instance due to overtone structures of instruments to reinforce the vibrato analysis' results.

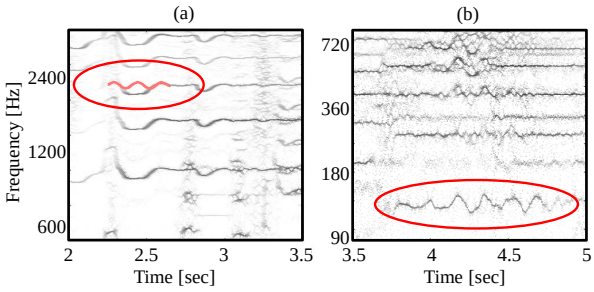


Figure 7. Error sources for our template-based vibrato analysis. **(a):** Spurious template matches. **(b):** Vibrato does not have a sinusoidal form.

The opposite situation is visualized in Figure 7b. It shows a log-frequency spectrogram excerpt of "Gute Nacht", a song from Schubert's "Winterreise" for piano and tenor. In this excerpt, the singer sings a long note with strong vibrato. However, although there is a template reflecting an appropriate vibrato rate and extent in our template set $\mathcal{T}$, the vibrato is not detected by our procedure. This is the case since by our vibrato template's design—as described in Section 2.2—we generally assumed vibrato to have a sinusoidal spectro-temporal structure. This assumption is violated in the shown vibrato pattern. However, our approach is conceptually not limited to sinusoidal vibrato templates and one could further improve the templates' design in order to also capture these kind of vibrato patterns.

4. CONCLUSIONS AND FUTURE WORK

In this paper we presented a novel approach for analyzing vibrato in complex music signals. By locally comparing a signal's spectrogram with a set of predefined vibrato templates, we derived a vibrato salience spectrogram—a kind of mid-level feature representation—in order to locate and parameterize spectro-temporal vibrato patterns. Our approach has the advantage that the analysis does not rely on the estimation of a (possibly erroneous) F0-trajectory. Experiments indicated that our proposed procedure allows for a more robust vibrato detection than F0-based approaches, in particular for complex music signals.

In future work we would like to further explore the use of vibrato templates in various application scenarios. For example, deriving spectral masks from the vibrato salience spectrogram S could open up novel ways of decomposing a music signal into vibrato and non-vibrato components. Furthermore, we believe that the use of vibrato templates could be beneficial for tasks like F0-tracking [14, 19] or performance analysis [1].

Acknowledgments:
This work has been supported by the German Research Foundation (DFG MU 2686/6-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institut für Integrierte Schaltungen IIS. Sebastian Ewert is funded by the EPSRC (EP/L019981/1).

5. REFERENCES

[1] Jakob Abeßer, Hanna M. Lukashevich, and Gerald Schuller. Feature-based extraction of plucking and expression styles of the electric bass guitar. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2290–2293, Dallas, Texas, USA, March 2010.

[2] Jonathan Driedger, Stefan Balke, Sebastian Ewert, and Meinard Müller. Accompanying website: Towards template-based vibrato analysis in complex music signals. `http://www.audiolabs-erlangen.de/resources/MIR/2016-ISMIR-Vibrato/`.

[3] K. Glossop, P. J. G. Lisboa, P. C. Russell, A. Siddans, and G. R. Jones. An implementation of the hough transformation for the identification and labelling of fixed period sinusoidal curves. *Computer Vision and Image Understanding*, 74(1):96–100, 1999.

[4] Perfecto Herrera and Jordi Bonada. Vibrato extraction and parameterization in the spectral modeling synthesis framework. In *Digital Audio Effects Workshop (DAFX98)*, Barcelona, Spain, November 1998.

[5] Yoshiyuki Horii. Acoustic analysis of vocal vibrato: A theoretical interpretation of data. *Journal of Voice*, 3(1):36–43, 1989.

[6] Bernhard Lehner, Gerhard Widmer, and Reinhard Sonnleitner. On the reduction of false positives in singing voice detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 7480–7484, Florence, Italy, 2014.

[7] Meinard Müller. *Fundamentals of Music Processing*. Springer Verlag, 2015.

[8] Hee-Suk Pang. On the use of the maximum likelihood estimation for analysis of vibrato tones. *Applied Acoustics*, 65(1):101–107, 2004.

[9] Jeongsoo Park and Kyogu Lee. Harmonic-percussive source separation using harmonicity and sparsity constraints. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 148–154, Málaga, Spain, 2015.

[10] Eric Prame. Measurements of the vibrato rate of ten singers. *The Journal of the Acoustical Society of America (JASA)*, 96(4):1979–1984, 1994.

[11] Eric Prame. Vibrato extent and intonation in professional western lyric singing. *The Journal of the Acoustical Society of America (JASA)*, 102(1):616–621, 1997.

[12] Lise Regnier and Geoffroy Peeters. Singing voice detection in music tracks using direct voice vibrato detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1685–1688, Taipei, Taiwan, 2009.

[13] S. Rossignol, P. Depalle, J. Soumagne, X. Rodet, and J.-L. Collette. Vibrato: detection, estimation, extraction, modification. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, Trondheim, Norway, December 1999.

[14] Justin Salamon and Emilia Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[15] Justin Salamon, Emilia Gómez, Daniel P. W. Ellis, and Gaël Richard. Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, 2014.

[16] Carl E. Seashore. The natural history of the vibrato. 17(12):623–626, 1931.

[17] Mike Senior. Mixing secrets for the small studio—additional resources. `www.cambridge-mt.com/ms-mtk.htm`. Web resource, last consulted in January 2016.

[18] T. Shipp, R. Leanderson, and J. Sundberg. Some acoustic characteristics of vocal vibrato. *Journal of Research in Singing*, IV(1):18–25, 1980.

[19] Fabian-Robert Stöter, Nils Werner, Stefan Bayer, and Bernd Edler. Refining fundamental frequency estimates using time warping. In *Proceedings of EUSIPCO 2015*, Nice, France, September 2015.

[20] Johan Sundberg. Acoustic and psychoacoustic aspects of vocal vibrato. *STL-QPSR*, 35(2-3):45–68, 1994.

[21] Hideyuki Tachibana, Nobutaka Ono, and Shigeki Sagayama. Singing voice enhancement in monaural music signals based on two-stage harmonic/percussive sound separation on multiple resolution spectrograms. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):228–237, January 2014.

[22] Renee Timmers and Peter Desain. Vibrato: Questions and answers from musicians and science. In *Proceedings of the International Conference on Music Perception and Cognition (ICMPC)*, volume 2, 2000.

[23] Luwei Yang, Elaine Chew, and Khalid Z. Rajab. Vibrato performance style: A case study comparing erhu and violin. In *Proceedings of the International Conference on Computer Music Modeling and Retrieval (CMMR)*, 2013.

TOWARDS EVALUATING MULTIPLE PREDOMINANT MELODY ANNOTATIONS IN JAZZ RECORDINGS

Stefan Balke[1] **Jonathan Driedger**[1] **Jakob Abeßer**[2]
Christian Dittmar[1] **Meinard Müller**[1]
[1] International Audio Laboratories Erlangen, Germany
[2] Semantic Music Technologies Group, Fraunhofer IDMT, Ilmenau, Germany
stefan.balke@audiolabs-erlangen.de

ABSTRACT

Melody estimation algorithms are typically evaluated by separately assessing the task of voice activity detection and fundamental frequency estimation. For both subtasks, computed results are typically compared to a single human reference annotation. This is problematic since different human experts may differ in how they specify a predominant melody, thus leading to a pool of equally valid reference annotations. In this paper, we address the problem of evaluating melody extraction algorithms within a jazz music scenario. Using four human and two automatically computed annotations, we discuss the limitations of standard evaluation measures and introduce an adaptation of Fleiss' kappa that can better account for multiple reference annotations. Our experiments not only highlight the behavior of the different evaluation measures, but also give deeper insights into the melody extraction task.

1. INTRODUCTION

Predominant melody extraction is the task of estimating an audio recording's fundamental frequency trajectory values (F0) over time which correspond to the melody. For example in classical jazz recordings, the predominant melody is typically played by a soloist who is accompanied by a rhythm section (e. g., consisting of piano, drums, and bass). When estimating the soloist's F0-trajectory by means of an automated method, one needs to deal with two issues: First, to determine the time instances when the soloist is active. Second, to estimate the course of the soloist's F0 values at active time instances.

A common way to evaluate such an automated approach—as also used in the Music Information Retrieval Evaluation eXchange (MIREX) [5]—is to split the evaluation into the two subtasks of activity detection and F0 estimation. These subtasks are then evaluated by comparing the computed results to a single manually created reference annotation. Such an evaluation, however, is problematic since it assumes the existence of a single ground-truth. In practice, different humans may annotate the same recording in different ways thus leading to a low inter-annotator agreement. Possible reasons are the lack of an exact task specification, the differences in the annotators' experiences, or the usage of different annotation tools [21, 22]. Figure 1 exemplarily illustrates such variations on the basis of three annotations $A_1, ..., A_3$ of the same audio recording, where a soloist plays three consecutive notes. A first observation is that A_1 and A_2 have a fine frequency resolution which can capture fluctuations over time (e. g., vibrato effects). In contrast, A_3 is specified on the basis of semitones which is common when considering tasks such as music transcription. Furthermore, one can see that note onsets, note transitions, and durations are annotated inconsistently. Reasons for this might be differences in annotators' familiarity with a given instrument, genre, or a particular playing style. In particular, annotation deviations are likely to occur when notes are connected by slurs or glissandi.

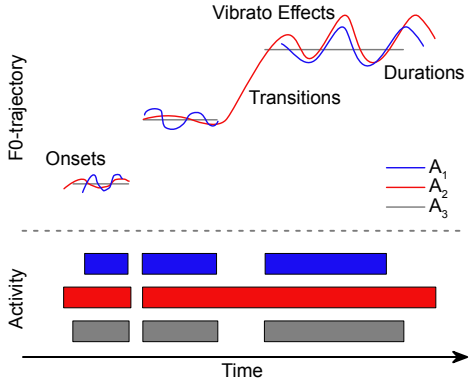


Figure 1. Illustration of different annotations and possible disagreements. A_1 and A_2 are based on a fine frequency resolution. Annotation A_3 is based on a coarser grid of musical pitches.

Inter-annotator disagreement is a generally known problem and has previously been discussed in the contexts of audio music similarity [8, 10], music structure analysis [16, 17, 23], and melody extraction [3]. In general, a

© Stefan Balke, Jakob Abeßer, Jonathan Driedger, Christian Dittmar, Meinard Müller. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Stefan Balke, Jakob Abeßer, Jonathan Driedger, Christian Dittmar, Meinard Müller. "Towards evaluating multiple predominant melody annotations in jazz recordings", 17th International Society for Music Information Retrieval Conference, 2016.

SoloID	Performer	Title	Instr.	Dur.
Bech-ST	Sidney Bechet	Summertime	Sopr. Sax	197
Brow-JO	Clifford Brown	Jordu	Trumpet	118
Brow-JS	Clifford Brown	Joy Spring	Trumpet	100
Brow-SD	Clifford Brown	Sandu	Trumpet	048
Colt-BT	John Coltrane	Blue Train	Ten. Sax	168
Full-BT	Curtis Fuller	Blue Train	Trombone	112
Getz-IP	Stan Getz	The Girl from Ipan.	Ten. Sax	081
Shor-FP	Wayne Shorter	Footprints	Ten. Sax	139

Table 1. List of solo excerpts taken from the WJD. The table indicates the performing artist, the title, the solo instrument, and the duration of the solo (given in seconds).

single reference annotation can only reflect a subset of the musically or perceptually valid interpretations for a given music recording, thus rendering the common practice of evaluating against a single annotation questionable.

The contributions of this paper are as follows. First, we report on experiments, where several humans annotate the predominant F0-trajectory for eight jazz recordings. These human annotations are then compared with computed annotations obtained by automated procedures (MELODIA [20] and pYIN [13]) (Section 2). In particular, we consider the scenario of soloist activity detection for jazz recordings (Section 3.1). Afterwards, we adapt and apply an existing measure (Fleiss' Kappa [7]) to our scenario which can account for jointly evaluating multiple annotations (Section 3.2). Note that this paper has an accompanying website at [1] where one can find the annotations which we use in the experiments.

2. EXPERIMENTAL SETUP

In this work, we use a selection of eight jazz recordings from the *Weimar Jazz Database* (WJD) [9, 18]. For each of these eight recordings (see Table 1), we have a pool of seven annotations $\mathcal{A} = \{A_1, \ldots, A_7\}$ which all represent different estimates of the predominant solo instruments' F0-trajectories. In the following, we model an annotation as a discrete-time function $A : [1 : N] \to \mathbb{R} \cup \{*\}$ which assigns to each time index $n \in [1 : N]$ either the solo's F0 at that time instance (given in Hertz), or the symbol '$*$'. The meaning of $A(n) = *$ is that the soloist is inactive at that time instance.

In Table 2, we list the seven annotations. For this work, we manually created three annotations $A_1, \ldots, A_3$ by using a custom graphical user interface as shown in Figure 2 (see also [6]). In addition to standard audio player functionalities, the interface's central element is a salience spectrogram [20]—an enhanced time-frequency representation with a logarithmically-spaced frequency axis. An annotator can indicate the approximate location of F0-trajectories in the salience spectrogram by drawing *constraint regions* (blue rectangles). The tool then automatically uses techniques based on *dynamic programming* [15] to find a plausible trajectory through the specified region. The annotator can then check the annotation by listening to the solo recording, along with a synchronized sonification of the F0-trajectory.

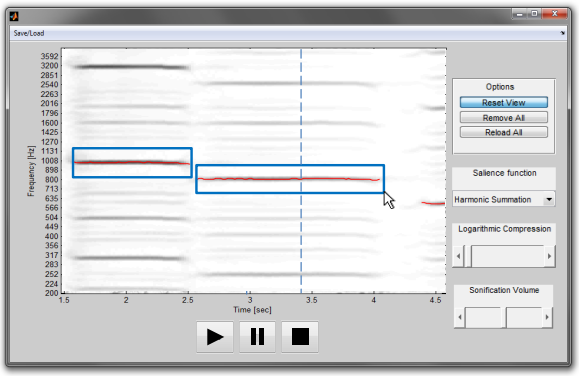


Figure 2. Screenshot of the tool used for the manual annotation of the F0 trajectories.

Annotation	Description
A_1	Human 1, F0-Annotation-Tool
A_2	Human 2, F0-Annotation-Tool
A_3	Human 3, F0-Annotation-Tool
A_4	Human 4, WJD, Sonic Visualiser
A_5	Computed, MELODIA [2, 20]
A_6	Computed, pYIN [13]
A_7	Baseline, all time instances active at 1 kHz

Table 2. Set $\mathcal{A}$ of all annotations with information about their origins.

In addition to the audio recordings, the WJD also includes manually annotated solo transcriptions on the semitone level. These were created and cross-checked by trained jazz musicians using the *Sonic Visualiser* [4]. We use these solo transcriptions to derive A_4 by interpreting the given musical pitches as F0 values by using the pitches' center frequencies.

A_5 and A_6 are created by means of automated methods. A_5 is extracted by using the MELODIA [20] algorithm as implemented in Essentia [2] using the default settings (sample rate = 22050 Hz, hop size = 3 ms, window size = 46 ms). For obtaining A_6, we use the tool Tony [12] (which is based on the pYIN algorithm [13]) with default settings and without any corrections of the F0-trajectory.

As a final annotation, we also consider a baseline $A_7(n) = 1$ kHz for all $n \in [1 : N]$. Intuitively, this baseline assumes the soloist to be always active. All of these annotations are available on this paper's accompanying website [1].

3. SOLOIST ACTIVITY DETECTION

In this section, we focus on the evaluation of the *soloist activity detection* task. This activity is derived from the annotations of the F0-trajectories $A_1, \ldots, A_7$ by only considering active time instances, i. e., $A(n) \neq *$. Figure 3 shows a typical excerpt from the soloist activity annotations for the recording Brow-JO. Each row of this matrix shows the annotated activity for one of our annotations from Table 2. Black denotes regions where the soloist is annotated as active and white where the soloist is annotated

Ref. \ Est.	A_1	A_2	A_3	A_4	A_5	A_6	A_7	∅
A_1	–	0.93	0.98	0.92	0.74	0.79	1.00	0.89
A_2	0.92	–	0.97	0.92	0.74	0.79	1.00	0.89
A_3	0.84	0.84	–	0.88	0.69	0.74	1.00	0.83
A_4	0.85	0.86	0.94	–	0.70	0.75	1.00	0.85
A_5	0.84	0.84	0.90	0.85	–	0.77	1.00	0.87
A_6	0.75	0.76	0.81	0.77	0.65	–	1.00	0.79
A_7	0.62	0.62	0.71	0.67	0.55	0.65	–	0.64
∅	0.80	0.81	0.89	0.83	0.68	0.75	1.00	0.82

Table 3. Pairwise evaluation: *Voicing Detection* (VD). The values are obtained by calculating the VD for all possible annotation pairs (Table 2) and all solo recordings (Table 1). These values are then aggregated by using the arithmetic mean.

as inactive. Especially note onsets and durations strongly vary among the annotation, see e. g., the different durations of the note event at second 7.8. Furthermore, a missing note event is noticeable in the annotations A_1 and A_6 at second 7.6. At second 8.2, A_6 found an additional note event which is not visible in the other annotations. This example indicates that the inter-annotator agreement may be low. To further understand the inter-annotator agreement in our dataset, we first use standard evaluation measures (e. g., as used by MIREX for the task of *audio melody extraction* [14]) and discuss the results. Afterwards, we introduce Fleiss' Kappa, an evaluation measure known from psychology, which can account for multiple annotations.

3.1 Standard Evaluation Measures

As discussed in the previous section, an estimated annotation A_e is typically evaluated by comparing it to a reference annotation A_r. For the pair (A_r, A_e), one can count the number of time instances that are *true positives* $\#\mathrm{TP}$ (A_r and A_e both label the soloist as being active), the number of *false positives* $\#\mathrm{FP}$ (only A_e labels the soloist as being active), the number of *true negatives* $\#\mathrm{TN}$ (A_r and A_e both label the soloist as being inactive), and the number *false negatives* $\#\mathrm{FN}$ (only A_e labels the soloist as being inactive).

In previous MIREX campaigns, these numbers are used to derive two evaluation measures for the task of activity detection. *Voicing Detection* (VD) is identical to *Recall* and describes the ratio that a time instance which is annotated as being active is truly active according to the reference annotation:

$$\mathrm{VD} = \frac{\#\mathrm{TP}}{\#\mathrm{TP} + \#\mathrm{FN}} \,. \tag{1}$$

The second measure is the *Voicing False Alarm* (VFA) and relates the ratio of time instances which are inactive according to the reference annotation but are estimated as being active:

$$\mathrm{VFA} = \frac{\#\mathrm{FP}}{\#\mathrm{TN} + \#\mathrm{FP}} \,. \tag{2}$$

In the following experiments, we assume that all annotations $A_1, \ldots, A_7 \in \mathcal{A}$ have the same status, i. e., each annotation may be regarded as either reference or estimate. Then, we apply the standard measures in a pairwise fashion. For all pairs $(A_r, A_e) \in \mathcal{A} \times \mathcal{A}$ with $A_r \neq A_e$, we extract VD and VFA (using the `MIR_EVAL` [19] toolbox) for each of the solo recordings listed in Table 1. The mean values over the eight recordings are presented in Table 3 for the VD-measure and in Table 4 for the VFA-measure.

As for the Voicing Detection (Table 3), the values within the human annotators $A_1, \ldots, A_4$ range from 0.84 for the pair (A_3, A_2) to 0.98 for the pair (A_1, A_3). This high variation in VD already shows that the inter-annotator disagreement even within the human annotators is substantial. By taking the human annotators as reference to evaluate the automatic approach A_5, the VD lies in the range of 0.69 for (A_3, A_5) to 0.74 for (A_2, A_5). Analogously, for A_6, we observe values from 0.74 for (A_3, A_6) to 0.79 for (A_1, A_6).

As for the Voicing False Alarm (see Table 4), the values among the human annotations range from 0.05 for (A_3, A_1) to 0.30 for (A_1, A_3). Especially annotation A_3 deviates from the other human annotations, resulting in a very high VFA (having many time instances being set as active).

In conclusion, depending on which human annotation we take as the reference, the evaluated performances of the automated methods vary substantially. Having multiple potential reference annotations, the standard measures

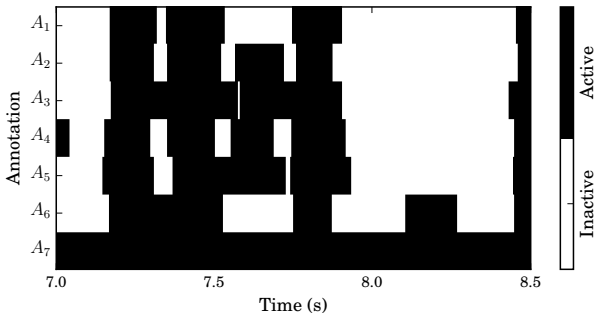


Figure 3. Excerpt from `Brow-JO`. $A_1, \ldots, A_4$ show the human annotations. A_5 and A_6 are results from automated approaches. A_7 is the baseline annotation which considers all frames as being active.

Ref. \ Est.	A_1	A_2	A_3	A_4	A_5	A_6	A_7	∅
A_1	–	0.13	0.30	0.27	0.22	0.44	1.00	0.39
A_2	0.12	–	0.29	0.26	0.22	0.43	1.00	0.39
A_3	0.05	0.07	–	0.14	0.18	0.43	1.00	0.31
A_4	0.16	0.16	0.27	–	0.24	0.46	1.00	0.38
A_5	0.34	0.35	0.48	0.44	–	0.49	1.00	0.52
A_6	0.38	0.38	0.54	0.49	0.35	–	1.00	0.52
A_7	0.00	0.00	0.00	0.00	0.00	0.00	–	0.00
∅	0.17	0.18	0.31	0.27	0.20	0.38	1.00	0.36

Table 4. Pairwise evaluation: *Voicing False Alarm* (VFA). The values are obtained by calculating the VFA for all possible annotation pairs (Table 2) and all solo recordings (Table 1). These values are then aggregated by using the arithmetic mean.

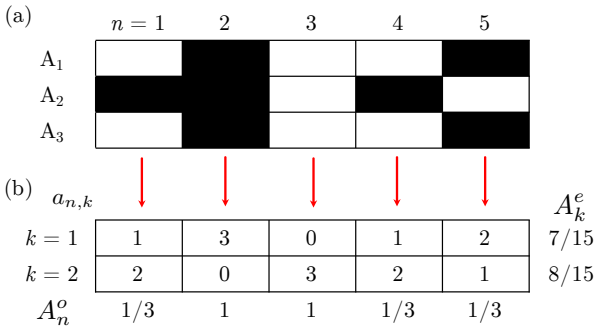


Figure 4. Example of evaluating Fleiss' κ for $K = 2$ categories, $N = 5$ frames, and three different annotations. (a) Annotations. (b) Number of annotations per category and time instance. Combining $A^o = 0.6$ and $A^e = 0.5$ leads to $\kappa = 0.2$.

< 0	$0 - 0.2$	$0.21 - 0.4$	$0.41 - 0.6$	$0.61 - 0.8$	$0.81 - 1$
poor	slight	fair	moderate	substantial	almost perfect

Table 5. Scale for interpreting κ as given by [11].

are not generalizable to take these into account (only by considering a mean over all pairs). Furthermore, although the presented evaluation measures are by design limited to yield values in $[0, 1]$, they can usually not be interpreted without some kind of baseline. For example, considering VD, the pair (A_2, A_3) yields a VD-value of 0.97, suggesting that A_3 can be considered as an "excellent" estimate. However, considering that our uninformed baseline A_7 yields a VD of 1.0, shows that it is meaningless to look at the VD alone. Similarly, an agreement with the trivial annotation A_7 only reflects the statistics on the active and inactive frames, thus being rather uninformative. Next, we introduce an evaluation measure that can overcome some of these problems.

3.2 Fleiss' Kappa

Having to deal with multiple human annotations is common in fields such as medicine or psychology. In these disciplines, measures that can account for multiple annotations have been developed. Furthermore, to compensate for chance-based agreement, a general concept referred to as *Kappa Statistic* [7] is used. In general, a kappa value lies in the range of $[-1, 1]$, where the value 1 means complete agreement among the raters, the value 0 means that the agreement is purely based on chance, and a value below 0 means that agreement is even below chance.

We now adapt *Fleiss' Kappa* to calculate the chance-corrected inter-annotator agreement for the soloist activity detection task. Following [7, 11], Fleiss' Kappa is defined as:

$$\kappa := \frac{A^o - A^e}{1 - A^e} \ . \tag{3}$$

In general, κ compares the mean observed agreement $A^o \in [0, 1]$ to the mean expected agreement $A^e \in [0, 1]$ which is solely based on chance. Table 5 shows a scale for the agreement of annotations with the corresponding range of κ.

SoloID \ Comb.	κ_H	$\kappa_{H,5}$	$\kappa_{H,6}$	ρ_5	ρ_6
Bech-ST	0.74	0.60	0.55	0.82	0.75
Brow-JO	0.68	0.56	0.59	0.82	0.87
Brow-JS	0.61	0.47	0.43	0.78	0.71
Brow-SD	0.70	0.61	0.51	0.87	0.73
Colt-BT	0.66	0.55	0.49	0.84	0.74
Full-BT	0.74	0.66	0.61	0.89	0.83
Getz-IP	0.72	0.69	0.64	0.96	0.90
Shor-FP	0.82	0.65	0.58	0.80	0.70
∅	0.71	0.60	0.55	0.85	0.78

Table 6. κ for all songs and different pools of annotations. κ_H denotes the pool of human annotations $A_1, \ldots, A_4$. These values are then aggregated by using the arithmetic mean.

To give a better feeling for how κ works, we exemplarily calculate κ for the example given in Figure 4(a). In this example, we have $R = 3$ different annotations $A_1, \ldots, A_3$ for $N = 5$ time instances. For each time instance, the annotations belong to either of $K = 2$ categories (*active* or *inactive*). As a first step, for each time instance, we add up the annotations for each category. This yields the number of annotations per category $a_{n,k} \in \mathbb{N}$, $n \in [1 : N]$, $k \in [1 : K]$ which is shown in Figure 4(b). Based on these distributions, we calculate the observed agreement A^o_n for a single time instance $n \in [1 : N]$ as:

$$A^o_n := \frac{1}{R(R-1)} \sum_{k=1}^{K} a_{n,k}(a_{n,k} - 1) \ , \tag{4}$$

which is the fraction of agreeing annotations normalized by the number of possible annotator pairs $R(R-1)$, e. g., for the time instance $n = 2$ in the example, all annotators agree for the frame to be active, thus $A^o_2 = 1$. Taking the arithmetic mean of all observed agreements leads to the mean observed agreement

$$A^o := \frac{1}{N} \sum_{n=1}^{N} A^o_n \ , \tag{5}$$

in our example $A^o = 0.6$. The remaining part for calculating κ is the expected agreement A^e. First, we calculate the distribution of agreements within each category $k \in [1 : K]$, normalized by the number of possible ratings NR:

$$A^e_k := \frac{1}{NR} \sum_{n=1}^{N} a_{n,k} \ , \tag{6}$$

e. g., in our example for $k = 1$ (active) results in $A^e_1 = {}^7/_{15}$. The expected agreement A_e is defined as [7]

$$A^e := \sum_{k=1}^{K} (A^e_k)^2 \tag{7}$$

which leads to $\kappa = 0.2$ for our example. According to the scale given in Table 5, this is a "slight" agreement.

In Table 6, we show the results for κ calculated for different pools of annotations. First, we calculate κ for the

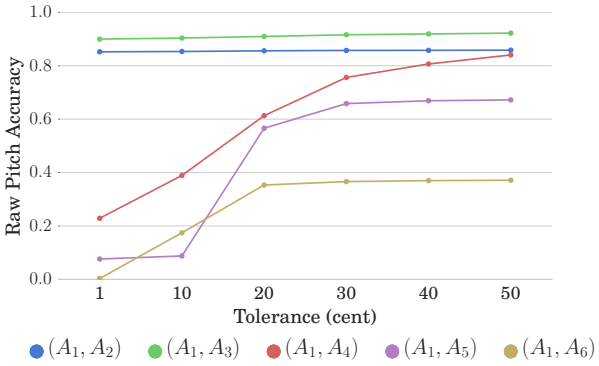


Figure 5. *Raw Pitch Accuracy* (RPA) for different pairs of annotations based on the annotations of the solo recording `Brow-JO`, evaluated on all active frames according to the reference annotation.

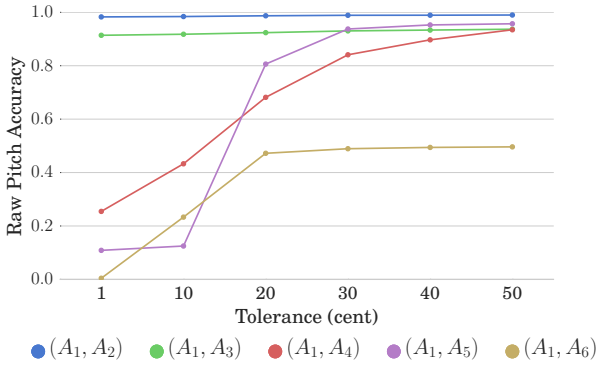


Figure 6. *Modified Raw Pitch Accuracy* for different pairs of annotations based on the annotations of the solo recording `Brow-JO`, evaluated on all active frames according to the *union* of reference and estimate annotation.

pool of human annotations $H := \{1, 2, 3, 4\}$, denoted as κ_H. κ_H yields values ranging from 0.61 to 0.82 which is considered as "substantial" to "almost perfect" agreement according to Table 5.

Now, reverting to our initial task of evaluating an automatically obtained annotation, the idea is to see how the κ-value changes when adding this annotation to the pool of all human annotations. A given automated procedure could then be considered to work correctly if it produces results that are just about as variable as the human annotations. Only if an automated procedure behaves fundamentally different than the human annotations, it will be considered to work incorrectly. In our case, calculating κ for the annotation pool $H \cup \{5\}$ yields values ranging from 0.47 to 0.69, as shown in column $\kappa_{H,5}$ of Table 6. Considering the annotation pool $H \cup \{6\}$, $\kappa_{H,6}$ results in κ-values ranging from 0.43 to 0.64. Considering the average over all individual recordings, we get mean κ-values of 0.60 and 0.55 for $\kappa_{H,5}$ and $\kappa_{H,6}$, respectively. Comparing these mean κ-values for the automated approaches to the respective κ_H, we can consider the method producing the annotation A_5 to be more consistent with the human annotations than A_6.

In order to quantify the agreement of an automatically generated annotation and the human annotations in a single value, we define the proportion $\rho \in \mathbb{R}$ as

$$\rho_5 := \frac{\kappa_{\mathrm{H},5}}{\kappa_{\mathrm{H}}}, \rho_6 := \frac{\kappa_{\mathrm{H},6}}{\kappa_{\mathrm{H}}}. \tag{8}$$

One can interpret ρ as some kind of "normalization" according to the inter-annotator agreement of the humans. For example, solo recording `Brow-JS` obtains the lowest agreement of $\kappa_H = 0.61$ in our test set. The algorithms perform "moderate" with $\kappa_{H,5} = 0.47$ and $\kappa_{H,6} = 0.43$. This moderate performance is partly alleviated when normalizing with the relatively low human agreement, leading to $\rho_5 = 0.78$ and $\rho_6 = 0.71$. On the other hand, for the solo recording `Shor-FP`, the human annotators had an "almost perfect" agreement of $\kappa_{H,6} = 0.82$. While the automated method's approaches were "substantial" with $\kappa_{H,5} = 0.65$ and "moderate" with $\kappa_{H,6} = 0.58$. However, although the automated method's κ-values are higher than for `Brow-JS`, investigating the proportions ρ_5 and ρ_6 reveal that the automated method's relative agreement with the human annotations is actually the same ($\rho_5 = 0.78$ and $\rho_5 = 0.71$ for `Brow-JS` compared to $\rho_5 = 0.80$ and $\rho_5 = 0.70$ for `Shor-FP`). This indicates the ρ-value's potential as an evaluation measure that can account for multiple human reference annotations in a meaningful way.

4. F0 ESTIMATION

One of the used standard measures for the evaluation of the F0 estimation in MIREX is the *Raw Pitch Accuracy* (RPA) which is computed for a pair of annotations (A_r, A_e) consisting of a reference A_r and an estimate annotation A_e. The core concept of this measure is to label an F0 estimate $A_e(n)$ to be correct, if its F0-value deviates from $A_r(n)$ by at most a fixed tolerance $\tau \in \mathbb{R}$ (usually $\tau = 50$ cent). Figure 5 shows the RPA for different annotation pairs and different tolerances $\tau \in \{1, 10, 20, 30, 40, 50\}$ (given in cent) for the solo recording `Brow-JO`, as computed by `MIR_EVAL`. For example, looking at the pair (A_1, A_4), we see that the RPA ascends with increasing value of τ. The reason for this becomes obvious when looking at Figure 7. While A_1 was created with the goal of having fine grained F0-trajectories, annotations A_4 was created with a transcription scenario in mind. Therefore, the RPA is low for very small τ but becomes almost perfect when considering a tolerance of half a semitone ($\tau = 50$ cent).

Another interesting observation in Figure 5 is that the annotation pairs (A_1, A_2) and (A_1, A_3) yield almost constant high RPA-values. This is the case since both annotations were created using the same annotation tool—yielding very similar F0-trajectories. However, it is noteworthy that there seems to be a "glass ceiling" that cannot be exceeded even for high τ-values. The reason for this lies in the exact definition of the RPA as used for MIREX. Let $\mu(A) := \{n \in [1 : N] : A(n) \neq *\}$ be the set of all active time instances of some annotation in $\mathcal{A}$. By definition, the RPA is only evaluated on the reference annotation's active time instances $\mu(A_r)$, where each

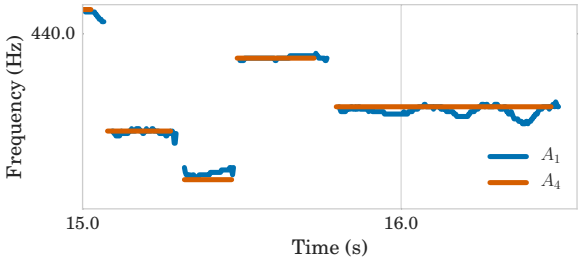


Figure 7. Excerpt from the annotations of the solo `Brow-JO` of A_1 and A_4.

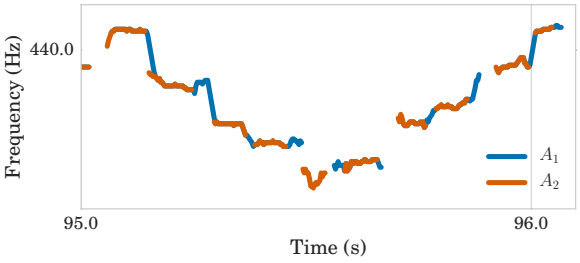


Figure 8. Excerpt from the annotations of the solo `Brow-JO` of A_1 and A_2.

$n \in \mu(A_r) \setminus \mu(A_e)$ is regarded as an incorrect time instance (for any τ). In other words, although the term "Raw Pitch Accuracy" suggests that this measure purely reflects correct F0-estimates, it is implicitly biased by the activity detection of the reference annotation. Figure 8 shows an excerpt of the human annotations A_1 and A_2 for the solo recording `Brow-JO`. While the F0-trajectories are quite similar, they differ in the annotated activity. In A_1, we see that transitions between consecutive notes are often annotated continuously—reflecting glissandi or slurs. This is not the case in A_2, where the annotation rather reflects individual note events. A musically motivated explanation could be that A_1's annotator had a performance analysis scenario in mind where note transitions are an interesting aspect, whereas A_2's annotator could have been more focused on a transcription task. Although both annotations are musically meaningful, when calculating the RPA for (A_1, A_2), all time instances where A_1 is active and A_2 not, are counted as incorrect (independent of τ)—causing the glass ceiling.

As an alternative approach that decouples the activity detection from the F0 estimation, one could evaluate the RPA only on those time instances, where reference *and* estimate annotation are active, i. e., $\mu(A_r) \cup \mu(A_e)$. This leads to the modified RPA-values as shown in Figure 6. Compared to Figure 5, all curves are shifted towards higher RPA-values. In particular, the pair (A_1, A_2) yields modified RPA-values close to one, irrespective of the tolerance τ—now indicating that A_1 and A_2 coincide perfectly in terms of F0 estimation.

However, it is important to note that the modified RPA evaluation measure may not be an expressive measure on its own. For example, in the case that two annotations are almost disjoint in terms of activity, the modified RPA would only be computed on the basis of a very small number of time instances, thus being statistically meaningless. Therefore, to rate a computational approach's performance, it is necessary to consider both, the evaluation of the activity detection as well as the F0 estimation, simultaneously but independent of each other. Both evaluations give valuable perspectives on the computational approach's performance for the task of predominant melody estimation and therefore help to get a better understanding of the underlying problems.

5. CONCLUSION

In this paper, we investigated the evaluation of automatic approaches for the task of predominant melody estimation—a task that can be subdivided into the subtask of soloist activity detection and F0 estimation. The evaluation of this task is not straightforward since the existence of a single "ground-truth" reference annotation is questionable. After having reviewed standard evaluation measures used in the field, one of our main contributions was to adapt Fleiss' Kappa—a measure which accounts for multiple reference annotations. We then explicitly defined and discussed Fleiss' Kappa for the task of the soloist activity detection.

The core motivation for using Fleiss' Kappa as an evaluation measure was to consider an automatic approach to work correctly, if its results were just about as variable as the human annotations. We therefore extended this the kappa measure by normalizing it by the variability of the human annotations. The resulting ρ-values allow for quantifying the agreement of an automatically generated annotation and the human annotations in a single value.

For the task of F0 estimation, we showed that the standard evaluation measures are biased by the activity detection task. This is problematic, since mixing both subtasks can obfuscate insights into advantages and drawbacks of a tested predominant melody estimation procedure. We therefore proposed an alternative formulation for RPA which decoupled the two tasks.

6. ACKNOWLEDGMENT

This work has been supported by the German Research Foundation (DFG MU 2686/6-1 and DFG PF 669/7-1). We would like to thank all members of the Jazzomat research project led by Martin Pfleiderer.

The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Fraunhofer-Institut für Integrierte Schaltungen IIS.

7. REFERENCES

[1] Accompanying website. `http://www.audiolabs-erlangen.de/resources/MIR/2016-ISMIR-Multiple-Annotations/`.

[2] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José R. Zapata, and Xavier Serra. Essentia: An audio analysis library for music information retrieval. In *Proc. of the Int. Society for Music Information Retrieval Conference (ISMIR)*, pages 493–498, Curitiba, Brazil, 2013.

[3] Juan J. Bosch and Emilia Gómez. Melody extraction in symphonic classical music: a comparative study of mutual agreement between humans and algorithms. In *Proc. of the Conference on Interdisciplinary Musicology (CIM)*, December 2014.

[4] Chris Cannam, Christian Landone, and Mark B. Sandler. Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files. In *Proc. of the Int. Conference on Multimedia*, pages 1467–1468, Florence, Italy, 2010.

[5] J. Stephen Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[6] Jonathan Driedger and Meinard Müller. Verfahren zur Schätzung der Grundfrequenzverläufe von Melodiestimmen in mehrstimmigen Musikaufnahmen. In Wolfgang Auhagen, Claudia Bullerjahn, and Richard von Georgi, editors, *Musikpsychologie – Anwendungsorientierte Forschung*, volume 25 of *Jahrbuch Musikpsychologie*, pages 55–71. Hogrefe-Verlag, 2015.

[7] Joseph L. Fleiss, Bruce Levin, and Myunghee Cho Paik. *Statistical Methods for Rates and Proportions*. John Wiley Sons, Inc., 2003.

[8] Arthur Flexer. On inter-rater agreement in audio music similarity. In *Proc. of the Int. Conference on Music Information Retrieval (ISMIR)*, pages 245–250, Taipei, Taiwan, 2014.

[9] Klaus Frieler, Wolf-Georg Zaddach, Jakob Abeßer, and Martin Pfleiderer. Introducing the jazzomat project and the melospy library. In *Third Int. Workshop on Folk Music Analysis*, 2013.

[10] M. Cameron Jones, J. Stephen Downie, and Andreas F. Ehmann. Human similarity judgments: Implications for the design of formal evaluations. In *Proc. of the Int. Conference on Music Information Retrieval (ISMIR)*, pages 539–542, Vienna, Austria, 2007.

[11] J. Richard Landis and Gary G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[12] Matthias Mauch, Chris Cannam, Rachel Bittner, George Fazekas, Justing Salamon, Jiajie Dai, Juan Bello, and Simon Dixon. Computer-aided melody note transcription using the Tony software: Accuracy and efficiency. In *Proc. of the Int. Conference on Technologies for Music Notation and Representation*, May 2015.

[13] Matthias Mauch and Simon Dixon. pYIN: A fundamental frequency estimator using probabilistic threshold distributions. In *IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7480–7484, 2014.

[14] MIREX. Audio melody extraction task. Website `http://www.music-ir.org/mirex/wiki/2015:Audio_Melody_Extraction`, last accessed 01/19/2016, 2015.

[15] Meinard Müller. *Fundamentals of Music Processing*. Springer Verlag, 2015.

[16] Oriol Nieto, Morwaread Farbood, Tristan Jehan, and Juan Pablo Bello. Perceptual analysis of the F-measure to evaluate section boundaries in music. In *Proc. of the Int. Society for Music Information Retrieval Conference (ISMIR)*, pages 265–270, Taipei, Taiwan, 2014.

[17] Jouni Paulus and Anssi P. Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(6):1159–1170, 2009.

[18] The Jazzomat Research Project. Database download, last accessed: 2016/02/17. `http://jazzomat.hfm-weimar.de`.

[19] Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis. MIR_EVAL: A transparent implementation of common MIR metrics. In *Proc. of the Int. Conference on Music Information Retrieval (ISMIR)*, pages 367–372, Taipei, Taiwan, 2014.

[20] Justin Salamon and Emilia Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[21] Justin Salamon, Emilia Gómez, Daniel P. W. Ellis, and Gaël Richard. Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, 2014.

[22] Justin Salamon and Julián Urbano. Current challenges in the evaluation of predominant melody extraction algorithms. In *Proc. of the Int. Society for Music Information Retrieval Conference (ISMIR)*, pages 289–294, Porto, Portugal, October 2012.

[23] Jordan Bennett Louis Smith, John Ashley Burgoyne, Ichiro Fujinaga, David De Roure, and J. Stephen Downie. Design and creation of a large-scale database of structural annotations. In *Proc. of the Int. Society for Music Information Retrieval Conference (ISMIR)*, pages 555–560, Miami, Florida, USA, 2011.

Oral Session 2

Rhythm

JOINT BEAT AND DOWNBEAT TRACKING WITH RECURRENT NEURAL NETWORKS

Sebastian Böck, Florian Krebs, and Gerhard Widmer
Department of Computational Perception
Johannes Kepler University Linz, Austria
sebastian.boeck@jku.at

ABSTRACT

In this paper we present a novel method for jointly extracting beats and downbeats from audio signals. A recurrent neural network operating directly on magnitude spectrograms is used to model the metrical structure of the audio signals at multiple levels and provides an output feature that clearly distinguishes between beats and downbeats. A dynamic Bayesian network is then used to model bars of variable length and align the predicted beat and downbeat positions to the global best solution. We find that the proposed model achieves state-of-the-art performance on a wide range of different musical genres and styles.

1. INTRODUCTION

Music is generally organised in a hierarchical way. The lower levels of this hierarchy are defined by the *beats* and *downbeats* which define the *metrical structure* of a musical piece. While considerable amount of research focused on finding the *beats* in music, far less effort has been made to track the *downbeats*, although this information is crucial for a lot of higher level tasks such as structural segmentation and music analysis and applications like automated DJ mixing. In western music, the *downbeats* often coincide with chord changes or harmonic cues, whereas in non-western music the start of a *measure* is often defined by the boundaries of rhythmic patterns. Therefore, many algorithms exploit one or both of these features to track the *downbeats*.

Klapuri et al. [18] proposed a system which jointly analyses a musical piece at three time scales: the tatum, tactus, and measure level. The signal is split into multiple bands and then combined into four accent bands before being fed into a bank of resonating comb filters. Their temporal evolution and the relation of the different time scales are modelled with a probabilistic framework to report the final position of the downbeats.

The system of Davies and Plumbley [5] first tracks the beats and then calculates the Kullback-Leibler divergence between two consecutive band-limited beat synchronous spectral difference frames to detect the downbeats, exploiting the fact that lower frequency bands are perceptually more important.

Papadopoulos and Peeters [24] jointly track chords and downbeats by decoding a sequence of (pre-computed) beat synchronous chroma vectors with a hidden Markov model (HMM). Two time signatures are modelled. In a later paper, the same authors [25] jointly model beat phase and downbeats while the tempo is assumed to be given. Beat and downbeat times are decoded using a HMM from three input features: the correlation of the local energy with a beat-template, chroma vector variation, and the spectral balance between high and low frequency content.

The system proposed by Khadkevich et al. [17] uses impulsive and harmonic components of a reassigned spectrogram together with chroma variations as observation features for a HMM. The system is based on the assumption that downbeats mostly occur at location with harmonic changes.

Hockman et al. [14] present a method designed specifically for hardcore, jungle, and drum and bass music, that often employ breakbeats. The system exploits onset features and periodicity information from a beat tracking stage, as well as information from a regression model trained on the breakbeats specific to the musical genre.

Durand et al. [10] first estimates the time signature by examining the similarity of the frames at the beat level – with the beat positions given as input. The downbeats are then selected by a linear support vector machine (SVM) model using a bag of complementary features, comprising chord changes, harmonic balance, melodic accents and pattern changes. In consecutive works [8, 9] they lifted the requirement of the beat positions to be given and enhanced their system considerably by replacing the SVM feature selection stage by several deep neural networks which learn higher level representations from which the final downbeat positions are selected by means of Viterbi decoding.

Krebs et al. [20] jointly model bar position, tempo, and rhythmic patterns with a dynamic Bayesian network (DBN) and apply their system to a dataset of ballroom dance music. Based on their work, [16] developed a unified model for metrical analysis of Turkish, Carnatic, and Cretan music. Both models were later refined by using a more sophisticated state space [21].

© Sebastian Böck, Florian Krebs, and Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Sebastian Böck, Florian Krebs, and Gerhard Widmer. "JOINT BEAT AND DOWNBEAT TRACKING WITH RECURRENT NEURAL NETWORKS", 17th International Society for Music Information Retrieval Conference, 2016.

The same state space has also been successfully applied to the beat tracking system proposed by Böck et al. [2]. The system uses a recurrent neural network (RNN) similar to the one proposed in [3] to discriminate between beats an non-beats at a frame level. A DBN then models the tempo and the phase of the beat sequence.

In this paper, we extend the RNN-based beat tracking system in order to jointly track the whole metrical cycle, including beats and downbeats. The proposed model avoids hand-crafted features such as harmonic change detection [8–10, 17, 24], or rhythmic patterns [14, 16, 20], but rather learns the relevant features directly from the spectrogram. We believe that this is an important step towards systems without cultural bias, as postulated by the "Roadmap for Music Information Research" [26].

2. ALGORITHM DESCRIPTION

The proposed method consists of a recurrent neural network (RNN) similar to the ones proposed in [2, 3], and is trained to jointly detect the beats and downbeats of an audio signal in a supervised classification task. A dynamic Bayesian network is used as a post-processing step to determine the globally best sequence through the state-space by jointly inferring the meter, tempo, and phase of the (down-)beat sequence.

2.1 Signal Pre-Processing

The audio signal is split into overlapping frames and weighted with a Hann window of same length before being transferred to a time-frequency representation with the Short-time Fourier Transform (STFT). Two adjacent frames are located 10 ms apart, which corresponds to a rate of 100 fps (frames per second). We omit the phase portion of the complex spectrogram and use only the magnitudes for further processing. To enable the network to capture features which are precise both in time and frequency, we use three different magnitude spectrograms with STFT lengths of 1024, 2048, and 4096 samples (at a signal sample rate of 44.1 kHz). To reduce the dimensionality of the features, we limit the frequencies range to [30, 17000] Hz and process the spectrograms with logarithmically spaced filters. A filter with 12 bands per octave corresponds to semitone resolution, which is desirable if the harmonic content of the spectrogram should be captured. However, using the same number of bands per octave for all spectrograms would result in an input feature of undesirable size. We therefor use filters with 3, 6, and 12 bands per octave for the three spectrograms obtained with 1024, 2028, and 4096 samples, respectively, accounting for a total of 157 bands. To better match human perception of loudness, we scale the resulting frequency bands logarithmically. To aid the network during training, we add the first order differences of the spectrograms to our input features. Hence, the final input dimension of the neural network is 314. Figure 1a shows the part of the input features obtained with 12 bands per octave.

2.2 Neural Network Processing

As a network we chose a system similar to the one presented in [3], which is also the basis for the current state-of-the-art in beat tracking [2, 19].

2.2.1 Network topology

The network consists of three fully connected bidirectional recurrent layers with 25 Long Short-Term Memory (LSTM) units each. Figures 1b to 1d show the output activations of the forward (i.e. half of the bidirectional) hidden layers. A softmax classification layer with three units is used to model the *beat*, *downbeat*, and *non-beat* classes. A frame can only be classified as downbeat *or* beat but not both at the same time, enabling the following dynamic Bayesian network to infer the meter and downbeat positions more easily. The output of the neural network are three activation functions b_k, d_k, and no_k, which represents the probability of a frame k being a beat but no downbeat, downbeat or non-beat position. Figure 1e shows b_k and d_k for an audio example.

2.2.2 Network training

We train the network on the datasets described in Section 3.1 — except the ones marked with an asterisk (*) which are used for testing only — with 8-fold cross validation based on a random splits. We initialise the network weights and biases with a uniform random distribution with range [-0.1, 0.1] and train it with stochastic gradient decent minimising the cross entropy error with a learning rate of 10^{-5} and 0.9 momentum. We stop training if no improvement on the validation set can be observed for 20 epochs. We then reduce the learning rate by a factor of ten and retrain the previously best model with the same early stopping criterion.

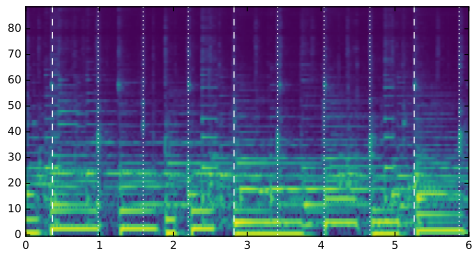
2.2.3 Network output thresholding

We experienced that the very low activations at the beginning and end of a musical excerpt can hurt the tracking performance of the system. This is often the case if a song starts with a (musically irrelevant) intro or has a long fade out at the end. We thus threshold the activations and use only the activations between the first and last time they exceed the threshold. We empirically found a threshold value $\theta = 0.05$ to perform well without harming pieces with overall low activations (e.g. choral works).

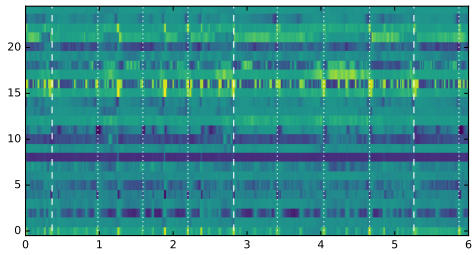
2.3 Dynamic Bayesian Network

We use the output of the neural network as observations of a *dynamic Bayesian network (DBN)* which jointly infers the meter, tempo, and phase of a (down-)beat sequence. The DBN is very good at dealing with ambiguous RNN observations and finds the global best state sequence given these observations. [1] We use the state-space proposed in [21] to model a whole bar with an arbitrary number of

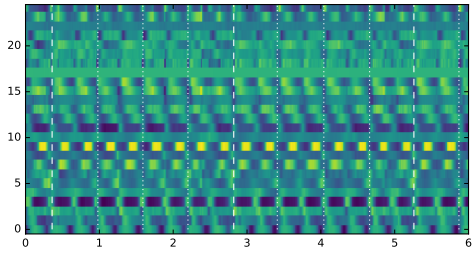
[1] The average performance gain of the DBN compared to simple thresholding and peak-picking of the RNN activations is about 15% F-measure on the validation set.



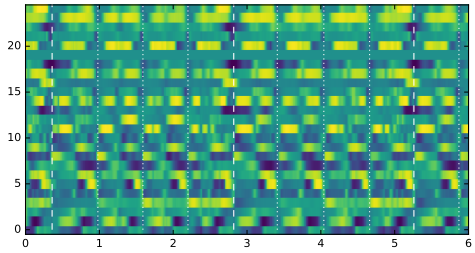
(a) Spectrogram of audio signal



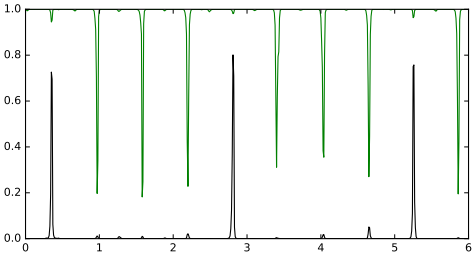
(b) Activations of the first hidden layer



(c) Activations of the second hidden layer



(d) Activations of the third hidden layer



(e) Activations of the softmax output layer

Figure 1: Signal propagation of a 6 second song excerpt in 4/4 time signature through the network: *(a)* part of the input features, *(b)* the first hidden layer shows activations at onset positions, *(c)* the second models mostly faster metrical levels (e.g. 1/8th notes at neuron 3), *(d)* the third layer models multiple metrical levels (e.g. neuron 8 firing at beat positions and neuron 16 around downbeat positions), *(e)* the softmax output layer finally models the relation of the different metrical levels resulting in clear downbeat (black) and beat (green, flipped for better visualisation) activations. Downbeat positions are marked with vertical dashed lines, beats as dotted lines.

beats per bar. We do not allow meter changes throughout a musical piece, thus we can model different meters with individual, independent state spaces. All parameters of the DBN are tuned to maximise the downbeat tracking performance on the validation set.

2.3.1 State Space

We divide the state space into discrete states $\mathbf{s}$ to make inference feasible. These states $s(\phi, \dot{\phi}, r)$ lie in a three-dimensional space indexed by the bar position state $\phi \in \{1..\Phi\}$, the tempo state $\dot{\phi} \in \{1..\dot{\Phi}\}$, and the time signature state r (e.g. $r \in \{3/4, 4/4\}$). States that fall on a *downbeat* position ($\phi = 1$) constitute the set of *downbeat* states $\mathcal{D}$, all states that fall on a *beat* position define the set of *beat* states $\mathcal{B}$. The number of bar-position states of a tempo $\dot{\phi}$ is proportional to its corresponding beat period $1/\dot{\phi}$, and the number of tempo states depends on the tempo ranges that the model accounts for. For generality, we assume equal tempo ranges for all time signatures in this paper but this could easily changed to adapt the model towards specific styles. In line with [21] we find that by distributing the tempo states logarithmically across the beat intervals, the size of the state space can be reduced efficiently without affecting the performance too much. Empirically we found that using $N = 60$ tempo states is a good compromise between computation time and performance.

2.3.2 Transition Model

Tempo transitions are only allowed at the beats and follow the same exponential distribution proposed in [21]. We investigated "peephole" transitions from the end of every beat back to the beginning of the bar, but found them to harm performance. Thus, we assume that there are no transitions between time signatures in this paper.

2.3.3 Observation Model

We adapted the observation model of the DBN from [2] to not only predict beats, but also downbeats. Since the activation functions (d, b) produced by the neural network are limited to the range $[0, 1]$ and show high values at beat/downbeat positions and low values at non-beat positions (cf. Figure 1e), the activations can be converted into state-conditional observation distributions $P(o_k|s_k)$ by

$$P(o_k|s_k) = \begin{cases} d_k & s_k \in \mathcal{D} \\ b_k & s_k \in \mathcal{B} \\ \frac{n_k}{\lambda_o - 1}, & otherwise \end{cases} \quad (1)$$

where $\mathcal{D}$ and $\mathcal{B}$ are the sets of downbeat and beat states respectively, and the observation lambda $\lambda_o \in [\frac{\Phi}{\Phi-1}, \Phi]$ is a parameter that controls the proportion of the beat/downbeat interval which is considered as beat/downbeat and non-beat locations inside one beat/downbeat period. On our validation set we achieved the best results with the value $\lambda_o = 16$. We found it to be advantageous to use both b_k and d_k as provided by the neural network instead of splitting the probability of b_k among the N beat positions of the transition model.

2.3.4 Initial State Distribution

The initial state distribution can be used to incorporate any prior knowledge about the hidden states, such as meter and tempo distributions. In this paper, we use a uniform distribution over all states.

2.3.5 Inference

We are interested in the sequence of hidden states $\mathbf{s}_{1:K}$, that maximise the posterior probability of the hidden states given the observations (activations of the network). We obtain the maximum a-posteriori state sequence $\mathbf{s}^*_{1:K}$ by

$$\mathbf{s}^*_{1:K} = \arg\max_{\mathbf{s}_{1:K}} p(\mathbf{s}_{1:K}|o_{1:K}) \tag{2}$$

which can be computed efficiently using the well-known Viterbi algorithm.

2.3.6 Beat and Downbeat Selection

The sequence of beat $\mathbf{B}$ and downbeat times $\mathbf{D}$ are determined by the set of time frames k which were assigned to a beat or downbeat state:

$$\mathbf{B} = \{k : s^*_k \in \mathcal{B}\} \tag{3}$$

$$\mathbf{D} = \{k : s^*_k \in \mathcal{D}\} \tag{4}$$

After having decided on the sequences of beat and downbeat times we further refine them by looking for the highest beat/downbeat activation value inside a window of size Φ/λ_o, i.e. the beat/downbeat range of the whole beat/downbeat period of the observation model (Section 2.3.3).

3. EVALUATION

In line with almost all other publications on the topic of downbeat tracking, we report the F-measure (F_1) with a tolerance window of $\pm$70 ms.

3.1 Datasets

For training and evaluation we use diverse datasets as shown in Table 1. Musical styles range from pop and rock music, over ballroom dances, modern electronic dance music, to classical and non-western music.

We do not report scores for all sets used for training, since comparisons with other works are often not possible due to different evaluation metrics and/or datasets. Results for all datasets, including additional metrics can be found online at the supplementary website `http://www.cp.jku.at/people/Boeck/ISMIR2016.html` which also includes an open source implementation of the algorithm.

Downbeat tracking dataset	# files	length
Ballroom [12, 20] [2]	685	5 h 57 m
Beatles [4]	180	8 h 09 m
Hainsworth [13]	222	3 h 19 m
HJDB [14]	235	3 h 19 m
RWC Popular [11]	100	6 h 47 m
Robbie Williams [7]	65	4 h 31 m
Rock [6]	200	12 h 53 m
Carnatic [28]	176	16 h 38 m
Cretan [16]	42	2 h 20 m
Turkish [27]	93	1 h 33 m
GTZAN [23, 29] *	999	8 h 20 m
Klapuri [18] [3] *	320	4 h 54 m
Beat tracking datasets		
SMC [15] *	217	2 h 25 m
Klapuri [18] [3] *	474	7 h 22 m

Table 1: Overview of the datasets used for training and evaluation of the algorithm. Sets marked with asterisks (*) are held-out datasets for testing only.

3.2 Results & Discussion

Table 2 to 4 list the results obtained by the proposed method compared to current and previous state-of-the-art algorithms on various datasets. We group the datasets into different tables for clarity, based on whether they are used for testing only, cover western, or non-western music. Since our system jointly tracks beats and downbeats, we compare with both downbeat and beat tracking algorithms.

First of all, we evaluate on completely unseen data. We use the recently published beat and downbeat annotations for the *GTZAN* dataset, the *Klapuri*, and the *SMC* set (built specifically to comprise hard-to-track musical pieces) for evaluation. Results are given in Table 2. Since these results are directly comparable (the only exception being the results of Durand et al. on the *Klapuri* set [4] and of Böck et al. on the *SMC* set [5]), we perform statistical significance tests on them. We use Wilcoxon's signed-rank test with a p-value of 0.01.

Additionally, we report the performance on other sets commonly used in the literature, comprising both western and non-western music. For western music, we give results on the *Ballroom*, *Beatles*, *Hainsworth*, and *RWC Popular* sets in Table 3. For non-western music we use the *Carnatic*, *Cretan*, and *Turkish* datasets and group the results in Table 4. Since these sets were also used during development and training of our system, we report results obtained with 8-fold cross validation. Please note that the results given in Table 3 and 4 are not directly comparable because they were either obtained via cross validation, leave-one-dataset-out evaluation, with overlapping train and test sets, or tested on unseen data. However, we still consider them

[2] We removed the 13 duplicates identified by Bob Sturm: http://media.aau.dk/null_space_pursuits/2014/01/ballroom-dataset.html

[3] The beat and downbeat annotations of this set were made independently, thus the positions do not necessarily match each other.

[4] 40 out of the 320 tracks were used for training.

[5] The complete set was used for training.

Test datasets	F_1 beat	F_1 downbeat
GTZAN		
new (bar lengths: 3, 4) *	0.856	0.640
Durand et al. [9] *	-	0.624
Böck et al. [2] *	0.864	-
Davies et al. [5] *	0.806	0.462
Klapuri et al. [18] *	0.706	0.309
Klapuri		
new (bar lengths: 3, 4) *	0.811	0.745
Durand et al. [9] ‡	-	0.689
Böck et al. [2] *	0.798	-
Davies et al. [5] *	0.698	0.528
Klapuri et al. [18] ‡	0.704	0.483
SMC		
new (bar lengths: 3, 4) *	0.516	
Böck et al. [2] §	0.529	
Davies et al. [5] *	0.337	
Klapuri et al. [18] *	0.352	

Table 2: Beat and downbeat tracking F-measure comparison with state-of-the-art algorithms on the test datasets. ‡ denotes overlapping train and test sets, § cross validation, and * testing only.

to be a good indicator for the overall performance and capabilities of the systems. For the music with non-western rhythms and meters (e.g. Carnatic art music contains 5/4 and 7/4 meters) we compare only with algorithms specialised on this type of music, since other systems typically fail completely on them.

Western music	F_1 beat	F_1 downbeat
Ballroom		
new (bar lengths: 3, 4) §	0.938	0.863
Durand et al. [9] †/‡	-	0.778 / 0.797
Krebs et al. [21] §	0.919	-
Böck et al. [2] §	0.910	-
Beatles		
new (bar lengths: 3, 4) §	0.918	0.832
Durand et al. [9] †/‡	-	0.815 / 0.842
Böck et al. [2] *	0.880	-
Hainsworth		
new (bar lengths: 3, 4) §	0.867	0.684
Durand et al. [9] †/‡	-	0.657 / 0.664
Böck et al. [2] §	0.843	-
Peeters et al. [25]	0.630	
RWC Popular		
new (bar lengths: 3, 4) §	0.943	0.861
Durand et al. [9] †/‡	-	0.860 / 0.879
Böck et al. [2] *	0.877	-
Peeters et al. [25]	0.840	0.800

Table 3: Beat and downbeat tracking F-measure comparison with state-of-the-art algorithms on western music datasets. † denotes leave-one-set-out evaluation, ‡ overlapping train and test sets, § cross validation, and * testing only.

Non-western music	F_1 beat	F_1 downbeat
Carnatic		
new (bar lengths: 3, 4) §	0.804	0.365
—— (bar lengths: 3, 5, 7, 8) §	0.792	0.593
Krebs et al. [21] §	0.805	0.472
Cretan		
new (bar lengths: 3, 4) §	0.982	0.605
—— (bar lengths: 2, 3, 4) §	0.981	0.818
—— (bar lengths: 2) §	0.980	0.909
Krebs et al. [21] §	0.912	0.774
Turkish		
new (bar lengths: 3, 4) §	0.740	0.495
—— (bar lengths: 4, 8, 9, 10) §	0.777	0.631
—— (tempo: 55..300 bpm) §	0.818	0.683
Krebs et al. [21] §	0.826	0.639

Table 4: Beat and downbeat tracking F-measure comparison with state-of-the-art algorithms on non-western music datasets. —— denotes the same system as the line above with altered parameters in parentheses, § cross validation.

3.2.1 Beat tracking

Compared to the current state-of-the-art [2], the new system performs on par or outperforms this dedicated beat tracking algorithm. It only falls a bit behind on the *GTZAN* and *SMC* sets. However, the results on the latter might be a bit biased, since [2] obtained their results with 8-fold cross validation. Although the new system performs better on the *Klapuri set*, the difference is not statistically significant. All results compared to those of other beat tracking algorithms on the test datasets in Table 2 are statistically significant.

Although the new algorithm and [2] have a very similar architecture and were trained on almost the same development sets (the new one plus those sets given in Table 1, except the *SMC* dataset), it is hard to conclude whether the new algorithm performs better sometimes because of the additional – more diverse – training material or due to the joint modelling of beats and downbeats. Future investigations with the same training sets should shed some light on this question, but it is safe to conclude that the joint training on beats and downbeats does not harm the beat tracking performance at all.

On non-western music the results are in the same range as the ones obtained by the method of Krebs et al. [21], an enhanced version of the algorithm proposed by Holzapfel et al. [16]. Our system shows almost perfect beat tracking results on the *Cretan* lap dances while performing a bit worse on the *Turkish* music.

3.2.2 Downbeat tracking

From Table 2 to 4, it can be seen that the proposed system not only does well for beat tracking, but also shows state-of-the-art performance in downbeat tracking. We outperform all other methods on all datasets – except *Beatles* and *RWC Popular* when comparing to the overfitted results obtained by the system of Durand et al. [9] – even the sys-

tems designed specifically for non-western music. We find this striking, since our new system is not designed specifically for a certain music style or genre. The results of our method w.r.t. the other systems on the test datasets in Table 2 are all statistically significant.

It should be noted however, that the dynamic Bayesian network must model the needed bar lengths for the respective music in order to achieve this performance. Especially when dealing with non-western music, this is crucial. However, we do not consider this a drawback, since the system is able to chose the correct bar length reliably by itself.

3.2.3 Meter selection

As mentioned above, for best performance the DBN must model measures with the correct number of beats per bar. Per default, our system works for 3/4 and 4/4 time signatures, but since the parameters of the DBN are not learnt, this can be changed during runtime in order to model any time signature and tempo range.

To investigate the system's ability to automatically decide on which bar length to select, we performed an experiment and limited the DBN to model only bars with lengths of three or four beats, both time signatures simultaneously (the default setting), or bar lengths of up to eight beats.

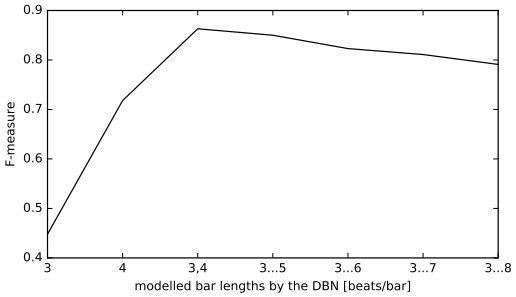


Figure 2: Downbeat tracking performance of the new system with different bar lengths on the *Ballroom* set.

Figure 2 shows this exemplarily for the *Ballroom* set, which comprises four times as many pieces in 4/4 as in 3/4 time signature. The performance is relatively low if the system is limited to model bars with only three or four beats per bar. When being able to model both time signatures present in the music, the system achieves it's maximum performance. The performance then slightly decreases if the DBN models bars with a length up to eight beats per bar, but remains on a relatively high performance level. This shows the system's ability to select the correct bar length automatically.

4. CONCLUSION

In this paper we presented a novel method for jointly tracking beats and downbeats with a recurrent neural network (RNN) in conjunction with a dynamic Bayesian network (DBN). The RNN is responsible for modelling the metrical structure of the musical piece at multiple interrelated levels and classifies each audio frame as being either a beat, downbeat, or no beat. The DBN then post-processes the probability functions of the RNN to align the beats and downbeats to the global best solution by jointly inferring the meter, tempo, and phase of the sequence. The system shows state-of-the-art beat and downbeat tracking performance on a wide range of different musical genres and styles. It does so by avoiding hand-crafted features such as harmonic changes, or rhythmic patterns, but rather learns the relevant features directly from audio. We believe that this is an important step towards systems without any cultural bias. We provide a reference implementation of the algorithm as part of the open-source *madmom* [1] framework.

Future work should address the limitation of the system of not being able to perform time signature changes within a musical piece. Due to the large state space needed this is intractable right now, but particle filters as used in [22] should be able to resolve this issue.

5. ACKNOWLEDGMENTS

This work is supported by the European Union Seventh Framework Programme FP7 / 2007-2013 through the GiantSteps project (grant agreement no. 610591) and the Austrian Science Fund (FWF) project Z159. We would like to thank all authors who shared their source code, datasets, annotations and detections or made them publicly available.

6. REFERENCES

[1] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer. madmom: a new Python Audio and Music Signal Processing Library. `arXiv:1605.07008`, 2016.

[2] S. Böck, F. Krebs, and G. Widmer. A multi-model approach to beat tracking considering heterogeneous music styles. In *Proc. of the 15th Int. Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[3] S. Böck and M. Schedl. Enhanced Beat Tracking with Context-Aware Neural Networks. In *Proc. of the 14th Int. Conference on Digital Audio Effects (DAFx)*, 2011.

[4] M. E. P. Davies, N. Degara, and M. D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. Technical Report C4DM-TR-09-06, Centre for Digital Music, Queen Mary University of London, 2009.

[5] M. E. P. Davies and M. D. Plumbley. A spectral difference approach to downbeat extraction in musical audio. In *Proc. of the 14th European Signal Processing Conference (EUSIPCO)*, 2006.

[6] T. de Clercq and D. Temperley. A corpus analysis of rock harmony. *Popular Music*, 30(1):47–70, 2011.

[7] B. Di Giorgi, M. Zanoni, A. Sarti, and S. Tubaro. Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony. In *8th Int. Workshop on Multidimensional Systems (nDS)*, pages 145–150, 2013.

[8] S. Durand, J. P. Bello, B. David, and G. Richard. Downbeat tracking with multiple features and deep neural networks. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[9] S. Durand, J. P. Bello, B. David, and G. Richard. Feature Adapted Convolutional Neural Networks for Downbeat Tracking. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[10] S. Durand, B. David, and G. Richard. Enhancing downbeat detection when facing different music styles. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[11] M. Goto, M. Hashiguchi, T. Nishimura, and R. Oka. RWC Music Database: Popular, Classical, and Jazz Music Databases. In *Proc. of the 3rd Int. Conference on Music Information Retrieval (ISMIR)*, 2002.

[12] F. Gouyon, A. P. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5), 2006.

[13] S. Hainsworth and M. Macleod. Particle filtering applied to musical tempo tracking. *EURASIP Journal on Applied Signal Processing*, 15, 2004.

[14] J. Hockman, M. E. P. Davies, and I. Fujinaga. One in the jungle: Downbeat detection in hardcore, jungle, and drum and bass. In *Proc. of the 13th Int. Society for Music Information Retrieval Conference (ISMIR)*, 2012.

[15] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon. Selective sampling for beat tracking evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(9), 2012.

[16] A. Holzapfel, F. Krebs, and A. Srinivasamurthy. Tracking the "odd": meter inference in a culturally diverse music corpus. In *Proc. of the 15th Int. Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[17] M. Khadkevich, T. Fillon, G. Richard, and M. Omologo. A probabilistic approach to simultaneous extraction of beats and downbeats. In *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[18] A. P. Klapuri, A. J. Eronen, and J. T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1), 2006.

[19] F. Korzeniowski, S. Böck, and G. Widmer. Probabilistic extraction of beat positions from a beat activation function. In *Proc. of the 15th Int. Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[20] F. Krebs, S. Böck, and G. Widmer. Rhythmic pattern modeling for beat and downbeat tracking in musical audio. In *Proc. of the 14th Int. Society for Music Information Retrieval Conference (ISMIR)*, 2013.

[21] F. Krebs, S. Böck, and G. Widmer. An Efficient State Space Model for Joint Tempo and Meter Tracking. In *Proc. of the 16th Int. Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[22] F. Krebs, A. Holzapfel, A. T. Cemgil, and G. Widmer. Inferring metrical structure in music using particle filters. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(5):817–827, 2015.

[23] U. Marchand and G. Peeters. Swing ratio estimation. In *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx)*, 2015.

[24] H. Papadopoulos and G. Peeters. Joint estimation of chords and downbeats from an audio signal. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1), 2011.

[25] G. Peeters and H. Papadopoulos. Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6), 2011.

[26] X. Serra et al. *Roadmap for Music Information ReSearch*. Creative Commons BY-NC-ND 3.0 license, ISBN: 978-2-9540351-1-6, 2013.

[27] A. Srinivasamurthy, A. Holzapfel, and X. Serra. In search of automatic rhythm analysis methods for turkish and indian art music. *Journal of New Music Research*, 43(1), 2014.

[28] A. Srinivasamurthy and X. Serra. A supervised approach to hierarchical metrical cycle tracking from audio music recordings. In *Proc. of the IEEE Int. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

[29] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5), 2002.

BAYESIAN METER TRACKING ON LEARNED SIGNAL REPRESENTATIONS

Andre Holzapfel, Thomas Grill
Austrian Research Institute for Artificial Intelligence (OFAI)
andre@rhythmos.org, thomas.grill@ofai.at

ABSTRACT

Most music exhibits a pulsating temporal structure, known as *meter*. Consequently, the task of *meter tracking* is of great importance for the domain of Music Information Retrieval. In our contribution, we specifically focus on Indian art musics, where meter is conceptualized at several hierarchical levels, and a diverse variety of metrical hierarchies exist, which poses a challenge for state of the art analysis methods. To this end, for the first time, we combine Convolutional Neural Networks (CNN), allowing to transcend manually tailored signal representations, with subsequent Dynamic Bayesian Tracking (BT), modeling the recurrent metrical structure in music. Our approach estimates meter structures simultaneously at two metrical levels. The results constitute a clear advance in meter tracking performance for Indian art music, and we also demonstrate that these results generalize to a set of Ballroom dances. Furthermore, the incorporation of neural network output allows a computationally efficient inference. We expect the combination of learned signal representations through CNNs and higher-level temporal modeling to be applicable to all styles of metered music, provided the availability of sufficient training data.

1. INTRODUCTION

The majority of musics in various parts of the world can be considered as metered, that is, their temporal organization is based on a hierarchical structure of pulsations at different related time-spans. In Eurogenetic music, for instance, one would refer to one of these levels as the beat or tactus level, and to another (longer) time-span level as the downbeat, measure, or bar level. In Indian art musics, the concepts of *tāḷa* for Carnatic and *tāl* for Hindustani music define metrical structures that consist of several hierarchical levels. However, important differences between meter(s) in Eurogenetic and Indian art musics are the presence of non-isochronicity in some of the metrical layers and the fact that an understanding of the progression of the meter is crucial for the appreciation of the listener, see, e.g. [3, p. 199ff]. Again, other cultures might not explicitly define metrical structure on several layers, but just define certain rhythmic modes that determine the length of a metrical cycle and some points of emphasis within this cycle, as is the case for Turkish makam music [2] or Korean music [13]. Common to all metered musics is the fact that the understanding of only one metrical level, such as the beat in Eurogenetic music, leads to an inferior understanding of the musical structure compared to an interpretation on several metrical layers; a couple dancing a Ballroom dance without a common understanding of beat and bar level will end up with four badly bruised feet, while a whirling dervish in Turkey who does not follow the long-term structure of the rhythmic mode will suffer pain of a rather spiritual kind.

Within the field of Music Information Research (MIR), the task of beat tracking has been approached by many researchers, using a large variety of methodologies, see the summary in [14]. Tracking of meter, i.e., tracking on several hierarchically related time-spans, was pursued by a smaller number of approaches, for instance by [9]. [15] were among the first to include experiments that document the importance of adapting a model automatically to musical styles in the context of meter tracking. In recent years, several approaches to beat and meter tracking were developed that include such adaptation to musical style, for instance by applying dynamic Bayesian networks [12] or Convolutional Neural Networks (CNN) [6] for meter tracking, or by combining Bayesian networks with Recurrent Neural Networks (RNN) for beat tracking in [1].

In this paper, we combine deep neural network and Bayesian approaches for meter tracking. To this end, we adapt an approach based on CNN that was previously applied to music segmentation with great success [18]. To the best of our knowledge, no other applications of CNNs to the task of combined tracking at several metrical levels have yet been published, although other groups apply CNN as well [6]. In this paper, the outputs of the CNN, i.e., the activations that imply probabilities of observing beats and downbeats [1], are then integrated as observations into a dynamic Bayesian network. This way, we explore in how far an approach [18] previously applied to supra-metrical

AH is supported by the Austrian Science Fund (FWF: M1995-N31).
TG is supported by the Vienna Science and Technology Fund (WWTF) through project MA14-018 and the Federal Ministry for Transport, Innovation & Technology (BMVIT, project TRP 307-N23).
We would like to thank Ajay Srinivasamurthy for advice and comments. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of a Tesla K40 GPU used for this research.

© Andre Holzapfel, Thomas Grill. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Andre Holzapfel, Thomas Grill. "Bayesian meter tracking on learned signal representations", 17th International Society for Music Information Retrieval Conference, 2016.

[1] We use these terms to denote the two levels, for the sake of simplicity.

Dance	#Pieces	Cycle Length: mean (std)
Cha cha (4/4)	111	1.96 (0.107)
Jive (4/4)	60	1.46 (0.154)
Quickstep (4/4)	82	1.17 (0.018)
Rumba (4/4)	98	2.44 (0.274)
Samba (4/4)	86	2.40 (0.177)
Tango (4/4)	86	1.89 (0.064)
Viennese Waltz (3/4)	65	1.01 (0.015)
Waltz (3/4)	110	2.10 (0.077)

Table 1: The Ballroom dataset. The columns depict time signature with the names of the dances, number of pieces/excerpts, and mean and standard deviation of the metrical cycle lengths in seconds.

Carnatic		
Tāḷa	#Pieces	Cycle Length: mean (std)
Adi (8/4)	50	5.34 (0.723)
Rūpaka (3/4)	50	2.13 (0.239)
Miśra chāpu (7/4)	48	2.67 (0.358)
Khanda chāpu (5/4)	28	1.85 (0.284)
Hindustani		
Tāl	#Pieces	Cycle Length: mean (std)
Tintāl (16/4)	54	10.36 (9.875)
Ektāl (12/4)	58	30.20 (26.258)
Jhaptāl (10/4)	19	8.51 (3.149)
Rūpak tāl (7/4)	20	7.11 (3.360)

Table 2: The Indian music dataset. The columns depict time signature with the names of the Tāḷa/tāl cycles, the number of pieces/excerpts, and mean and standard deviation of the metrical cycle lengths in seconds.

structure in music can serve to perform meter tracking as well. Furthermore, we want to evaluate in how far the meter tracking performed by the CNN can be further improved by imposing knowledge of metrical structure that is expressed using a Bayesian model. The evaluation in this paper is performed on Indian musics as well as Latin and international Ballroom dances. This choice is motivated by the fact that meter tracking in Indian musics revealed to be particularly challenging [8], but at the same time a novel approach should generalize to non-Indian musics. Our results improve over the state of the art in meter tracking on Indian music, while results on Ballroom music are highly competitive as well.

We present the used music corpora in Section 2. Section 3 provides detail on the CNN structure and training, and Section 4 on the Bayesian model and its combination with the CNN activations. In both sections we aim at providing a concise presentation of both methods, emphasizing the novel elements compared to previously published approaches. Section 5 illustrates our findings, and Section 6 provides a summary and directions for future work.

2. MUSIC CORPORA

For the evaluation of meter tracking performance, we use two different music corpora. The first corpus consists of 697 monaural excerpts ($f_s = 44.1$ kHz) of Ballroom dance music, with a duration of 30 s for each excerpt. The corpus was first presented in [5], and beat and bar annotations were compiled by [10]. Table 1 lists all the eight contained dance styles and their time signatures, and depicts the mean durations of the metrical cycles and their standard deviations in seconds. In general, the bar durations can be seen to have a range from about a second (Viennese Waltz) to 2.44 s (Rumba), with small standard deviations.

The second corpus unites two collections of Indian art music that are outcomes of the ERC project CompMusic. The first collection, the Carnatic music rhythm corpus contains 176 performance recordings of South Indian Carnatic music, with a total duration of more than 16 hours. [2] The second collection, the Hindustani music rhythm corpus, contains 151 excerpts of 2 minutes length each, summing up to a total duration of a bit more than 5 hours. [3] All samples are monaural at $f_s = 44.1$ kHz. Within this paper we unite these two datasets to one corpus, in order to obtain a sufficient amount of training data for the neural networks described in Section 3. This can be justified by the similar instrumental timbres that occur in these datasets. However, we carefully monitor the differences of tracking performance for the two musical styles. As illustrated in Table 2, metrical cycles in the Indian musics have longer durations with large standard deviations in most cases. This difference is in particular accentuated for Hindustani music, where, for instance, the Ektāl cycles range from 2.23 s up to a maximum of 69.73 s. This spans five tempo octaves and represents a challenge for meter tracking. The rhythmic elaboration of the pieces within a metrical class varies strongly depending on the tempo, which is likely to create difficulties when using the recordings in these classes for training one unified tracking model.

3. CNN FOR METER TRACKING

CNNs are feed-forward networks that include *convolutional layers*, computing a convolution of their input with small learned filter kernels of a given size. This allows processing large inputs with few trainable parameters, and retains the input's spatial layout. When used for binary classification, the network usually ends in one or more dense layers integrating information over the full input at once, discarding the spatial layout. The architecture for this work is based on the one used by Ullrich et al. [18] on MLS (Mel-scaled log-magnitude spectrogram) features for their MIREX submission [16]. Therein, CNN-type networks have been employed for the task of musical structure segmentation. [7] have expanded on this approach by introducing two separate output units, yielding predictions for 'fine' and 'coarse' segment boundaries. For the research at hand, we can use this architecture to train and predict

[2] http://compmusic.upf.edu/carnatic-rhythm-dataset

[3] http://compmusic.upf.edu/hindustani-rhythm-dataset

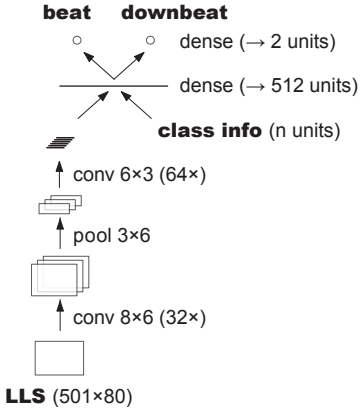


Figure 1: The CNN architecture in use.

beats and downbeats in the same manner with two output units, enabling the network to exploit information shared between these two temporal levels.

3.1 Data

For both datasets under examination, we use a train/ validation/test split. The sizes are 488/70/140 for the ballroom data set and 228/33/66 for the combination of the two Indian data sets. From the audio files, we compute log-scaled logarithmic-magnitude spectrograms (LLS) of 80 bands (instead of mel-scaled MLS in [18]), ranging from 80 Hz to 16 kHz. We have found log-scaled features to work better in early stages of research, most probably because of their harmonic translational invariance, supporting the convolutional filters. The STFT size used is 2048, with a frame rate of 100 fps. In order to be able to train and predict on spectrogram excerpts near the beginning and ending of a music piece, we apply a simple padding strategy for the LLS features. If the first (or last, respectively) non-zero spectrogram frame has a mean volume of ≥ -40 dBFS, we assume an abrupt boundary and pad the spectrogram with a -100 dBFS constant. Conversely, we pad with repeated copies of this first or last non-zero spectrogram frame. To either padding, we add ± 3 dB of uniform noise to avoid unnatural spectral clarity. Over the entire data sets, we normalize to zero mean and unit variance for each frequency band, yielding a suitable range of input values for the CNN.

3.2 Network Structure and training

Figure 1 shows the network architecture used for our experiments, unchanged from our previous experiments in [18]. On the input side, the CNN sees a temporal window of 501 frames with 80 frequency bands, equivalent to 5 seconds of spectral information. The LLS input is subjected to a convolutional layer of 32 parallel 8×6 kernels (8 time frames and 6 frequency bands), a max-pooling layer with pooling factors of 3×6, and another convolution of 64 parallel 6×3 kernels. Both convolutional layers employ linear rectifier units. While the first convolution emphasizes certain low-level aspects of the time-frequency patches it processes (for example the contrast between patches), the subsequent pooling layer spatially condenses both dimensions. This effectively expands the scope with regard to the input features for the second convolution. The resulting learned features are fed into a dense layer of 512 sigmoid units encoding the relevance of individual feature components of the time-frequency window and the contribution of individual convolutional filters. Finally, the network ends in a dense output layer with two sigmoid units. Additionally, the class information (Indian tāḷa/tāl class or ballroom style class, which can generally be assumed to be known) is fed through one-hot coding directly to the first dense layer. Using this class information improves results in the range of 1–2%.

During training, the beat and downbeat units are tied to the target information from the ground-truth annotations using a binary cross-entropy loss function. The targets are set to one with a tolerance window of 5 frames, equivalent to 50 milliseconds, around the exact location of the beat or downbeat. Training weights decline according to a Gaussian window around this position ('target smearing'). Training is done by mini-batch stochastic gradient descent, using the same hyper-parameters and tweaks as in [18]. The dense layers use dropout learning, updating only 50% of the weights per training step.

3.3 Beat and downbeat prediction

In order to obtain beat and downbeat estimations from a trained CNN, we follow the basic peak-picking strategy described in [18] to retrieve likely boundary locations from the network output. Note that the class information is provided in the same way as in the training, which means that we assume the meter type (e.g., 7/4) known, and target the tracking of the given metrical hierarchy. The adjustable parameters for peak picking have been optimized on the validation set. Several individual network models have been trained individually from random initializations, yielding slightly different predictions. Differently than in [18] we did not 'bag' (that is, average) multiple models, but rather selected the model with the best results as evaluated on the validation set. Although the results directly after peak picking are inferior to bagged models by up to 3%, the Bayesian post-processing works better on non-averaged network outputs, as also tested on the validation set. The CNN output vectors that represents the beat probability will be referred to as $P(b)$, and the vector representing the downbeat probabilities as $P(d)$, respectively. The results obtained from the peak picking on these vectors will be denoted as **CNN-PP**.

4. METER TRACKING USING BAYESIAN NETWORKS

The Bayesian network used for meter tracking is an extension of the model presented in [11]. Within the model in [11], activations from RNN were used as observations in a Bayesian network for beat tracking in music, whereas in this paper we extend the approach to the tracking of a metrical cycle. We will shortly summarize the principle of the

algorithm presented in [11] in Section 4.1. In Section 4.2, we present the extension of the existing approach to meter tracking using activations from a CNN.

4.1 Summary: A Bayesian meter tracking model

The underlying concept of the approach presented in [11] is an improvement of [8], and was first described by [19] as the bar pointer model. In [11], given a series of observations/features $\mathbf{y}_k$, with $k \in \{1, ..., K\}$, computed from a music signal, a set of hidden variables x_k is estimated. The hidden variables describe at each analysis frame k the position Φ_k within a beat (in the case of beat tracking) or within a bar (in the case of meter tracking), and the tempo in positions per frame ($\dot{\Phi}_k$). The goal is to estimate the hidden state sequence that maximizes the posterior (MAP) probability $P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$. If we express the temporal dynamics as a Hidden Markov Model (HMM), the posterior is proportional to

$$P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K}) \propto P(\mathbf{x}_1) \prod_{k=2}^{K} P(\mathbf{x}_k|\mathbf{x}_{k-1}) P(\mathbf{y}_k|\mathbf{x}_k) \quad (1)$$

In (1), $P(\mathbf{x}_1)$ is the *initial state distribution*, $P(\mathbf{x}_k|\mathbf{x}_{k-1})$ is the *transition model*, and $P(\mathbf{y}_k|\mathbf{x}_k)$ is the *observation model*. When discretizing the hidden variable $\mathbf{x}_k = [\Phi_k, \dot{\Phi}_k]$, the inference in this model can be performed using the Viterbi algorithm. In this paper, for the sake of simplicity of representation we do not apply approximate inference, as for instance in [17], but strictly follow the approach in [11].

In [11], efficiency of the inference was improved by a flexible sampling of the hidden variables. The position variable Φ_k takes $M(T)$ values $1, 2, ..., M(T)$, with

$$M(T) = round\left(\frac{N_{beats} * 60}{T * \Delta}\right) \quad (2)$$

where T denotes the tempo in beats per minute (bpm), and Δ the analysis frame duration in seconds. In the case of meter tracking, N_{beats} denotes the number of beats in a measure (e.g., nine beats in a 9/8), and is set to 1 in the case of beat tracking. This sampling results in one position state per analysis frame. The discretized tempo states $\dot{\Phi}_k$ were distributed logarithmically between a minimum tempo T_{min} and a maximum tempo T_{max}.

As in [11], a uniform initial state distribution $P(\mathbf{x}_1)$ was chosen in this paper. The transition model factorizes into two components according to

$$P(\mathbf{x}_k|\mathbf{x}_{k-1}) = P(\Phi_k|\Phi_{k-1}, \dot{\Phi}_{k-1}) P(\dot{\Phi}_k|\dot{\Phi}_{k-1}) \quad (3)$$

with the two components describing the transitions of position and tempo states, respectively. The position transition model increments the value of Φ_k deterministically by values depending on the tempo $\dot{\Phi}_{k-1}$, starting from a value of 1 (at the beginning of a metrical cycle) to a value of $M(T)$. The tempo transition model allows for tempo transitions according to an exponential distribution in exactly the same way as described in [11].

We incorporated the *GMM-BarTracker* (**GMM-BT**) as described in [11] as a baseline in our paper. The observation model in the *GMM-BarTracker* divides a whole note into 64 discrete bins, using the beat and downbeat annotations that are available for the data. For instance, a 5/4 meter would be divided into 80 metrical bins, and we denote this number of bins within a specific meter as N_{bins}. Spectral-flux features obtained from two frequency bands, computed as described in [12], are assigned to one of these metrical bins. Then, the parameters of a two-component Gaussian Mixture Model (GMM) are determined in exactly the same way as documented in [12], using the same training data as for the training of the CNN in Section 3.1. Furthermore, the fastest and the slowest pieces were used to determine the tempo range T_{min} to T_{max}. A constant number of 30 tempo states were used, a denser sampling did not improve tracking on any of the validation sets.

4.2 Extension of the Bayesian network: CNN observations

The proposed extensions of the GMM-BT approach affect the observation model $P(\mathbf{y}_k|\mathbf{x}_k)$, as well as the parametrization of the state space. We will refer to this novel model as **CNN-BT**.

Regarding the observation model, we incorporate the beat and downbeat probabilities $P(b)$ and $P(d)$, respectively, obtained from the CNN as described in Section 3. Network activations were incorporated in [11] on the beat level only, and in this paper our goal is to determine in how far the downbeat probabilities can help to obtain an accurate tracking not only of the beat, but the entire metrical cycle. Let us denote the metrical bins that are beat instances by $\mathcal{B}$ (excluding the downbeat), and the downbeat position as $\mathcal{D}$. Then we calculate the observation model $P(\mathbf{y}_k|\mathbf{x}_k)$ as follows

$$P(\mathbf{y}_k|\mathbf{x}_k) = \begin{cases} P_k(d) * P_k(b), & \Phi_k \in \mathcal{D}, \mathcal{D}+1; \\ P_k(b) * (1 - P_k(d)) & \Phi_k \in \mathcal{B}, \mathcal{B}+1; \\ (1 - P_k(b)) * (1 - P_k(d)) & else; \end{cases} \quad (4)$$

Including the bin that follows a beat and downbeat was found to slightly improve the performance on the evaluation data. In simple terms, the network outputs $P(b)$ and $P(d)$ are directly plugged into the observation model. The two separate probabilities for beats and downbeats combined according to the metrical bin. For instance, downbeats are also instances of the beat layer, and at these positions the activities are multiplied in the first row of (4). The columns of the obtained observation matrix of size $N_{bins} \times K$ are then normalized to sum to one.

The CNN activations $P(b)$ and $P(d)$ are characterized by clearly accentuated peaks in the vicinity of beats and downbeats, as will be illustrated in Section 5. We take advantage of this property in order to restrict the number of possible tempo hypotheses $\dot{\Phi}_k$ in the state space of the model. To this end, the autocorrelation function (ACF) of the beat activation function $P(b)$ is computed, and the highest peak at tempi smaller than 500 bpm is determined. This peak serves as an initial tempo hypothesis

T_0, and we define T_{min}=0.4∗T_0 and T_{max}=2.2∗T_0, in order to include half and double tempo as potential tempo hypotheses into the search space. Then we determine the peaks of the ACF in that range, and if their number is higher than 5, we choose the 5 highest peaks only. This way we obtain N_{hyp} tempo hypotheses, covering T_0, its half and double value (in case the ACF has peaks at these values), as well as possible secondary tempo hypotheses. These peaks are then used to determine the number of position variables at these tempi according to (2). In order to allow for tempo changes around these modes, we include for a mode T_n, n∈{1,...,N_{hyp}}, all tempi related to $M(T_n)-3$,$M(T_n)-2$,...,$M(T_n)+3$. This means that for each of the N_{hyp} tempo modes we use seven tempo samples with the maximum possible accuracy at a given analysis frame rate Δ, resulting in a total of at most 35 tempo states (for N_{hyp}=5). Using more modes or more tempo samples per mode did not result in higher accuracy on the validation data. While this focused tempo space has not been observed to lead to large improvements over a logarithmic tempo distribution between T_{min} and T_{max}, the more important consequence is a more efficient inference. As will be shown in Section 5, metrically simple pieces are characterized by only 2 peaks in the ACF between T_{min} and T_{max}, which leads to a reduction of the state space size by more than 50% over the GMM-BT.

5. SYSTEM EVALUATION

5.1 Evaluation measures

We use three evaluation measures in this paper [4]. For **F-measure** (0% to 100%), estimations are considered accurate if they fall within a ±70 ms tolerance window around annotations. Its value is measured as a function of the number of true and false positives and false negatives. **AMLt** (0% to 100%) is a continuity-based method, where beats are accurate when consecutive beats fall within tempo-dependent tolerance windows around successive annotations. Beat sequences are also accurate if the beats occur on the off-beat, or are at double or half the annotated tempo. Finally, **Information Gain (InfG)** (0 bits to approximately 5.3 bits) is determined by calculating the timing errors between an annotation and all beat estimations within a one-beat length window around the annotation. Then, a beat error histogram is formed from the resulting timing error sequence. A numerical score is derived by measuring the K-L divergence between the observed error histogram and the uniform case. This method gives a measure of how much information the beats provide about the annotations.

Whereas the F-measure does not evaluate the continuity of an estimation, the AMLt and especially the InfG measure penalize random deviations from a more or less regular underlying beat pulse. Because it is not straight-forward to apply such regularity constraints on the downbeat level, downbeat evaluation is done using the F-measure only, denoting the F-measure at the downbeat and beat levels as $F(d)$ and $F(b)$, respectively.

Evaluation Measure	$F(d)$	$F(b)$	AMLt	InfG
CNN-PP	54.29	75.15	60.80	1.820
GMM-BT	63.84	77.00	74.92	1.942
CNN-BT	**69.93**	80.75	**87.46**	**2.314**
CNN-BT (T_{ann})	73.63	85.27	89.22	2.499

Table 3: Results on Indian music.

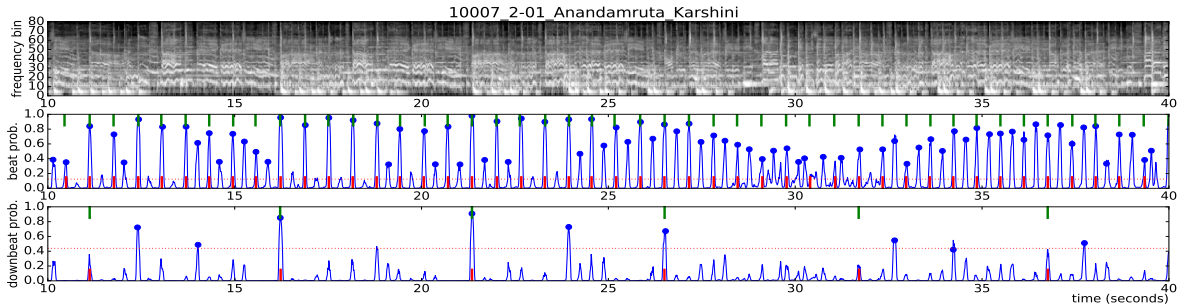
Evaluation Measure	$F(d)$	$F(b)$	AMLt	InfG
CNN-PP	79.30	93.59	88.98	3.216
GMM-BT	77.51	90.67	91.45	2.961
CNN-BT	**89.63**	93.74	**93.89**	3.244
CNN-BT (T_{ann})	90.67	94.81	94.25	3.240

Table 4: Results on Ballroom music.

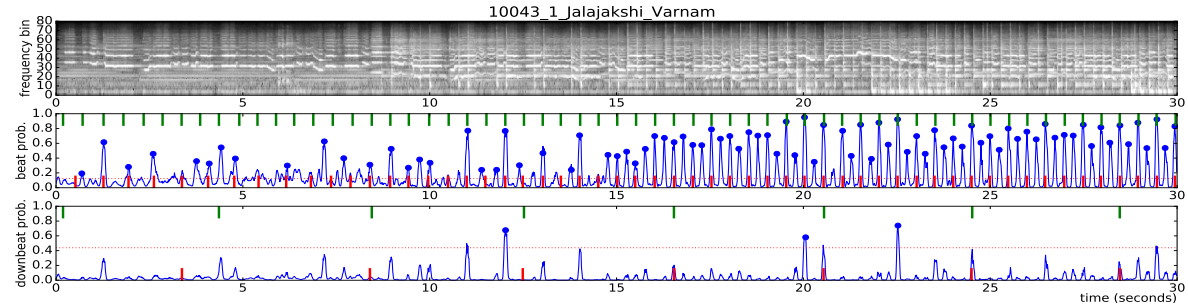
5.2 Results

Results are presented separately for the Indian and the Ballroom datasets in Tables 3 and 4, respectively. The first two columns represent F-scores for downbeats ($F(d)$) and beats ($F(b)$), followed by AMLt and InfG. We evaluated CNNs with subsequent peak-picking on the network activations (CNN-PP) as explained in Section 3, the Bayesian network from [11] using Spectral Flux in its observation model (GMM-BT), and the Bayesian network that incorporates the novel observation model obtained from CNN activations (CNN-BT). Bold numbers indicate significant improvement of CNN-BT over CNN-PP, underlining indicates significant improvement of CNN-BT over GMM-BT. Paired-sample t-tests were performed with a 5% significance level. Performing a statistical test over both corpora reveals a significant improvement by CNN-BT over CNN-PP for all measures, and for $F(d)$ and AMLt over GMM-BT. These results demonstrate that beat and downbeat estimations obtained from a CNN can be further improved using a Bayesian model that incorporates hypotheses about metrical regularity and the dynamic development of tempo. On the other hand, employing CNN activations yields significant improvements over the Bayesian model that incorporates hand-crafted features (Spectral Flux).

Figure 2 visualizes the improvement of CNN-BT over CNN-PP by depicting the network outputs along with reference annotations, and beat and downbeat estimations from CNN-BT and CNN-PP. It is apparent that the Bayesian network finds a consistent path through the pieces that is supported by the network activations as well as by the underlying regular metrical structure. Both figures depict examples of Carnatic Adi tāḷa, which has a symmetric structure that caused tempo halving/doubling errors when using spectral flux features as in GMM-BT [8]. In Figure 2a, the spectrogram, especially in the first two depicted cycles, is characterized by a similar melodic progression that marks the cycle. The CNN is able to capture such regularities, leading to an improved performance. In Figure 2b, the music provides no clear metrical cues in the beginning, but the output of the CNN-BT can be seen to be nicely synchronized from the third cycle on (at about 8 s), demonstrating the advantage of the regularity imposed by the Bayesian network.



(a) Indian music example 1



(b) Indian music example 2

Figure 2: Input LLS features and network outputs for beat (upper curve) and downbeat (lower curve) predictions for two music examples. Ground-truth positions as green vertical marks on top, peak-picking thresholds as red dotted lines, picked peaks from the CNN-PP as blue circle markers, and final predictions by the Bayesian tracking (CNN-BT) as red vertical marks on the bottom.

Corpus	Ballroom	Carnatic	Hindustani
Correct tempo (%)	97.1	100	81.8
ACF-peaks	2.67	3.60	4.15

Table 5: Some characteristics of the focused state space in CNN-BT. The first row depicts the percentage of pieces for which the true tempo was between T_{min}=0.4∗T_0 to T_{max}=2.2∗T_0 that was selected using the autocorrelation function (ACF) of $P(b)$. The second row depicts the number of peaks in the ACF in the selected tempo range.

In Table 5, we depict some characteristics of the tempo states that are chosen in the CNN-BT, as described in Section 4.2. We depict the Carnatic and Hindustani musics separately in order to illustrate differences. It can be seen that the true tempo is almost always in the chosen range from T_{min} to T_{max} for Ballroom and Carnatic music, but drops to 81.8% for Hindustani music. Furthermore, the number of peaks in the ACF of $P(b)$ is lowest for the Ballroom corpus, while the increased number for the Hindustani music indicates an increased metrical complexity for this style. Indeed, the performance values are generally lower for Hindustani musics than for Carnatic musics, with, for instance, the downbeat F-measure $F(d)$ being 0.76 for Carnatic, and 0.64 for Hindustani musics. This is to some extent related to the extremely low tempi that occur in Hindustani music, which cause the incorrect tempo ranges for Hindustani depicted in Table 5.

The last rows in Tables 3 and 4 depict the performance that is achieved when the correct tempo T_{ann} is given in CNN-BT. To do this evaluation, we use 30 logarithmically-spaced tempo coefficients in a range of $\pm 20\%$ around T_{ann}, in order to allow for gradual tempo changes, excluding, however, double and half tempo. For the Ballroom corpus, only marginal improvement can be observed, with none of the changes compared to the non-informed CNN-BT case being significant. For the Indian data the improvement is larger, however, again not significantly. This illustrates that even a perfect tempo estimation cannot further improve the results. The reasons for this might be, especially for Hindustani music, the large variability within the data due to the huge tempo ranges. The CNNs are not able to track pieces at extreme slow tempi, due to their limited temporal horizon of 5 seconds – slightly shorter than the beat period in the slowest pieces. However, further increasing this horizon was found to generally deteriorate the results, due to more network weights to learn with the same, limited amount of training data.

6. DISCUSSION

In this paper, we have combined CNNs and Bayesian networks for the first time in the context of meter tracking. Results clearly indicate the advantage of this combination that results from the flexible signal representations obtained from CNNs with the knowledge of metrical progression incorporated into a Bayesian model. Furthermore, the clearly accentuated peaks in the CNN activations enable us to restrict the state space in the Bayesian model to certain tempi, thus reducing computational complexity depending on the metrical complexity of the musical signal. Limitations of the approach can be seen in the ability to track very long metrical structures in Hindustani music. To this end, the incorporation of RNN will be evaluated in the future.

7. REFERENCES

[1] Sebastian Böck, Florian Krebs, and Gerhard Widmer. A multi-model approach to beat tracking considering heterogeneous music styles. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 602–607, Taipei, Taiwan, 2014.

[2] Baris Bozkurt, Ruhi Ayangil, and Andre Holzapfel. Computational analysis of makam music in Turkey: Review of state-of-the-art and challenges. *Journal for New Music Research*, 43(1):3–23, 2014.

[3] Martin Clayton. *Time in Indian Music : Rhythm , Metre and Form in North Indian Rag Performance*. Oxford University Press, 2000.

[4] M. E. P. Davies, N. Degara, and M. D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. Technical Report C4DM-TR-09-06, Queen Mary University of London, Centre for Digital Music, 2009.

[5] S. Dixon, F. Gouyon, and G. Widmer. Towards characterisation of music via rhythmic patterns. In *Proceedings of International Conference on Music Information Retrieval*, pages 509–516, 2004.

[6] Simon Durand, Juan Pablo Bello, Bertrand David, and Ga el Richard. Feature adapted convolutional neural networks for downbeat tracking. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing. ICASSP*, 2016.

[7] Thomas Grill and Jan Schlüter. Music Boundary Detection Using Neural Networks on Combined Features and Two-Level Annotations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Malaga, Spain, 2015.

[8] Andre Holzapfel, Florian Krebs, and Ajay Srinivasamurthy. Tracking the "odd": Meter inference in a culturally diverse music corpus. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 425–430, Taipei, Taiwan, 2014.

[9] A. P. Klapuri, A. J. Eronen, and J. T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech and Language Processing*, 14(1):342–355, 2006.

[10] Florian Krebs, Sebastian Böck, and Gerhard Widmer. Rhythmic pattern modeling for beat- and downbeat tracking in musical audio. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, 2013.

[11] Florian Krebs, Sebastian Böck, and Gerhard Widmer. An efficient state-space model for joint tempo and meter tracking. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Malaga, Spain, 2015.

[12] Florian Krebs, Andre Holzapfel, Ali Taylan Cemgil, and Gerhard Widmer. Inferring metrical structure in music using particle filters. *IEEE Transactions on Audio, Speech and Language Processing*, 23(5):817–827, 2015.

[13] Donna Lee Kwon. *Music in Korea : experiencing music, expressing culture*. Oxford University Press, 2011.

[14] Meinard Müller, Daniel P. W. Ellis, Anssi Klapuri, and Gaël Richard. Signal processing for music analysis. *J. Sel. Topics Signal Processing*, 5(6):1088–1110, 2011.

[15] Geoffroy Peeters and Helene Papadopoulos. Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation. *IEEE Transactions on Audio, Speech and Language Processing*, 19(6):1754–1769, 2011.

[16] Jan Schlüter, Karen Ullrich, and Thomas Grill. Structural segmentation with convolutional neural networks mirex submission. In *Tenth running of the Music Information Retrieval Evaluation eXchange (MIREX 2014)*, 2014.

[17] Ajay Srinivasamurthy, Andre Holzapfel, Ali Taylan Cemgil, and Xavier Serra. Particle filters for efficient meter tracking with dynamic bayesian networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Malaga, Spain, 2015.

[18] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary Detection in Music Structure Analysis using Convolutional Neural Networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, Taiwan, 2014.

[19] N. Whiteley, A. T. Cemgil, and S. J. Godsill. Bayesian modelling of temporal structure in musical audio. In *Proceedings of International Conference on Music Information Retrieval*, Victoria, Canada, 2006.

TEMPO ESTIMATION FOR MUSIC LOOPS AND A SIMPLE CONFIDENCE MEASURE

Frederic Font and Xavier Serra
Music Technology Group, Universitat Pompeu Fabra
frederic.font@upf.edu, xavier.serra@upf.edu

ABSTRACT

Tempo estimation is a common task within the music information retrieval community, but existing works are rarely evaluated with datasets of music loops and the algorithms are not tailored to this particular type of content. In addition to this, existing works on tempo estimation do not put an emphasis on providing a confidence value that indicates how reliable their tempo estimations are. In current music creation contexts, it is common for users to search for and use loops shared in online repositories. These loops are typically not produced by professionals and lack annotations. Hence, the existence of reliable tempo estimation algorithms becomes necessary to enhance the reusability of loops shared in such repositories. In this paper, we test six existing tempo estimation algorithms against four music loop datasets containing more than 35k loops. We also propose a simple and computationally cheap confidence measure that can be applied to any existing algorithm to estimate the reliability of their tempo predictions when applied to music loops. We analyse the accuracy of the algorithms in combination with our proposed confidence measure, and see that we can significantly improve the algorithms' performance when only considering music loops with high estimated confidence.

1. INTRODUCTION

Tempo estimation is a topic that has received considerable attention within the music information retrieval (MIR) community and has had a dedicated task in the Music Information Retrieval Evaluation eXchange (MIREX) since its first edition in 2005. Tempo estimation consists in the automatic determination of the "rate of musical beats in time" [10], that is to say, in the identification of the rate at which periodicities occur in the audio signal that convey a rhythmic sensation. Tempo is typically expressed in beats per minute (BPM), and is a fundamental property to characterise rhythm in music [13]. Applications of tempo estimation include, just to name a few, music recommendation, music remixing, music browsing, and beat-aware audio analysis and effects.

© Frederic Font and Xavier Serra. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Frederic Font and Xavier Serra. "Tempo Estimation for Music Loops and a Simple Confidence Measure", 17th International Society for Music Information Retrieval Conference, 2016.

Our particular research is aimed at automatically annotating user provided music loops hosted in online sound sharing sites to enhance their potential reusability in music creation contexts. We can define music loops as short music fragments which can be repeated seamlessly to produce an "endless" stream of music. In this context, BPM is an important music property to annotate. The kind of music loops we are targeting can include noisy and low quality content, typically not created by professionals. This may increase the difficulty of the tempo estimation task. Taking that into consideration, it is particularly relevant for us to not only estimate the tempo of music loops, but to also quantify how reliable an estimation is (i.e., to provide a confidence measure). Except for the works described in [10, 14] (see below), tempo estimation has been rarely evaluated with datasets of music loops, and we are not aware of specific works describing algorithms that are specifically tailored to this particular case.

In this paper we evaluate the accuracy of six state of the art tempo estimation algorithms when used to annotate four different music loop datasets, and propose a simple and computationally cheap confidence measure that can be used in combination with any of the existing methods. The confidence measure we propose makes the assumption that the audio signal has a steady tempo thorough its whole duration. While this assumption can be safely made in the case of music loops, it does not necessarily hold for other types of music content such as music pieces. Hence, the applicability of the confidence measure we propose is restricted to music loops. Using our confidence measure in combination with existing tempo estimation algorithms, we can automatically annotate big datasets of music loops and reach accuracies above 90% when only considering content with high BPM estimation confidence. Such reliable annotations can allow music production systems to, for example, present relevant loops to users according to the BPM of a music composition, not only by showing loops with the same BPM but also by automatically transforming loops to match a target BPM. This effectively increases the reusability of user provided music loops in real-world music creation contexts.

The rest of the paper is organised as follows. In Sec. 2 we give a quick overview of related work about tempo estimation. In Sec. 3 we describe the confidence measure that we propose. Sections 4 and 5 describe the evaluation methodology and show the results of our work, respectively. We end this paper with some conclusions in Sec. 6.

In the interest of research reproducibility, the source code and one of the datasets used in this paper have been made available online in a public source code repository [1] .

2. RELATED WORK

A significant number of works within the MIR research field have been focused on the task of tempo estimation. In general, tempo estimation algorithms are based on detecting onsets in an audio signal, either as a continuous function [3, 14, 15] or as discrete events in time [5]. Then, a dominant period is extracted from the onsets either by analysing inter-onset intervals, using autocorrelation [11] or resonating filters [12]. Some approaches perform more complex operations such as analysing periodicities in different frequency bands [8, 19], performing source separation [6, 9], or using neural networks to learn features to use instead of usual onset information [1].

While comparative studies of tempo estimation algorithms have been carried out in the past [10, 21], we are not aware of any study solely devoted to the evaluation of tempo estimation algorithms for music loops. One of the typical datasets that some of the existing tempo estimation works use for evaluation is the ISMIR 2004 dataset released for the tempo induction contest of that year [2] . This dataset is divided into three subsets, one of them composed of 2k audio loops. Gouyon et. al. [10] published the evaluation results for the contest considering the different subsets of the dataset, but no significant differences are reported regarding the accuracies of the tempo estimation algorithms with the loops subset compared to the other subsets. To the best of our knowledge, the only other work that uses the loops subset of the ISMIR 2004 dataset and reports its accuracy separated form other datasets is by Oliveira et. al. [14]. The authors report lower estimation accuracies when evaluating with the loops dataset and attribute this to the fact that loops are typically shorter than the other audio signals (in many cases shorter than 5 seconds).

Surprisingly enough, there has not been much research on confidence measures for tempo estimation algorithms. Except for the work by Zapata et al. [22] in which a confidence measure that can be used for tempo estimation is described (see below), we are not aware of other works directly targeted at this issue. Among these few, Grosche and Müller [11] describe a confidence measure for their tempo estimation algorithm based on the amplitude of a predominant local pulse curve. By analysing tempo estimation accuracy and disregarding the regions of the analysis with bad confidence, the overall accuracy significantly increases. Alternatively, Percival and Tzanetakis [15] suggest that beat strength [18] can be used to derive confidence for tempo candidates, but no further experiments are carried out to asses its impact on the accuracy of tempo estimation. Finally, a very recent work by Quinton et. al. [16] proposes the use of rhythmogram entropy as a measure of reliability for a number of rhythm features, and report a statistical correlation between measured entropy and the resulting accuracies for different tasks.

[1] https://github.com/ffont/ismir2016

[2] http://mtg.upf.edu/ismir2004/contest/tempoContest

3. CONFIDENCE MEASURE

Assuming that we obtain a BPM estimate for a given audio signal, the confidence measure that we propose is based on comparing the duration of the whole audio signal with a multiple of the duration of a single beat according to the estimated BPM. If the actual duration of the signal is close to a multiple of the duration of a single beat, we hypothesise that the BPM estimation is reliable. The first thing we do to compute the confidence measure is to round the estimated tempo value to its nearest integer. The reasoning behind this is that it is very unlikely that loops are created with less than 1 BPM resolution tempo (see Sec. 4.1), and thus we consider the best BPM estimate of a tempo estimation algorithm to be its nearest integer. Given the sample rate SR of an audio signal and its estimated tempo BPM^e, we can estimate the duration (or length) of an individual beat in number of samples l^b as

$$l^b = \frac{60 \cdot SR}{BPM^e}.$$

Then, potential durations for the audio signal can be computed as multiples of the individual beat duration, $L[n] = n \cdot l^b$, where $n \in \mathbb{Z}^+$. In our computation, we restrict n to the range $1 \leq n \leq 128$. This is decided so that the range can include loops that last from only 1 beat to 128 beats, which would correspond to a maximum of 32 bars in 4/4 meter. In practice, what we need here is a number big enough such that we won't find loops longer than it. Given L, what we need to see at this point is if any of its elements closely matches the actual length of the original audio signal. To do that, we take the actual length of the audio signal l^a (in number of samples), compare it with all elements of L and keep the minimum difference found:

$$\Delta l = \min\{|L[n] - l^a| : n \leq 128\}.$$

A value of Δl near 0 means that there is a close match between one of the potential lengths and the actual length of the audio signal. Having computed Δl, we finally define our confidence measure as

$$confidence(L, l^a) = \begin{cases} 0 & \text{if } \Delta l > \lambda \\ 1 - \frac{\Delta l}{\lambda} & \text{otherwise} \end{cases},$$

where λ is a parameter set to half the duration of a single beat ($\lambda = 0.5 \cdot l^b$). In this way, if l^a exactly matches one of the multiples of l^b, the confidence will be 1. If Δl is as long as half the duration between beats, the confidence will be 0 (see Fig. 1, top).

The reasoning behind this simple confidence measure is that it is very unlikely that, only by chance, an audio signal has a duration which closely matches a multiple of the beat duration for a given estimated BPM. This means that we assume that there is a relation between the duration of the signal and its BPM, and therefore our proposed confidence will fail if the audio signal contains silence (either at

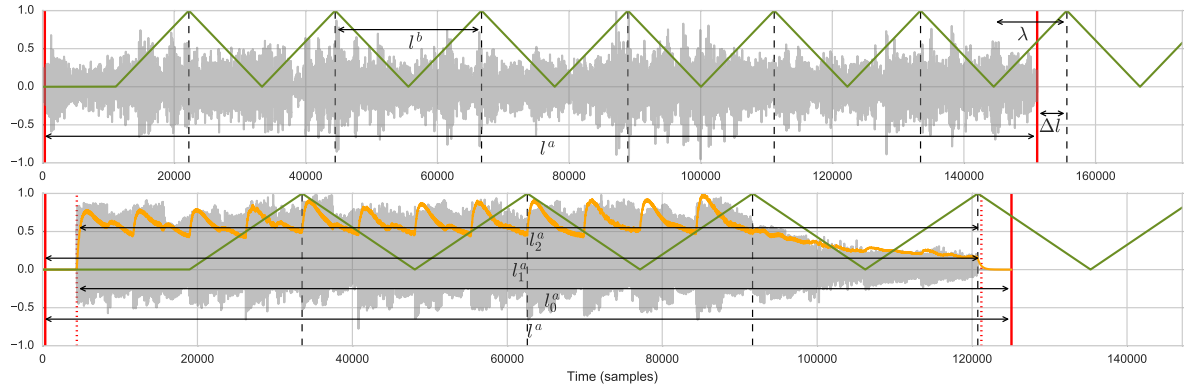


Figure 1. Visualisation of confidence computation output according to BPM estimation and signal duration (green curves). The top figure shows a loop whose annotated tempo is 140 BPM but the predicted tempo is 119 BPM. The duration of the signal l^a does not closely match any multiple of l^b (dashed vertical lines), and the output confidence is 0.59 (i.e., $1 - \frac{\Delta l}{\lambda}$). The figure at the bottom shows a loop that contains silence at the beginning and at the end, and for which tempo has been correctly estimated as being 91 BPM. The yellow curve represents its envelope and the vertical dashed red lines the estimated effective start and end points. Note that l_2^a closely matches a multiple of l^b, resulting in a confidence of 0.97. The output confidence computed with l^a, l_0^a and l_1^a produces lower values.

the beginning or at the end) which is not part of the loop itself (i.e., the loop is not *accurately cut*). To account for this potential problem, we estimate the duration that the audio signal would have if we removed silence at the beginning, at the end, or both at the beginning and at the end. We take the envelope of the original audio [3] and consider the effective starting point of the loop as being the point in time t_s where the envelope amplitude raises above 5% of the maximum. Similarly, we consider the effective end t_e at the *last point* where the envelope goes below the 5% of the maximum amplitude (or at the end of the audio signal if envelope is still above 5%). Taking t_s, t_e, and l^a (the original signal length), we can then compute three alternative estimates for the duration of the loop (l_0^a, l_1^a and l_2^a) by *i)* disregarding silence at the beginning ($l_0^a = l^a - t_s$), *ii)* disregarding silence at the end ($l_1^a = t_e$), and *iii)* disregarding silence both at the beginning and at the end ($l_2^a = t_e - t_s$). Then, we repeat the previously described confidence computation with the three extra duration estimates l_0^a, l_1^a and l_2^a. Note that these will produce meaningful results in cases where the original loop contains silence which is not relevant from a musical point of view, but they will not result in meaningful confidence values if the loop contains silence at the beginning or at the end which is in fact part of the loop (i.e., which is needed for it seamless repetition). Our final confidence value is taken as the maximum confidence obtained when using any of l^a, l_0^a, l_1^a and l_2^a estimated signal durations (see Fig. 1, bottom).

Because the confidence measure that we propose only relies on a BPM estimate and the duration of the audio signal, it can be used in combination with any existing tempo estimation algorithm. Also, it is computationally cheap to compute as the most complex operation it requires is the envelope computation. However, this confidence measure should not be applied to content other than music loops as it only produces meaningful results under the assumption that tempo is completely steady across the whole signal.

[3] We use the Envelope algorithm from the open-source audio analysis library Essentia [2], which applies a non-symmetric lowpass filter and rectifies the signal.

4. EVALUATION

4.1 Datasets

Our evaluation is conducted using 4 different datasets collected from different sources and containing a total of more than 35k loops. Table 1 shows basic statistics of each dataset. We now briefly describe each of the datasets:

- **FSL4**: This dataset contains user-contributed loops uploaded to Freesound [7]. It has been built in-house by searching Freesound for sounds with the query terms *loop* and *bpm*, and then automatically parsing the returned sound filenames, tags and textual descriptions to identify tempo annotations made by users. For example, a sound containing the tag *120bpm* is considered to have a ground truth of 120 BPM. Detailed instructions on how this dataset was created and on how can be reproduced are found in the source code repository (see Sec. 1).

- **APPL**: This dataset is composed of the audio loops bundled in Apple's Logic Pro [4] music production software. We parsed the metadata embedded in the audio files using source code available in a public repository [5], and extracted in this way tempo annotations for all the loops.

- **MIXL**: This dataset contains all the loops bundled with Acoustica's Mixcraft 7 music production software [6]. Tempo annotations are provided in its loop

[4] http://apple.com/logic-pro
[5] http://github.com/jhorology/apple-loops-meta-reader
[6] http://acoustica.com/mixcraft

Dataset	N instances	Total duration	Mean loop duration	Duration range	Tempo range	Source
FSL4	3,949	8h 22m	7.63s	0.15s - 30.00s	32 - 300	Freesound
APPL	4,611	9h 34m	7.47s	1.32s - 40.05s	53 - 140	Logic Pro
MIXL	5,451	14h 11m	9.37s	0.32s - 110.77s	55 - 220	Mixcraft 7
LOOP	21,226	50h 30m	8.57s	0.26s - 129.02s	40 - 300	Looperman

Table 1. Basic statistics about the datasets used for evaluation. Additional information and plots can be found in the paper's source code repository (see Sec. 1).

browser and can be easily exported into a machine-readable format.

- **LOOP**: This dataset is composed of loops downloaded from Looperman [7], an online loop sharing community. It was previously used for research purposes in [17]. Tempo annotations are available as metadata provided by the site.

Because of the nature of how the datasets were collected, we found that some of the loops do not have a BPM annotation that we can use as ground truth or have a BPM annotation which is outside what could be intuitively considered a reasonable tempo range. To avoid inconsistencies with the annotations, we clean the datasets by removing instances with no BPM annotation or with a BPM annotation outside a range of $[25, 300]$. Interestingly, we see that all the loops in our datasets are annotated with integer tempo values, meaning that it is not common for music loops to be produced with tempo values with less than 1 BPM resolution. For analysis purposes, all audio content from the dataset is converted to linear PCM mono signals with 44100 Hz sampling frequency and 16 bit resolution.

4.2 Tempo estimation algorithms

In our evaluation we compare six existing tempo estimation algorithms. These have been chosen based on their availability and to represent different approaches to the tempo estimation task. We now briefly describe each of the algorithms, further details on how the algorithms work can be found in corresponding papers.

- **Gkiokas12**: Gkiokas et. al. [9] propose a tempo estimation algorithm based on the separation of the audio signal into its percussive and harmonic components. Periodicity analysis is carried out by convolving extracted features (filterbank energies for the percussive component and chroma features for the harmonic component) with a bank of resonators. Output tempo value is computed by applying heuristics based on metrical relations knowledge (meter, tactus, tatum) to the periodicity vector. We use a Matlab implementation of the algorithm kindly provided to us by the authors.

- **Degara12**: Degara et. al. [4] describe a probabilistic approach for beat tracking based on inter-onset-interval times and a salience measure for individual beat estimates. This method builds from previous probabilistic beat tracking methods such as Klapuri et. al. [12]. We use the implementation provided in Essentia, where final estimated tempo is given based on the mean of estimated beat intervals (see RhythmExtractor2013 algorithm [8]).

- **Zapata14**: Zapata et. al. [20] propose a beat tracking algorithm which estimates beat positions based on computing the agreement of alternative outputs of a single model for beat tracking using different sets of input features (i.e., using a number of onset detection functions based on different audio features). Again, we use the implementation provided in Essentia, which outputs a single BPM estimate based on estimated beat intervals.

- **Percival14**: Percival and Tzanetakis [15] describe a tempo estimation algorithm optimised for low computational complexity that combines several ideas from existing tempo estimation algorithms and simplifies their steps. The algorithm computes an onset strength function based on filtered spectral flux from which tempo lag candidates are estimated using autocorrelation. The most prominent tempo lag is selected and a simple decision tree algorithm is used to chose the octave of the final BPM output. We use a Python implementation of the algorithm provided by the authors in their original paper [9].

- **Böck15**: Böck et. al. [1] propose a novel tempo estimation algorithm based on a recurrent neural network that learn an intermediate beat-level representation of the audio signal which is then feed to a bank of resonating comb filters to estimate the dominant tempo. This algorithm got the highest score in ISMIR 2015 Audio Tempo Estimation task. An implementation by the authors is included in the open-source Madmom audio signal processing library [10].

- **RekBox**: We also include an algorithm from a commercial DJ software, Rekordbox [11]. Details on how the algorithm works are not revealed, but a freely downloadable application is provided that can analyse a music collection and export the results in a machine-readable format.

[7] http://looperman.com

[8] http://essentia.upf.edu/documentation/algorithms_reference.html

[9] http://opihi.cs.uvic.ca/tempo

[10] http://github.com/CPJKU/madmom

[11] http://rekordbox.com

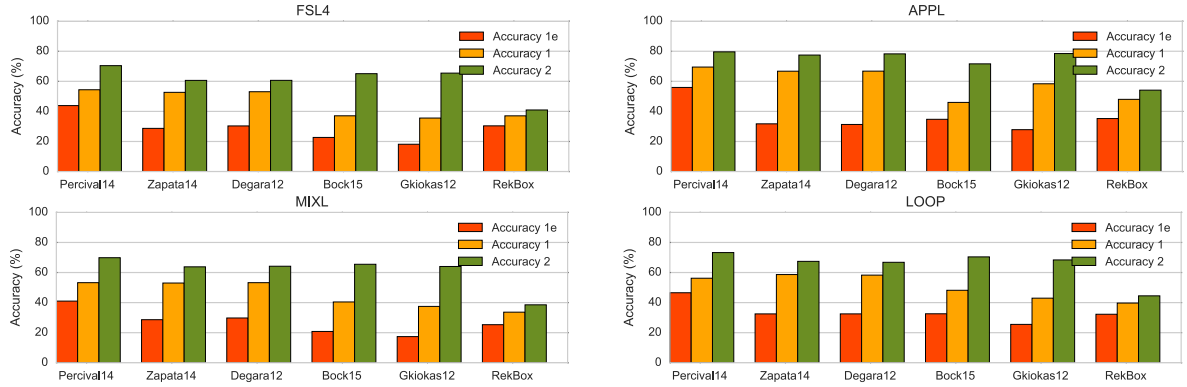


Figure 2. Overall accuracy for the six tempo estimation algorithms tested against the four datasets.

4.3 Methodology

For testing the above algorithms against the four collected datasets we follow standard practice and adopt the methodology described by Gouyon et al. [10]. In addition to the standard *Accuracy 1* and *Accuracy 2* measures [12], we add an extra measure that we call *Accuracy 1e* and that represents the percentage of instances whose estimated BPM is exactly the same as the ground truth after rounding the estimated BPM to the nearest integer. Accuracy 1e is therefore more strict than Accuracy 1. The reason why we added this extra accuracy measure is that, imagining a music creation context where loops can be queried in a database, it is of special relevance to get returned instances whose BPM exactly matches that specified as target.

Besides the overall accuracy measurements, we are also interested in observing how accuracy varies according to the confidence values that we estimate (Sec. 3). We can intuitively imagine that if we remove instances from our datasets whose estimated BPM confidence is below a certain threshold, the overall accuracy results will increase. However, the higher we set the minimum confidence threshold, the smaller the size of filtered dataset will be. Hence, we want to quantify the relation between the overall accuracy and the total number of music loops that remain in a dataset after filtering by minimum confidence. To do that, given one of the aforementioned accuracy measures, we can define a minimum confidence threshold γ and a function $A(\gamma)$ that represents overall BPM estimation accuracy when only evaluating loop instances whose estimated confidence value is above γ for a given dataset and tempo estimation algorithm. Similarly, we can define another function $N(\gamma)$ which returns the percentage of instances remaining in a dataset after filtering out those whose estimated confidence value (for a given tempo estimation method) is below γ. $A(\gamma)$ and $N(\gamma)$ can be understood as standard precision and recall curves, and therefore we can define a combined score measure $S(\gamma)$ doing the analogy with an f-measure computation:

$$S(\gamma) = 2 \cdot \frac{A(\gamma) \cdot N(\gamma)}{A(\gamma) + N(\gamma)}.$$

An overall score for a given dataset, tempo estimation algorithm and accuracy measure can thus be given by taking the mean of $S(\gamma)$, $\bar{S}$.

5. RESULTS

5.1 Overall accuracy

Overall accuracy results show that Percival14 obtains the highest accuracy scores for all accuracy measures and all datasets except for the LOOP dataset, in which highest score for Accuracy 1 is obtained by Zapata14 (Fig. 2). Considering the data from all datasets at once, mean accuracy values for Percival14 range from 47% (Accuracy 1e) to 73% (Accuracy 2), with an average increase of 7% accuracy when compared with the second best-scored method. With a few exceptions, pairwise accuracy differences between Percival14 and the second best-scored method in all datasets and accuracy measures are statistically significant using McNemar's test and a significance value of $\alpha = 0.01$ (i.e., $p \ll 0.01$). We also observe that accuracies for the APPL dataset tend to be higher than for other datasets. This can be explained by the fact that APPL contains professionally created and curated loops, while the other datasets contain user contributed content, not necessarily created by professionals (Mixcraft's loop library also contains content gathered from online repositories).

5.2 Accuracy vs confidence measure

Fig. 3 shows the accuracy of the three best-scoring tempo estimation algorithms and the number of instances remaining in the dataset when filtering by different values of a confidence threshold γ (Sec. 4.3). As we expected, we can see how accuracy increases with γ but the number of instances decreases. Interestingly, we observe that the number of instances decays later for estimations performed with Percival14 algorithm than for the other algorithms. This reflects the fact that Percival14 produces better BPM estimates. Filtering by the confidence measure, a potential

[12] Accuracy 1 is the percentage of instances whose estimated BPM is within 4% of the annotated ground truth. Accuracy 2 is the percentage of instances whose estimated BPM is within a 4% of $\frac{1}{3}$, $\frac{1}{2}$, 1, 2, or 3 times the ground truth BPM.

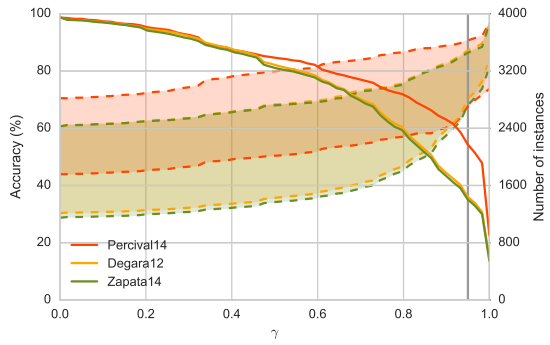


Figure 3. Accuracy vs confidence measure for FSL4 dataset. Lower bounds of the filled areas correspond to Accuracy 1e, while upper bounds correspond to Accuracy 2. Solid lines represent the number of instances remaining in the dataset.

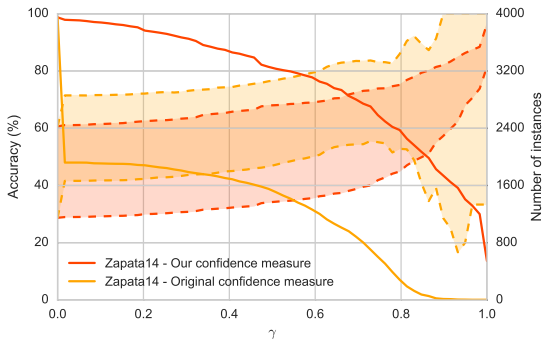


Figure 4. Comparison of our proposed confidence measure with the confidence measure proposed in [22] for FSL4 dataset and Zapata14 tempo estimation algorithm.

user searching for loops in a dataset could define a minimum threshold to get more accurate results at the expense of getting less loops returned. For instance, if we set a hard confidence threshold of $\gamma = 0.95$ (vertical line in Fig. 3), we find that the accuracies for Percival14 method range, on average, from 67% (Accuracy 1e) to 92% (Accuracy 2) while preserving an average of 52% of the instances. Interestingly enough, we observe that when setting that hard threshold, reported RekBox accuracies outperform these of Percival14 in all datasets, with an average increase ranging from 2% for Accuracy 2 to 14% for Accuracy 1e (all statistically significant with $p \ll 0.01$). We attribute this to the fact that RekBox seems to have a built-in confidence measure thresholding step in which the algorithm outputs 0 BPM when the analysis does not meet certain confidence requirements. Therefore, once filtering the datasets by γ (even with small values), all those instances whose BPM estimation is 0 BPM get discarded. Nevertheless, it is also important to note that filtering with the hard threshold, RekBox only preserves an average of 31% of the instances (lower than the 52% reported above by Percival14).

If we look at the combined accuracy and confidence measure $\bar{S}$ described in Sec. 4.3, we again find that Percival14 obtains the best score in all datasets and for all accuracy measures (i.e. for $A(\gamma)$ computed with Accuracy 1e, Accuracy 1 or Accuracy 2). This means that Percival14 offers the overall best balance between estimation accuracy and number of preserved instances in the dataset when filtering by a minimum confidence threshold.

5.3 Comparison of confidence measures

Zapata et. al. [22] propose a confidence measure that can be used for tempo estimation and that is based on computing the mutual agreement between an ensemble of tempo estimation algorithms. To make this confidence measure numerically comparable to the one we propose, we normalise the confidence output of Zapata et. al. to take values from 0 to 1. Similarly to Fig. 3, we plot the estimation accuracy and the number of remaining instances as a function of a minimum confidence threshold γ (Fig. 4). We observe that Zapata's confidence measure allows to achieve accuracies which are around 15% higher than when using our confidence. However, the number of remaining instances in the dataset is drastically reduced, and accuracy values for $\gamma > 0.75$ become inconsistent. Looking at the $\bar{S}$ score, we find that Zapata14 in combination with our confidence measure gets better results than when using the original measure, with an average $\bar{S}$ increase of 17%, 29% and 31% (for the three accuracy measures respectively). This indicates that our confidence measure is able to better maximise accuracy and number of remaining instances.

6. CONCLUSION

In this paper paper we have compared several tempo estimation algorithms using four datasets of music loops. We also described a simple confidence measure for tempo estimation algorithms and proposed a methodology for evaluating the relation between estimation accuracy and confidence measure. This methodology can also be applied to other MIR tasks, and we believe it encourages future research to put more emphasis on confidence measures. We found that by setting a high enough minimum confidence threshold, we can obtain reasonably high tempo estimation accuracies while preserving half of the instances in a dataset. However, these results are still far from being optimal: if we only consider exact BPM estimations (Accuracy 1e), the maximum accuracies we obtain are still generally lower than 70%. We foresee two complementary ways of improving these results by *i)* adapting tempo estimation algorithms to the case of music loops (e.g. taking better advantage of tempo steadiness and expected signal duration), and *ii)* developing more advanced confidence measures that take into account other properties of loops such as the beat strength or the rate of onsets. Overall, the work we present here contributes to the improvement of the reusability of unstructured music loop repositories.

7. ACKNOWLEDGMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688382.

8. REFERENCES

[1] Sebastian Böck, Florian Krebs, and Gerhard Widmer. Accurate Tempo Estimation based on Recurrent Neural Networks and Resonating Comb Filters. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2015.

[2] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, Jose Zapata, and Xavier Serra. ESSENTIA: An audio analysis library for music information retrieval. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 493–498, 2013.

[3] Matthew EP Davies and Mark D Plumbley. Context-dependent beat tracking of musical audio. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1009–1020, 2007.

[4] Norberto Degara, Enrique Argones Rua, Antonio Pena, Soledad Torres-Guijarro, Matthew EP Davies, and Mark D Plumbley. Reliability-Informed Beat Tracking of Musical Signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):290–301, 2012.

[5] Simon Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30:39–58, 2001.

[6] Anders Elowsson, Anders Friberg, Guy Madison, and Johan Paulin. Modelling the Speed of Music using Features from Harmonic/Percussive Separated Audio. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 481–486, 2013.

[7] Frederic Font, Gerard Roma, and Xavier Serra. Freesound technical demo. In *Proc. of the ACM Int. Conf. on Multimedia (ACM MM)*, pages 411–412, 2013.

[8] Mikel Gainza and Eugene Coyle. Tempo detection using a hybrid multiband approach. *IEEE Transactions on Audio, Speech and Language Processing*, 19(1):57–68, 2011.

[9] Aggelos Gkiokas, Vassilis Katsouros, George Carayannis, and Themos Stafylakis. Music Tempo Estimation and Beat Tracking By Applying Source Separation and Metrical Relations. In *Proc. of the Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, volume 7, pages 421–424, 2012.

[10] Fabien Gouyon, Anssi Klapuri, Simon Dixon, Miguel Alonso, George Tzanetakis, Christian Uhle, and Pedro Cano. An Experimental Comparison of Audio Tempo Induction Algorithms. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5):1832–1844, 2006.

[11] Peter Grosche and Meinard Müller. Extracting Predominant Local Pulse Information from Music Recordings. *IEEE Transactions on Audio, Speech and Language Processing*, 19(6):1688–1701, 2011.

[12] Anssi Klapuri, Antti Eronen, and Jaakko Astola. Analysis of the meter of musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–355, 2006.

[13] Meinard Müller. *Fundamentals of Music Processing*. Springer, 2015.

[14] João L Oliveira, Fabien Gouyon, Luis Gustavo Martins, and Luis Paolo Reis. IBT: A real-time tempo and beat tracking system. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 291–296, 2010.

[15] Graham Percival and George Tzanetakis. Streamlined tempo estimation based on autocorrelation and cross-correlation with pulses. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(12):1765–1776, 2014.

[16] Elio Quinton, Mark Sandler, and Simon Dixon. Estimation of the Reliability of Multiple Rhythm Features Extraction from a Single Descriptor. In *Proc. of the Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 256–260, 2016.

[17] Gerard Roma. *Algorithms and representations for supporting online music creation with large-scale audio databases*. PhD thesis, Universitat Pompeu Fabra, 2015.

[18] George Tzanetakis, Georg Essl, and Perry Cook. Human Perception and Computer Extraction of Musical Beat Strength. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx)*, pages 257–261, 2002.

[19] Fu-Hai Frank Wu and Jyh-Shing Roger Jang. A Supervised Learning Method for Tempo Estimation of Musical Audio. In *Proc. of the Mediterranean Conf. on Control and Automation (MED)*, pages 599–604, 2014.

[20] Jose R Zapata, Matthew EP Davies, and Emilia Gómez. Multi-Feature Beat Tracking. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(4):816–825, 2014.

[21] Jose R Zapata and Emilia Gómez. Comparative Evaluation and Combination of Audio Tempo Estimation Approaches. In *Proc. of the AES Int. Conf. on Semantic Audio*, pages 198 – 207, 2011.

[22] Jose R Zapata, André Holzapfel, Matthew EP Davies, João L Oliveira, and Fabien Gouyon. Assigning a Confidence Threshold on Automatic Beat Annotation in Large Datasets. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 157–162, 2012.

BRAIN BEATS: TEMPO EXTRACTION FROM EEG DATA

Sebastian Stober[1] **Thomas Prätzlich**[2] **Meinard Müller**[2]
[1] Research Focus Cognitive Sciences, University of Potsdam, Germany
[2] International Audio Laboratories Erlangen, Germany
sstober@uni-potsdam.de, {thomas.praetzlich, meinard.mueller}@audiolabs-erlangen.de

ABSTRACT

This paper addresses the question how music information retrieval techniques originally developed to process audio recordings can be adapted for the analysis of corresponding brain activity data. In particular, we conducted a case study applying beat tracking techniques to extract the tempo from electroencephalography (EEG) recordings obtained from people listening to music stimuli. We point out similarities and differences in processing audio and EEG data and show to which extent the tempo can be successfully extracted from EEG signals. Furthermore, we demonstrate how the tempo extraction from EEG signals can be stabilized by applying different fusion approaches on the mid-level tempogram features.

1 Introduction

Recent findings in cognitive neuroscience suggest that it is possible to track a listener's attention to different speakers or music signals [1,24], or to identify beat-related or rhythmic features in electroencephalography (EEG) recordings [1] of brain activity during music perception. In particular, it has been shown that oscillatory neural activity is sensitive to accented tones in a rhythmic sequence [19]. Neural oscillations entrain (synchronize) to rhythmic sequences [2, 14] and increase in anticipation of strong tones in a non-isochronous (not evenly spaced), rhythmic sequence [3, 4, 10]. When subjects hear rhythmic sequences, the magnitude of the oscillations changes for frequencies related to the metrical structure of the rhythm [16, 17].

EEG studies [5] have further shown that perturbations of the rhythmic pattern lead to distinguishable electrophysiological responses—commonly referred to as event-related potentials (ERPs). This effect appears to be independent of the listener's level of musical proficiency. Furthermore, [26] showed that accented (louder) beats imagined by a listener on top of a steady metronome beat can be recognized from ERPs. EEG signals have also been used to distinguish perceived rhythmic stimuli [21] with convolutional neural networks. First preliminary results using autocorrelation for tempo estimation from the EEG signal during perception and imagination of music have been reported in [20].

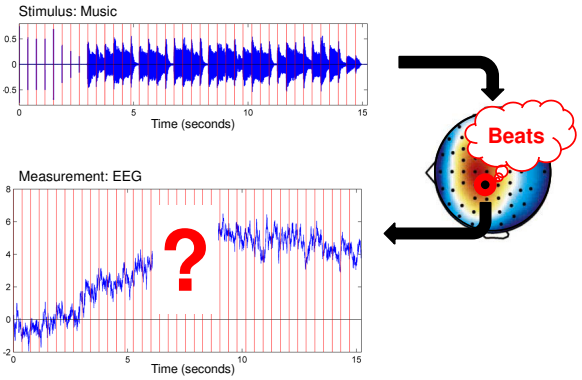


Figure 1. Question: Can we extract the tempo of a music recording from brain activity data (EEG) recorded during listening? The red vertical lines in the audio waveform (top) and the EEG signal (bottom) mark the beat positions.

This raises the question whether MIR techniques originally developed to detect beats and extract the tempo from music recordings could also be used for the analysis of corresponding EEG signals. One could argue that as the brain processes the perceived music, it generates a transformed representation which is captured by the EEG electrodes. Hence, the recorded EEG signal could in principle be seen as a mid-level representation of the original music piece that has been heavily distorted by two consecutive black-box filters—the brain and the EEG equipment.

This transformation involves and intermingles with several other brain processes unrelated to music perception and is limited by the capabilities of the recording equipment that can only measure cortical brain activity (close to the scalp). It further introduces artifacts caused by electrical noise or the participant's movements such as eye blinks. Figuratively speaking, this could be compared to a cocktail-party situation where the listener is not in the same room as the speakers but in the next room separated by a thick wall.

In this paper, we address the question whether well-established tempo and beat tracking methods, originally developed for MIR, can be used to recover tempo information from EEG data recorded from people listening to music, see Figure 1. In the remainder of this paper, we

[1] Electroencephalography (EEG) is a non-invasive brain imaging technique that relies on electrodes placed on the scalp to measure the electrical activity of the brain. A recent review of neuroimaging methods for music information retrieval (MIR) that also includes a comparison of EEG with different approaches is given in [11].

© Sebastian Stober, Thomas Prätzlich, Meinard Müller. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Sebastian Stober, Thomas Prätzlich, Meinard Müller. "Brain Beats: Tempo Extraction from EEG Data", 17th International Society for Music Information Retrieval Conference, 2016.

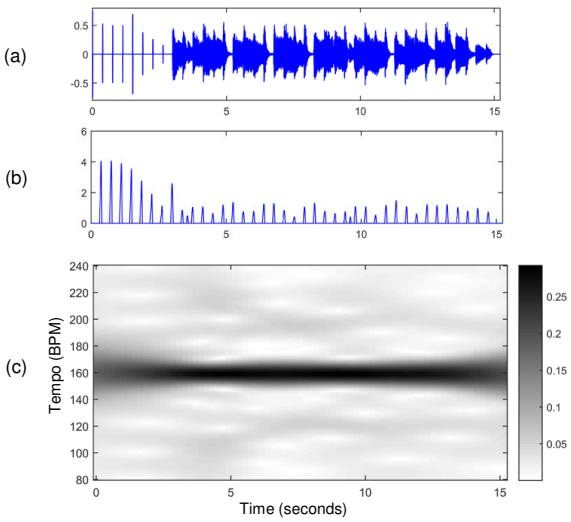


Figure 2. Tempogram computation for music signals. **(a)** Waveform signal. **(b)** Novelty curve. **(c)** Tempogram representation.

first briefly describe the EEG dataset (Section 2). As a first contribution, we explain how an MIR technique for tempo extraction can be applied on EEG signals (Section 3). Then, in Section 4, we evaluate the tempo extraction on the EEG signals by comparing it to the tempo extracted from the corresponding audio signals. As another contribution, we show that the tempo extraction on EEG signals can be stabilized by applying different fusion approaches. Finally, we conclude the paper with a summary and indication of possible research directions (Section 5).

2 Recording Setup and Dataset

In this study, we use a subset of the OpenMIIR dataset [22]—a public domain dataset of EEG recordings taken during music perception and imagination. [2] For our study, we use only the music perception EEG data from the five participants $p \in P := \{09, 11, 12, 13, 14\}$ [3] who listened to twelve short music stimuli—each 7s to 16s long. These stimuli were selected from well-known pieces of different genres. They span several musical dimensions such as meter, tempo, instrumentation (ranging from piano to orchestra) and the presence of lyrics (singing or no singing present), see Table 1. All stimuli were normalized in volume and kept similar in length, while ensuring that they all contained complete musical phrases starting from the beginning of the piece. The EEG recording sessions consisted of five trials $t \in T := \{1, \ldots, 5\}$ in which all stimuli $s \in S := \{01,02,03,04,11,12,13,14,21,22,23,24\}$ were presented in randomized order. This results in a total of $|S| \cdot |T| \cdot |P| = 12 \cdot 5 \cdot 5 = 300$ trials for the five participants, $|S| \cdot |T| = 12 \cdot 5 = 60$ trials per particpant, and $|P| \cdot |T| = 25$ trials per stimulus.

[2] The dataset is available at https://github.com/sstober/openmiir

[3] The remaining participants in the dataset had some of the stimuli presented at a slightly different tempo (c.f. [22]), which would not allow our fusion approaches discussed later in Section 4.

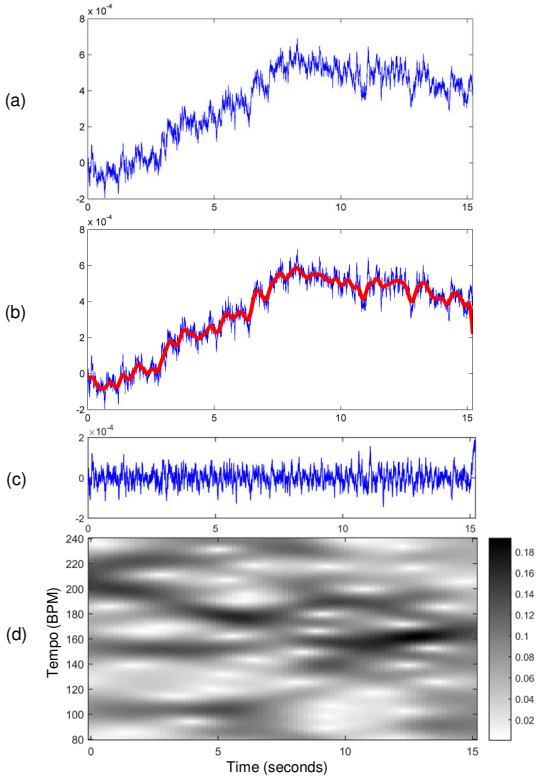


Figure 3. Tempogram computation for EEG signals. **(a)** EEG signal. **(b)** Local average curve. **(c)** Normalized EEG signal (used as novelty curve). **(d)** Tempogram representation.

EEG was recorded with a BioSemi Active-Two system using 64+2 EEG channels at 512 Hz. Horizontal and vertical electrooculography (EOG) channels were used to record eye movements. As described in [22], EEG pre-processing comprised the removal and interpolation of bad channels as well as the reduction of eye blink artifacts by removing highly correlated components computed using extended Infomax independent component analysis (ICA) [12] with the MNE-python toolbox [6].

3 Computation of Tempo Information

In this section, we describe how tempo information can be extracted both from music and EEG signals. To this end, we transform a signal into a *tempogram* $\mathcal{T} : \mathbb{R} \times \mathbb{R}_{>0} \to \mathbb{R}_{\geq 0}$ which is a time-tempo representation of a signal. A tempogram reveals periodicities in a given signal, similar to a spectrogram. The value $\mathcal{T}(t,\tau)$ indicates how predominant a tempo value $\tau \in \mathbb{R}_{>0}$ (measured in BPM) is at time position $t \in \mathbb{R}$ (measured in seconds) [15, Chapter 14].

In the following, we provide a basic description of the tempogram extraction for music recordings (Section 3.1) and EEG signals (Section 3.2). For algorithmic details,

we refer to the descriptions in [8, 15]. To compute the tempograms for the experiments in this work, we used the implementations from the Tempogram Toolbox. [4] Furthermore, we describe how the tempo information of a tempogram can be aggregated into a tempo histogram similar to [25] from which a global tempo value can be extracted (Section 3.3).

3.1 Tempogram for Music Audio Signals

To compute a tempogram, a given music audio signal is first transformed into a novelty curve capturing note onset information. In the following, we use a novelty curve computed as the positive part of a spectral flux, see [8]. Figure 2a shows the waveform of an audio stimulus, which begins with a set of cue clicks (in beats) followed by a short music excerpt of the same tempo. In Figure 2b, the novelty curve extracted from the waveform is shown. The onsets of the cue clicks are clearly reflected by peaks in the novelty curve. For the subsequent music excerpt, one can see that the peaks are similarly spaced as the cue clicks. However, there are some additional peaks in the music excerpt that correspond to additional notes or noise. Especially for music with soft onsets, the novelty curve may contain some noise in the peak structures. As for the tempo extraction, we further transform the novelty curve into an audio tempogram that reveals how dominant different tempi are at a given time point in the audio signal. In this study, we use a tempogram computed by short-term Fourier analysis of the novelty curve with a tempo window of 8 seconds, see [8] for details. The frequency axis (given in Hz) is transformed into a tempo axis (given in BPM). In Figure 2c, the audio tempogram of the example is shown, which reveals a predominant tempo of 160 BPM throughout the recording.

3.2 Tempogram for EEG Signals

In this section, we describe how we extract a tempogram from EEG signals that were measured when participants listened to a music stimulus. In principle, we use a similar approach for the tempo extraction from EEG signals as for the music recordings.

First, we aggregate the 64 EEG channels into one signal. Note that there is a lot of redundancy in these channels. This redundancy can be exploited to improve the signal-to-noise ratio. In the following, we use the channel aggregation filter shown in Figure 4. It was learned as part of a convolutional neural network (CNN) during a previous experiment attempting to recognize the stimuli from the EEG recordings [23]. In [23], a technique called "similarity-constraint encoding" (SCE) was applied that is motivated by earlier work on learning similarity measures from relative similarity constraints as introduced in [18]. The CNN

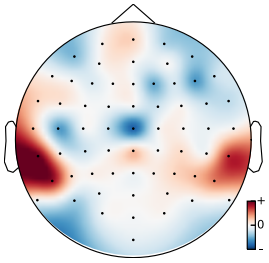


Figure 4. Topographic visualization of the SCE-trained channel aggregation filter used to compute a single signal from the 64 EEG channels (indicated by black dots). The filter consists of a weighted sum with the respective channel weights (shown in a color-coded fashion) and a subsequent application of the $\tanh$ which results in an output range of $[-1,1]$.

was trained using triplets of trials consisting of a reference trial, a paired trial from the same class (i.e., the same stimulus) and a third trial from a different class. For each triplet, the network had to predict which trial belongs to the same class as the reference trial. This way, it learned channel aggregation weights that produce signals that are most similar for trials belonging to the same class. In our earlier experiments, we found that the resulting aggregated EEG signals capture important characteristics of the music stimuli such as downbeats. We hypothesized that the learned filter from [23] could also be useful in our tempo extraction scenario, even though it is a very different task. [5]

Figure 3a shows an example of an aggregated EEG signal. From the aggregated EEG signal, we then compute a novelty curve. Here, opposed to the novelty computation for the audio signal, we assume that the beat periodicities we want to measure are already present in the time-domain EEG signal. We therefore interpret the EEG signal as a kind of novelty curve. As pre-processing, we normalize the signal by subtracting a moving average curve, see Figure 3b. This ensures that the signal is centered around zero and low frequent components of the signal are attenuated. The resulting signal (Figure 3c) is then used as a novelty curve to compute an EEG tempogram that reveals how dominant different tempi are at a given time point in the EEG signal (see Figure 3d). Note that, compared to the audio novelty curve, the EEG novelty curve is much nosier. As a result, there is more noise in the EEG tempogram compared to the audio tempogram, making it hard to determine a predominant global tempo.

3.3 Tempo Histograms

In this section, we explain how we extract a single tempo value from the audio and EEG tempograms. First, we aggregate the time-tempo information over the time by

[4] The Tempogram Toolbox contains MATLAB implementations for extracting various types of tempo and pulse related audio representations [9] A free implementation can be obtained at https://www.audiolabs-erlangen.de/resources/MIR/tempogramtoolbox

[5] We compared the tempo extraction on the SCE-trained channel aggregation with simply averaging the raw data across channels and found that the tempo extraction on the raw EEG data often performed roughly 10% points worse and was only on par with SCE in the best cases.

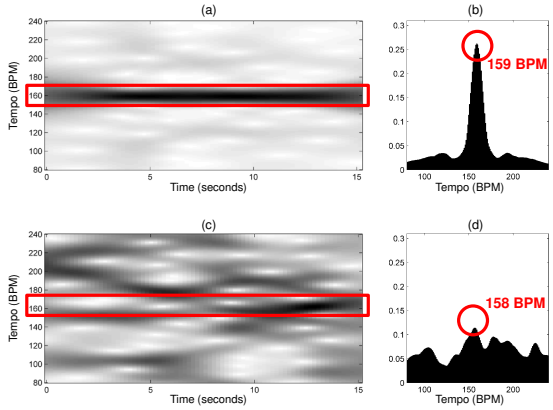


Figure 5. **(a)** Tempogram for the music signal from Figure 2 and **(b)** resulting tempo histogram. **(c)** Tempogram for EEG signal from Figure 3 and **(d)** resulting tempo histogram.

computing a tempo histogram $\mathcal{H} : \mathbb{R}_{>0} \to \mathbb{R}_{\geq 0}$ from the tempogram (similar to [25]). A value $\mathcal{H}(\tau)$ in the tempo histogram indicates how present a certain tempo τ is within the entire signal. In Figure 5, a tempogram for a music recording and an EEG signal are shown along with their respective tempo histograms. In the audio tempo histogram, the highest peak at $\tau = 159$ BPM indicates the correct tempo of the music recording. The tempogram for the EEG data is much noisier, where it is hard to identify a predominant tempo from the tempogram. In the tempo histogram, however, the highest peak in the example corresponds to a tempo of 158 BPM, which is nearly the same as the main tempo obtained from the audio tempo histogram.

4 Evaluation

In this section, we report on our experiments to show to which extent the tempo extraction for the audio signals and the EEG signals are related. In the following, $\mathcal{H}_{s,p,t}$ denotes to the tempo histogram stemming from the audio stimulus $s \in S$, participant $p \in P$, and trial $t \in T$ (see Section 2). An overview of the stimuli is given in Table 1. For all experiments, we used a tempo window of 8 seconds, see [7]. Furthermore, we applied a moving average filter on the EEG data of 0.5 seconds. In Section 4.1, we introduce our evaluation measures and discuss quantitative results for different tempo extraction strategies. Then, in Section 4.2, to better understand the benefits and limitations of our approach, we look at some representative examples for tempograms and tempo histograms across the dataset.

4.1 Quantitative Results

To determine the tempo a of a given audio stimulus, we consider the highest peak in the respective audio tempo histogram $\mathcal{H}^{\text{audio}}$, see Table 1.[6] The EEG tempo histogram $\mathcal{H}^{\text{EEG}}$ is much noisier. To obtain some insights on the tempo information contained in $\mathcal{H}^{\text{EEG}}$, we look at the tempi corresponding to the highest peak as well as subsequent peaks. To this end, after selecting the tempo corresponding to the highest peak, we set the values within ± 10 BPM in the neighborhood of the peak in the tempo histogram to zero. This procedure is repeated until the top n peaks are selected. In the following, we consider the first three tempi b_1, b_2, b_3 obtained from a given tempo histogram and build the sets of tempo estimates $\mathcal{B}_1 := \{b_1\}$ (top 1 peak), $\mathcal{B}_2 := \{b_1, b_2\}$ (top 2 peaks), and $\mathcal{B}_3 := \{b_1, b_2, b_3\}$ (top 3 peaks). To determine the error of the tempo estimates $\mathcal{B}_n$ with $n \in \{1,2,3\}$, we compute the *minimum absolute BPM deviation* compared to the audio tempo: $\varepsilon(\mathcal{B}_n, a) := \min_{b \in \mathcal{B}_n} |b - a|$. Furthermore, as small errors are less severe as large errors, we quantify different error classes with an error tolerance $\delta \geq 0$. To this end, we compute the *BPM error rate* $E_\delta(\mathcal{B}_n)$ which is defined as the percentage of absolute BPM deviations with $\varepsilon(\mathcal{B}_n, a) > \delta$. In our experiments, we use different $\delta \in \{0,3,5,7\}$ (given in BPM).

Table 1. Information about the tempo, meter and length of the stimuli (with cue clicks) used in this study. Note that stimuli 1–4 and 11–14 are different versions of the same song with and without lyrics.

ID	Name	Meter	Length with cue	Tempo [BPM]
1	Chim Chim Cheree (lyrics)	3/4	14.9s	213
2	Take Me Out to the Ballgame (lyrics)	3/4	9.5s	188
3	Jingle Bells (lyrics)	4/4	12.0s	199
4	Mary Had a Little Lamb (lyrics)	4/4	14.6s	159
11	Chim Chim Cheree	3/4	15.1s	213
12	Take Me Out to the Ballgame	3/4	9.6s	188
13	Jingle Bells	4/4	11.3s	201
14	Mary Had a Little Lamb	4/4	15.2s	159
21	Emperor Waltz	3/4	10.3s	174
22	Hedwig's Theme (Harry Potter)	3/4	18.2s	165
23	Imperial March (Star Wars Theme)	4/4	11.5s	104
24	Eine Kleine Nachtmusik	4/4	10.2s	140
	mean		12.7s	175

We performed the tempo extraction from the EEG tempo histograms with three different strategies:

(S1) *Single-trial tempo extraction*: For each trial, the tempo is extracted individually. This results in extracting the tempi from $|S| \cdot |P| \cdot |T| = 12 \cdot 5 \cdot 5 = 300$ tempo histograms (see Section 4).

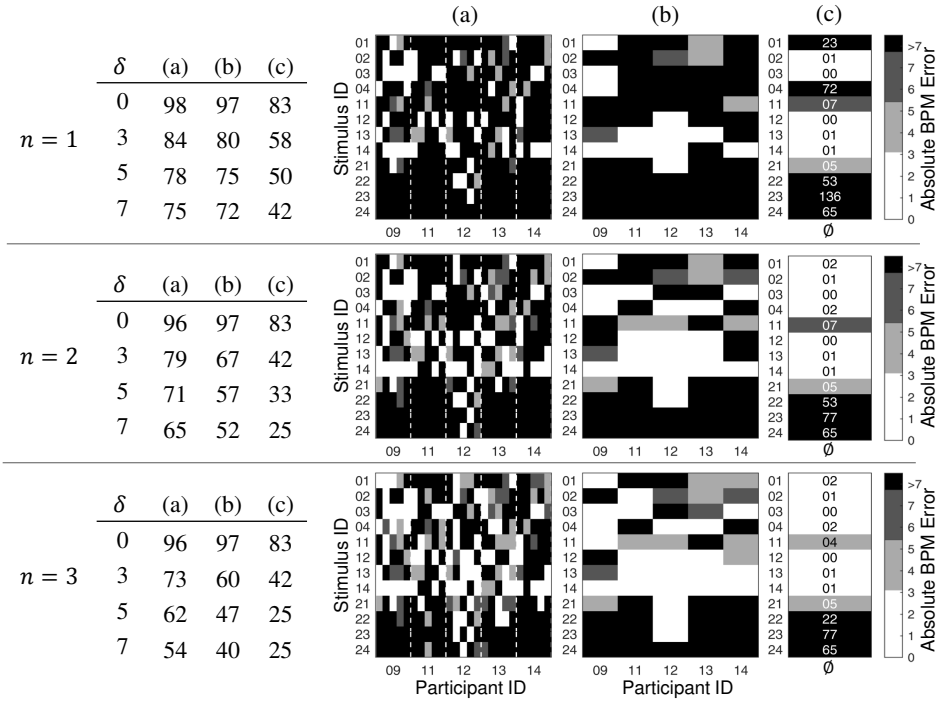
(S2) *Fusion I*: Fixing a stimulus $s \in S$ and a participant $p \in P$, we average over the tempo histograms of the trials $t \in T$:

$$\mathcal{H}_{s,p}(\tau) := \frac{1}{|T|} \sum_{t \in T} \mathcal{H}_{s,p,t}(\tau).$$

This results in extracting the tempi from $|S| \cdot |P| = 60$ tempo histograms.

(S3) *Fusion II*: Fixing a stimulus $s \in S$, we average the tempo histograms over the participants $p \in P$ and the

[6] The OpenMIIR dataset also provides ground-truth tempi in the metadata. Except for stimulus 21 with a difference of 4 BPM, our computed tempi differed at most 1 BPM from the OpenMIIR ground-truth.



$n = 1$	δ	(a)	(b)	(c)
	0	98	97	83
	3	84	80	58
	5	78	75	50
	7	75	72	42

$n = 2$	δ	(a)	(b)	(c)
	0	96	97	83
	3	79	67	42
	5	71	57	33
	7	65	52	25

$n = 3$	δ	(a)	(b)	(c)
	0	96	97	83
	3	73	60	42
	5	62	47	25
	7	54	40	25

Figure 6. Tables with BPM error rates in percent (left) and absolute BPM error (right) for the set of tempo estimates $\mathcal{B}_n$. **(a)** strategy S1. Note that for each participant, there are five columns in the matrix that correspond to the different trials. **(b)** strategy S2. **(c)** strategy S3.

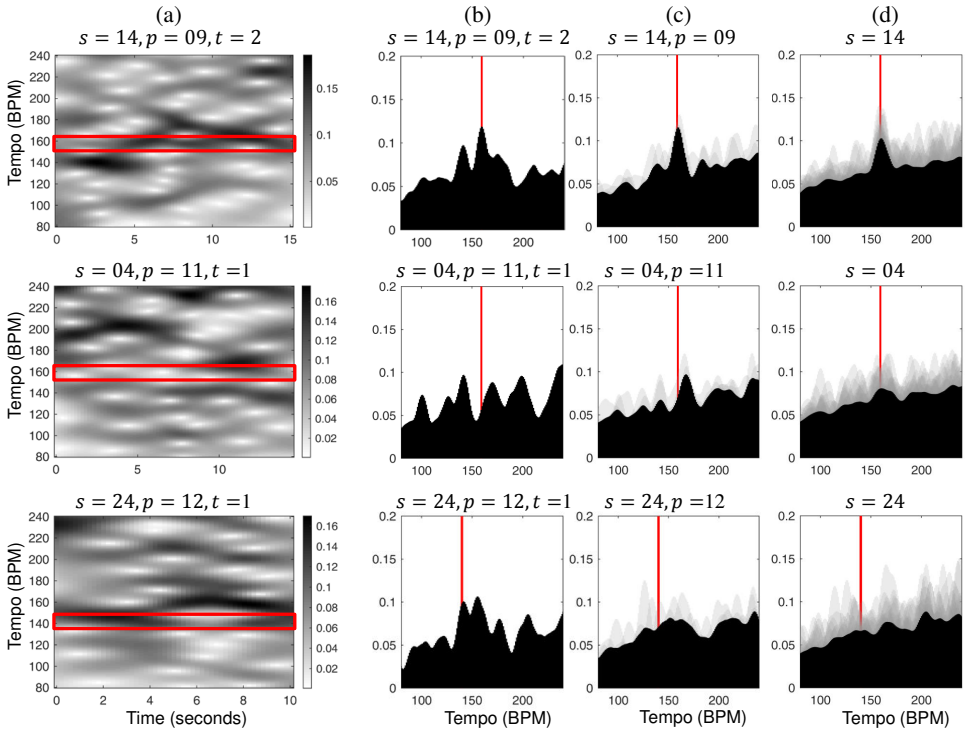


Figure 7. Tempograms and tempo histograms for stimuli 14, 04, and 24 (top to bottom). The red boxes and lines mark the audio tempo. The gray histograms in the background were averaged in the fusion strategies. **(a)** Tempogram for S1. **(b)** Tempo histogram for S1, derived from (a). **(c)** Tempo histogram for S2. $\mathcal{H}_{s,p}(\tau)$ was computed from five tempo histograms (5 trials). **(d)** Tempo histogram for S3. $\mathcal{H}_s(\tau)$ was computed from the 25 tempo histograms (5 participants with 5 trials each).

trials $t \in T$:

$$\mathcal{H}_s(\tau) := \frac{1}{|P| \cdot |T|} \sum_{p \in P} \sum_{t \in T} \mathcal{H}_{s,p,t}(\tau).$$

This results in extracting the tempi from $|S| = 12$ tempo histograms.

Note that it is a common approach in EEG signal processing to average the EEG signals over different trials as described in [13]. This usually reduces the noise in the signals. In this study, instead of averaging over the EEG signals, we averaged over the tempo histograms, which is a kind of mid-level feature representation.

Figure 6 shows the BPM error rates (left) as well as the absolute BPM error (right). Each row in the figure corresponds to the results for a different set of tempo estimates $\mathcal{B}_n$. For $n = 1$, a strict error tolerance of $\delta = 0$, and strategy S1, the tempo extraction basically fails, having a BPM error rate of 98%. This is not surprising, as no deviation from the audio tempo is allowed. When allowing a deviation of five BPM ($\delta = 5$), the tempo extraction using only the top peak ($n = 1$) fails in 78% of the cases. By applying the fusion strategy S2 for the tempo extraction, the BPM error rate significantly drops to 75%, which is an improvement of 3% points. The BPM error rate goes down to 50% for the fusion strategy S3 which averages over all trials for a given stimulus. This shows that averaging stabilizes the results. When looking at the results by considering the set of tempo estimates $\mathcal{B}_2$ ($n = 2$) and $\mathcal{B}_3$ ($n = 3$), we can see that the second and third peak often correspond to the expected tempo. For example, for $\delta = 5$ and strategy S3, the BPM error rate goes down from 50% (for $n = 1$), to 33% (for $n = 2$), and 25% (for $n = 3$).

Furthermore, Figure 6 shows that the results strongly depend on the music stimulus used. The extraction for stimulus $s = 14$, for example, works well for nearly all participants. This is a piece performed on a piano which has clear percussive onsets. Also, for the first eight stimuli ($01 - 04$ and $11 - 14$) the tempo extraction seems to work better than for the last four stimuli ($21 - 24$). This may have different reasons. For instance, $s = 21$, $s = 23$ and $s = 24$ are amongst the shortest stimuli in the dataset and $s = 22$ has very soft onsets. Furthermore, the stimuli 21–24 are purely instrumental (soundtracks and classical music) without lyrics.

4.2 Qualitative examples

Figure 7 shows the tempograms and tempo histograms for some representative examples. We subsequently discuss the top, middle, and bottom row of the figure corresponding to stimulus 14, 04, and 24, respectively.

The EEG tempogram shown in Figure 7a (top row) clearly reflects the correct tempo of the music stimulus. In the corresponding tempo histogram (b), a clear peak can be seen at the correct tempo. In the tempo histograms (c) and (d), corresponding to strategies S2 and S3, one can clearly see the stabilizing and noise reducing effect of the two fusion strategies, resulting in a very clear tempo peak.

In Figure 7b (middle row), the tempo histogram does not reveal the expected tempo. As also indicated by the tempogram in Figure 7a, the listener does not seem to follow the beat of the music stimulus. However, when averaging over the trials of participant $p = 11$, the tempo peak near 160 BPM becomes more dominant (see tempo histogram (c)). When averaging over all trials and all participants for the stimulus $s = 04$, the tempo peak becomes more blurry, but appears at the expected position, (see tempo histogram (d)).

For the third example in Figure 7 (bottom row), the tempogram (a) shows predominant values near the correct tempo. In the corresponding tempo histogram (b), the correct tempo is revealed by the second peak. However, the histograms for strategy S2 (c) and S3 (d) lead to very blurry peaks where the correct tempo peak is not among the top three peaks. These examples illustrate that the fusion strategies often stabilize the tempo extraction. When the data is too noisy, however, these strategies may sometimes degrade the results.

5 Conclusions

In this paper, we presented a case study where we applied an MIR tempo extraction technique, originally developed for audio recordings, to EEG signals. In experiments, we showed that it is possible to extract the tempo from EEG signals using a similar technique as for audio signals. We could see that the averaging over trials and participants typically stabilized the tempo estimation. Furthermore, we noticed that the quality of the tempo estimation was highly dependent on the music stimulus used. Exploring this effect is beyond the scope of this small study. To properly understand the reasons for this effect, a large-scale music perception experiment using stimuli with systematically adapted tempi would be needed. Possible reasons might be the complexity of the music stimuli, the presence of lyrics, the participants, or the applied methodology and techniques. Investigating these issues could be a starting point for interdisciplinary research between MIR and music perception. Supplementary material and code is available at https://dx.doi.org/10.6084/m9.figshare.3398545.

Acknowledgments Sebastian Stober would like to acknowledge the support by the German Academic Exchange Service (DAAD). Thomas Prätzlich and Meinard Müller are supported by the German Research Foundation (DFG MU 2686/6-1, DFG MU 2686/7-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institute for Integrated Circuits IIS. Furthermore, we would like to thank Colin Raffel and the other organizers of the *HAMR* Hack Day at ISMIR 2015, where the core ideas of the presented work were born.

6 References

[1] A. Aroudi, B. Mirkovic, M. De Vos, and S. Doclo. Auditory attention decoding with EEG recordings using noisy acoustic reference signals. In *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 694–698, 2016.

[2] L.K. Cirelli, D. Bosnyak, F.C. Manning, C. Spinelli, C. Marie, T. Fujioka, A. Ghahremani, and L.J. Trainor. Beat-induced fluctuations in auditory cortical beta-band activity: Using EEG to measure age-related changes. *Frontiers in Psychology*, 5(Jul):1–9, 2014.

[3] T. Fujioka, L.J. Trainor, E.W. Large, and B. Ross. Beta and gamma rhythms in human auditory cortex during musical beat processing. *Annals of the New York Academy of Sciences*, 1169:89–92, 2009.

[4] T. Fujioka, L.J. Trainor, E.W. Large, and B. Ross. Internalized Timing of Isochronous Sounds Is Represented in Neuromagnetic Beta Oscillations. *Journal of Neuroscience*, 32(5):1791–1802, 2012.

[5] E. Geiser, E. Ziegler, L. Jancke, and M. Meyer. Early electrophysiological correlates of meter and rhythm processing in music perception. *Cortex*, 45(1):93–102, January 2009.

[6] A. Gramfort, M. Luessi, E. Larson, D.A. Engemann, D. Strohmeier, C. Brodbeck, R. Goj, M. Jas, T. Brooks, L. Parkkonen, and M. Hämäläinen. MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience*, 7, December 2013.

[7] P. Grosche and M. Müller. A mid-level representation for capturing dominant tempo and pulse information in music recordings. In *Proc. of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 189–194, 2009.

[8] P. Grosche and M. Müller. Extracting predominant local pulse information from music recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6):1688–1701, 2011.

[9] P. Grosche and M. Müller. Tempogram toolbox: Matlab implementations for tempo and pulse analysis of music recordings. In *Late-Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, 2011.

[10] J.R. Iversen, B.H. Repp, and A.D. Patel. Top-down control of rhythm perception modulates early auditory responses. *Annals of the New York Academy of Sciences*, 1169:58–73, 2009.

[11] B. Kaneshiro and J.P. Dmochowski. Neuroimaging methods for music information retrieval: Current findings and future prospects. In *Proc. of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 538–544, 2015.

[12] T.-W. Lee, M. Girolami, and T.J. Sejnowski. Independent Component Analysis Using an Extended Infomax Algorithm for Mixed Subgaussian and Supergaussian Sources. *Neural Computation*, 11(2):417–441, 1999.

[13] S.J. Luck. *An introduction to the event-related potential technique*. MIT press, 2014.

[14] H. Merchant, J.A. Grahn, L.J. Trainor, M. Rohrmeier, and W.T. Fitch. Finding the beat: a neural perspective across humans and non-human primates. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 370(1664), 2015.

[15] M. Müller. *Fundamentals of Music Processing*. Springer Verlag, 2015.

[16] S. Nozaradan, I. Peretz, M. Missal, and A. Mouraux. Tagging the neuronal entrainment to beat and meter. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 31(28):10234–10240, 2011.

[17] S. Nozaradan, I. Peretz, and A. Mouraux. Selective Neuronal Entrainment to the Beat and Meter Embedded in a Musical Rhythm. *The Journal of Neuroscience*, 32(49):17572–17581, December 2012.

[18] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. *Advances in neural information processing systems (NIPS)*, pages 41–48, 2004.

[19] J.S. Snyder and E.W. Large. Gamma-band activity reflects the metric structure of rhythmic tone sequences. *Cognitive Brain Research*, 24:117–126, 2005.

[20] A. Sternin, S. Stober, J.A. Grahn, and A.M. Owen. Tempo estimation from the EEG signal during perception and imagination of music. In *International Workshop on Brain-Computer Music Interfacing / International Symposium on Computer Music Multidisciplinary Research (BCMI/CMMR)*, 2015.

[21] S. Stober, D.J. Cameron, and J.A. Grahn. Using convolutional neural networks to recognize rhythm stimuli from electroencephalography recordings. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1449–1457, 2014.

[22] S. Stober, A. Sternin, A.M. Owen, and J.A. Grahn. Towards music imagery information retrieval: Introducing the OpenMIIR dataset of EEG recordings from music perception an imagination. In *Proc. of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 763–769, 2015.

[23] S. Stober, A. Sternin, A.M. Owen, and J.A. Grahn. Deep feature learning for EEG recordings. *arXiv preprint arXiv:1511.04306*, 2015.

[24] M.S. Treder, H. Purwins, D. Miklody, I. Sturm, and B. Blankertz. Decoding auditory attention to instruments in polyphonic music using single-trial EEG classification. *Journal of Neural Engineering*, 11(2):026009, April 2014.

[25] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[26] R.J. Vlek, R.S. Schaefer, C.C.A.M. Gielen, J.D.R. Farquhar, and P. Desain. Shared mechanisms in perception and imagery of auditory accents. *Clinical Neurophysiology*, 122(8):1526–1532, August 2011.

Oral Session 3

Users

A PLAN FOR SUSTAINABLE MIR EVALUATION

Brian McFee
Center for Data Science / MARL
New York University
brian.mcfee@nyu.edu

Eric J. Humphrey
Spotify, Ltd.
ejhumphrey@spotify.com

Julián Urbano
Music Technology Group
Universitat Pompeu Fabra
urbano.julian@gmail.com

ABSTRACT

The Music Information Retrieval Evaluation eXchange (MIREX) is a valuable community service, having established standard datasets, metrics, baselines, methodologies, and infrastructure for comparing MIR methods. While MIREX has managed to successfully maintain operations for over a decade, its long-term sustainability is at risk. The imposed constraint that input data cannot be made freely available to participants necessitates that all algorithms run on centralized computational resources, which are administered by a limited number of people. This incurs an approximately linear cost with the number of submissions, exacting significant tolls on both human and financial resources, such that the current paradigm becomes *less* tenable as participation increases. To alleviate the recurring costs of future evaluation campaigns, we propose a distributed, community-centric paradigm for system evaluation, built upon the principles of openness, transparency, reproducibility, and incremental evaluation. We argue that this proposal has the potential to reduce operating costs to sustainable levels. Moreover, the proposed paradigm would improve scalability, and eventually result in the release of large, open datasets for improving both MIR techniques and evaluation methods.

1. INTRODUCTION

Evaluation plays a central role in the development of music information retrieval (MIR) systems. In addition to empirical results reported in individual articles, community evaluation campaigns like MIREX [6] provide a mechanism to standardize methodology, datasets, and metrics to benchmark research systems. MIREX has earned a special role within the MIR community as the central forum for system benchmarking. However, the annual operating costs incurred by MIREX are unsustainable by the MIR community. Much of these costs derive from one-time expenditures — *e.g.*, the time spent getting a participant's algorithm to run — which primarily benefit individual participants, but not the MIR community at large. If we, as a community, are to continue hosting regular evaluation campaigns, we will soon require a more efficient and sustainable model.

The evaluation problem has lurked the MIR community for years. The MIReS Roadmap for Music Information ReSearch identified it as one of the main technical-scientific grand challenges in MIR research [20], and during the ISMIR 2012 conference a discussion panel was held to explicitly address this issue [17]. Previous research has discussed some limitations of MIREX-like evaluation and made general proposals to avoid them [21, 23], and other community-led platforms have been put forward to try to minimize them in practice, most notably MusiClef [14] and the MSD Challenge [11]. However, for different reasons, they have been unable to continue operating.

Reflecting upon the prior work, we propose in this article a sustainable, open framework for community-driven MIR evaluation campaigns. Our proposal is motivated by three complementary factors. First, we strive to reduce the cost of running and maintaining the evaluation framework. Second, we hope to improve transparency and openness wherever possible. Third, we plan to establish a sustainable framework that will produce open, public data sets consisting of both inputs and reference annotations. By directing the majority of resources toward the production of open data, the proposed framework will be of value to the greater MIR community in perpetuity, and benefits will not be limited to participants in a particular year's campaign.

We stress that this document describes not a fully implemented framework, but a specific *proposal* put forward by a group of authors dedicated to seeing it put into practice. Our goal in writing this document at this early stage is to solicit input from the MIR community before implementation details have been finalized. In collaboration with the community, we hope to develop a framework that benefits as many people as possible and requires minimal financial support for years to come.

2. MIREX

The Music Information Retrieval Evaluation eXchange is a framework for the community driven evaluation of MIR algorithms [6, 7]. The annual tradition of MIREX was established early in the lifetime of ISMIR, and drew significant inspiration from the well-established TREC framework [24]. Thanks in large part to the vision of MIR pioneers, the first official iteration took place at ISMIR 2005 after much preliminary work, including a trial run the year

© Brian McFee, Eric J. Humphrey, Julián Urbano. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Brian McFee, Eric J. Humphrey, Julián Urbano. "A Plan for Sustainable MIR Evaluation", 17th International Society for Music Information Retrieval Conference, 2016.

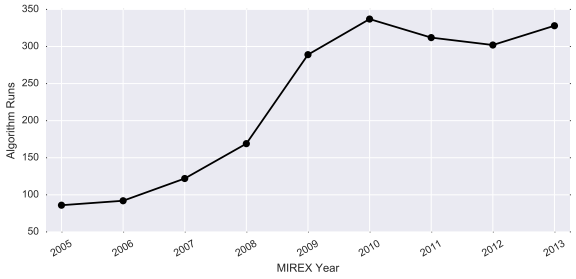


Figure 1. Number of algorithms run at MIREX over the course of almost a decade.

prior called the Audio Description Contest (ADC). The practicalities of MIREX are hosted by the IMIRSEL group at UIUC, and the organization has successfully earned multiple grants to jump-start the evaluation effort at ISMIR.

At a high level, MIREX operates in the following steps:

1. Identify some task of interest, *e.g.*, chord estimation.
2. Formulate the problem and evaluation metrics.
3. Construct (and annotate) a corpus of data.
4. Release a subset of the data for development purposes; retain the rest as private data for evaluation.
5. Invite participants to submit system implementations, which then are executed on private servers.
6. Evaluate predicted outputs against reference annotations or human judgments.
7. Repeat steps 5–6 annually. Intermittently revisit step 2 if needed, and steps 3 and 4 if possible.

Importantly, this approach differs from TREC-style evaluation by operating in an "algorithm-to-data" model, where facilitators oversee the application of code submissions to privately held data, rather than participants submitting predictions over a freely available dataset. The rationale for this decision is understandable. In contrast to other machine perception domains, such as natural language processing, speech recognition, or computer vision, intellectual property and copyright law imposes stiff penalties on the illegal distribution of recorded music. Due to a history of litigation from the recording industry, there is a pervasive sense of fear in the MIR community that sharing copyrighted audio data would invite crippling lawsuits [6].

However, experience with MIREX over the last decade has demonstrated that bringing algorithms to data entails fundamental limitations. First, as a matter of practicality, doing so incurs steep computational and financial costs that the community cannot hope to sustain indefinitely. Running hundreds of research systems demands significant computational resources, which must be either rented or purchased outright. More often than not, these systems are research prototypes which require substantial, manual intervention to operate correctly, and are seldom optimized for efficiency or ease-of-use. While task-dependent runtime limits are placed on algorithm execution (between 6 and 72 hours), MIREX requires months, if not years, of annual compute-time.

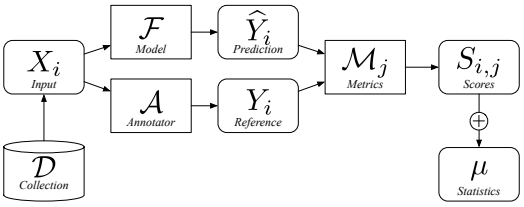


Figure 2. Diagram of the standard approach to the benchmarking of MIR systems.

The financial burden of computation can be negligible in comparison to the requisite human effort. As a point of reference, MIREX 2007 alone required "nearly 1000 person-hours" to supervise the execution of 122 algorithms from 40 teams [6]. As illustrated in Figure 1, the number of algorithm runs at MIREX has nearly tripled in the years since. [1] As a rough estimate, the last decade has likely consumed on the order of *10,000* person-hours just bringing algorithms to data. Not only is this rate unsustainable, but the combined operating costs only *increase* with participation. Said differently, the worst thing that could happen to MIREX in its current form is growth.

Operating costs aside, MIREX has indeed produced valuable insights into the development of MIR systems [6]. Unfortunately, many scientific endeavors are largely impeded or, at worst, wholly obfuscated in the current paradigm. To illustrate, consider the standard approach to benchmarking MIR systems as depicted in Figure 2. An input, X_i, is observed by an annotator, $\mathcal{A}$, who produces a reference output, Y_i. Similarly, a system $\mathcal{F}$ operates on the same input, and produces an estimate $\widehat{Y}_i$. Each of several performance metrics, $\mathcal{M}_j$, are applied to these two representations, yielding a number of performance scores, $S_{i,j}$ This process is repeated over a sample of n input-output pairs, $\{X_i, Y_i\} \in \mathcal{D}$, and the sample-wise scores are aggregated into summary statistics, μ, the reliability of which generally increases with $n = |\mathcal{D}|$.

In the current MIREX formulation, a lack of transparency renders participants scientifically blind in a number of ways [23]. First, there is no direct access to the reference annotations Y_i, and, in most cases, no access to the input X_i either. Furthermore, $\widehat{Y}_i$ is of little use without X_i for context. This makes it exceedingly difficult to learn from the results of the evaluation. Without access to the underlying data, how can one diagnose the cause of an erroneous estimate, or discover avenues for improvement? Similarly, there is no way to gauge the distribution of the data or estimate any bias in the sampling, beyond what may be inferred from public development sets.

The behavior of the human annotators is also obscured, as are the instructions provided when the annotation was initially performed [16]. Consequently, the problem formulation itself is effectively hidden, and subject to drift over time. For the sake of visibility into the evaluation metrics $\mathcal{M}_j$, the original NEMA infrastructure is open

[1] `https://www.hathitrust.org/documents/mirex_htrc_same_rda.pdf`

source, and an ongoing community-led effort continues to standardize and improve upon these implementations [18]. Still, without access to the data, it is exceedingly difficult to perform *meta-evaluation*, like comparing new metrics on old data, without seeking privileged access to the historical records.

Finally, it is only fair to admit that large scale evaluation is a considerable undertaking with plenty of room for error and misinterpretations. Conducting these campaigns in the open makes it easier to detect and diagnose any missteps.

Due to the issues highlighted above, resources that could have been devoted to constructing and enlarging open data sets have instead been absorbed by irrecoverable operating costs. This is not only detrimental to system development, but also to evaluation, because it is critical to routinely refresh the evaluation data to reduce bias and prevent statistical over-fitting. Without a fresh source of annotated data, early concerns about the community eventually over-fitting to benchmark datasets are beginning to manifest.

In some cases, this is because the data used for evaluation exists in the wild: one submission in 2011 achieved nearly perfect scores in chord estimation due to being pre-trained on the test data. [2] In other cases, participants may mis-interpret the task guidelines, as evidenced by submissions of offline systems for tasks that are online, or by others that mistakenly use annotations from previous folds in a train-test task. Across the board, hidden evaluation data is slowly being over-fit by trial and error, as teams implicitly leverage the weak error signal across campaigns to "improve" systems. These results can be misleading due to the fixed but unknown biases in the data distribution, which become apparent when datasets are expanded — like the introduction of the Billboard dataset to the chord estimation task in 2012 [2] — or as further insight is gained, as in the disclosure of the meta-data in the music similarity corpus.

To make matters worse, there is no feasible plan in place to replenish the evaluation datasets currently used by MIREX, nor any long-term plans to replace that data when it inevitably becomes stale. MIREX has primarily relied upon the generosity of the community to both curate and donate data for the purposes of evaluation. This approach also struggles with transparency, making it susceptible to issues of methodology and problem formulation, and can hardly be relied upon as a regular resource. Data collection is a challenging, unavoidable reality that demands a viable plan going forward.

After a decade of MIREX, we have learned which techniques work and which do not. Most importantly, we have learned the importance of establishing a collective endeavor to periodically and systematically evaluate MIR systems. Unfortunately, we have also learned of the burdens entailed by such an initiative, and the limitations of its current form.

[2] http://nema.lis.illinois.edu/nema_out/mirex2011/results/ace/nmsd2.html

3. OPEN EVALUATION OF MIR SYSTEMS

Summarizing the previous section, MIREX suffers from three deficiencies that render the situation untenable: (i) the financial and labor costs cannot be sustained indefinitely, (ii) the lack of data transparency limits the scientific value of the endeavor, (iii) the lack of a strategy for obtaining and annotating new data.

Thus, to address these deficiencies, our proposed plan has three key differentiators from the MIREX model:

(i) Distributed computation reduces operating costs to scale favorably with increased participation.

(ii) Freely distributable audio facilitates reproducibility and benefits the entire community.

(iii) Incremental evaluation reduces bias by keeping test data fresh, and provides a feasible strategy for collecting new data.

3.1 Distributed computation

The primary difficulty in running an evaluation campaign is computing the outputs of all participating systems. This difficulty stems from two sources. First is the obvious computational complexity of running m submissions over n inputs. Second is the less obvious "human" complexity entailed in the task captains successfully executing the submitted programs in a foreign environment. While the computational issues can be ameliorated by running systems in parallel over multiple machines, the cost in human effort has no easy solution in the MIREX framework.

An alternative to this framework is exemplified by Kaggle. [3] Kaggle competitions are conducted with all input data publicly available, and participants submit only the outputs of their systems, *e.g.*, predictions made for each data point. This paradigm effectively resolves the difficulties listed above: both computation and human effort are distributed to the participants, rather than centralized at the evaluation site and task captains. This dramatically reduces the cost of maintenance and administration. However, we note a few potential challenges inherent to bringing data to the computation.

First, the input data must be made openly available to participants. This may increase the risk of bias if participants (unintentionally) tune their systems to the evaluation set. To mitigate bias, we propose the use of a large and diverse corpus of common tracks which are shared across *all tasks*, rather than a collection of small, task-specific datasets as is done in MIREX. For any given task, only a subset of the data need to be considered when comparing systems, and the evaluation set may be independently selected for each task. The knowledge of which items comprise a given evaluation subset would remain hidden from participants at submission time. This implies that each submission must span the entire corpus, reducing the feasibility of participants tuning their algorithms to a particular subset. Moreover, while this requirement increases computational overhead for the participant, it results in a large

[3] http://kaggle.com

collection of outputs for various methods on a common corpus, which is a valuable data source in its own right.

Second, distributed computation entails its own challenges with respect to transparency. While MIREX requires submissions to execute on a remote server beyond the participants' control, the scheme proposed here drops that requirement in favor of visibility into system inputs. Consequently, restrictions on the implementation (*e.g.*, running time) would become infeasible, and it may open the possibility of cheating by manual participant intervention. Using a large corpus with opaque evaluation sets will limit the practicality of this approach. [4] Obscuring which items belong to the evaluation subset comes at the expense of sample-wise measures, however, as doing so would reveal the partition. This is not inherently problematic if done following the completion of a campaign (instead of powering a continuous leader-board), but would require changing the evaluation set annually. These are minor concessions, and we argue that open data benefits the community at large, since its availability will serve a broader set of interests over time.

Finally, the proposed scheme assumes that multiple tasks operate on a common input form, *i.e.*, recorded audio. While the majority of current MIREX tasks do fit into this framework, at present it precludes those which require additional input data (score following, singing voice separation) or user interaction (grand challenges, query-by-humming/tapping). Our long-term plan is to devise ways of incorporating these tasks as well, while keeping with the principles outlined above. This is of course an open question for further research.

3.2 Open content

For the distributed computation model to be practical, we first need a source of diverse and freely distributable audio content. This is significantly easier to acquire now than when MIREX began, and in particular, a wealth of data can be obtained from services like Jamendo [5] and the Free Music Archive (FMA). [6] Both of these sites host a variety of music content under either public domain or Creative Commons (CC) licenses. [7] Since CC-licensed music can be freely redistributed (with attribution), it is (legally) possible to create and share persistent data archives.

The Jamendo and FMA collections are both large and diverse, and both can be linked to meta-data: Jamendo via DBTune to MusicBrainz [19] and FMA to Echo Nest/Spotify identifiers. Jamendo claims over 500,000 tracks charted under six major categories: *classical*, *electronic*, *hip-hop*, *jazz*, *pop*, and *rock*. FMA houses approximately 90,000 tracks which are charted under fifteen categories: *blues*, *classical*, *country*, *electronic*, *experimental*, *folk*, *hip-hop*, *instrumental*, *international*, *jazz*, *old-time/historic*, *pop*, *rock*, *soul/rnb*, and *spoken*. These categories should not necessarily be taken as ground truth annotations, but they reflect the tastes and priorities of their respective user communities. While there is undoubtedly a strong western bias in these corpora, the same can be said of MIREX's private data and the MIR field itself. However, using open content at least permits practitioners to quantify and possibly correct this bias.

Aside from western/non-western bias, there is also the potential for free/commercial bias. A common criticism of basing research on CC-licensed music is that the music is of substantially lower "quality" — which may refer to either artistry or production value, or both — than commercial music. This point is obviously valid for high-level tasks such as recommendation, which depend on a variety of cultural, semantic, and subjective factors beyond the raw acoustic content of a track. However, for the majority of MIREX tasks, in particular perceptual tasks like onset detection or source identification, this is a much more tenuous case. We do, however, note that FMA includes content by a variety of commercially successful artists, [8] but the vast majority of content in both sources is provided by relatively unknown artists, which makes it difficult to control for "quality".

To help users navigate the collections, both Jamendo and FMA provide popularity-based charts and community-based curation, in addition to the previously mentioned genre categories. Taken in combination, these features can be leveraged to pre-emptively filter the collections down to subsets of tracks which are either of interest to a large number of listeners, of interest to a small number of listeners with specific tastes, or representative of particular styles. This approach is similar in spirit to previous work using chart-based sampling, *e.g.* Billboard [2].

3.3 Incremental evaluation

In its first cycle of operation, the proposed framework requires a new, unannotated corpus of audio. Rather than fully annotating the corpus up front for each task, we plan to adopt an *incremental evaluation* strategy [4].

With incremental evaluation, the set of reference annotations need not be complete: some (most) input examples will not have corresponding annotations. Consequently, performance scores are estimated over a subset of annotated tracks, which may itself grow over time. Systems are ranked as usual, but with a degree of uncertainty inversely proportional to the number of available annotations.

Initially, uncertainty in the performance estimates will be maximal, due to a small number of available annotations. Subsequently, as reference annotations are collected and integrated to the evaluation system, they will be compared to the submissions' estimated annotations, and provide incrementally accurate and precise performance estimates. Prior research in both text and video IR has demonstrated that evaluation against incomplete reference data is feasible, even when only a small fraction of the annotations are known [3, 15, 25].

[4] Even in the unlikely event that a participant "cheats" by obtaining human-generated annotations, the results can be publicly redistributed as free training data, so the community ultimately wins out.

[5] http://jamendo.com

[6] http://freemusicarchive.org/

[7] https://creativecommons.org/licenses/

[8] https://en.wikipedia.org/wiki/Free_Music_Archive#Notable_artists

This raises the question: which inputs are worth annotating? Consider two systems that produce the same annotation for a fixed input example. Whether they are both right or wrong with respect to the reference annotation, there is no way to distinguish between them according to that example, so there is little value in seeking its annotation. Conversely, examples upon which multiple systems *disagree* are more likely to produce useful information for distinguishing among competing systems. Several recent studies have investigated the use of algorithm disagreement for this issue, be it for beat tracking [8], structural analysis [13, chapter 4], or chord recognition [9]. Others have studied alternative methods for music similarity [22], choosing for annotation the examples that will be most informative for differentiating between systems. In general, these methods allow us to minimize the required annotator effort by prioritizing the most informative examples. With many participating systems, and multiple complex performance metrics, prioritizing data for annotation is by no means straightforward. We hope that the community will assist in specifying these procedures for each task.

In the proposed framework, annotations may come from different sources, so it is imperative that we can trust or validate whatever information is submitted as reference data. Another line of work is thus the development and enforcement of standards, guidelines, and tools to collect and manage annotations. This will require developing web services for music annotation, as well as appropriate versioning and quality control mechanisms. Quality control is particularly important if annotations are collected via crowd-sourcing platforms like Amazon Mechanical Turk. [9]

3.4 Putting it all together

In contrast to the outline in Section 2, our proposed strategy would proceed as follows:

1. Identify and collect freely distributable audio.
2. Define or update tasks, *e.g.*, chord transcription.
3. Release a development set (with annotation), and the remaining unlabeled data for evaluation.
4. Collect *predictions* over the unlabeled data from participating systems.
5. Collect reference annotations, prioritized by disagreement among predictions and informativeness.
6. Estimate and summarize each submission's performance against the reference annotations.
7. Retire newly annotated evaluation data, adding it to the training set for the next campaign.
8. Go to step 3 and repeat. Revisit steps 1–2 if needed.

Steps 1–3 essentially constitute the startup cost, and are unavoidable for tasks which lack existing, open data sets. However, from the perspective of administration, only steps 3 and 5 require significant human intervention (*i.e.*, annotation), and both steps directly result in public goods. In this way, the proposed system will be significantly more efficient and cost-effective than MIREX.

[9] `https://www.mturk.com/`

4. DISCUSSION

In this section, we enumerate the goals and implementation of the proposed framework.

4.1 Timing: why now?

The challenges described in this document, which our proposed strategy aims to address, are not news to the community. The operating costs of MIREX became apparent early in its lifetime, and concerns about its sustainability loom large among researchers. That said, little has been done to resolve the situation in the intervening years. This raises an obvious question: why will things change *now*?

In many ways, the approach taken by MIREX made perfect sense in the early 2000's. However, recent infrastructural advances, coupled with growth and maturation of the MIR community, have introduced new possibilities. First and foremost, creative commons music is now ample, bringing the dream of sharing data within grasp. Cloud computing is cheap and ubiquitous, and can dramatically reduce the administrative costs of evaluation infrastructure and persistent data storage. Improvements in web infrastructure have also resolved many of the challenges of large-scale data distribution. Finally, browser support for interactive applications enables the development of web-based annotation tools, which significantly reduces the barrier to entry for annotators.

More broadly speaking, the community has matured significantly since MIREX began in 2005. Open source development and reproducible methods are now commonplace, but we remain hindered by the lack of open data for evaluation. Only by developing frameworks for open evaluation and data collection, can we further develop as a scientific discipline.

4.2 Implementation details

Effectively deploying the proposed framework will require two things: infrastructure development, and hosting. On the infrastructure side, we can leverage several existing open source projects: `mir_eval` for scoring [18], JAMS for data format standardization [10], and CodaLab for running the evaluation campaign. [10] The last remaining software component is a platform for collecting annotations. In addition to traditional desktop software, browser-based annotation tools would facilitate distributed data collection, and a simple content management system could collect annotations as they are completed.

As for hosting, since the burden of executing arbitrary (submitted) code is removed, the remaining software components can reside on either a private university server, or, more likely, cloud-based hosting such as Amazon Web Services. [11] Similarly, the audio data can be distributed via BitTorrent to participants, or hosted (at some minor cost) for traditional download.

[10] `http://codalab.org/`
[11] `https://aws.amazon.com/`

4.3 Data and evaluation integrity

Allowing participants to submit predictions, rather than software, may raise questions about integrity: how can we verify the process which generated the predictions? Ultimately, we must rely on participants to be honest in describing their methods. Although it is *possible* for a participant to manually annotate each track and achieve high scores, we hope that the scale of the dataset will make this approach fundamentally impractical. Additionally, in keeping with the spirit of open science and reproducible methods, we will encourage participants to link to a repository containing their software implementation, which can be used to independently verify the predictions.

When it comes to data integrity, we acknowledge that music is unique in that its perception is impacted by a plethora of cultural and experiential factors. In particular, CC-licensed music may lie outside of the gamut of commonly studied music, and differences in compositional style, instrumentation, or production, may lead to difficulties in validation and generalization. While this is unlikely to impact low-level tasks such as onset detection or instrument identification, more abstract tasks, such as chord estimation or structural analysis, may be more sensitive to selection bias. Relying on curation and chart popularity as a pre-filter in data selection may help to mitigate these effects. After collecting an initial set of annotated CC-licensed music, it will be possible to quantitatively measure the differences in system performance compared to existing corpora of commercial music.

4.4 Collecting annotations

Statistically valid evaluation requires a continuous source of freshly annotated data. At present, we see three potential sources to satisfy this need.

First is the traditional route of raising funds and paying expert annotators. This option incurs both a direct financial cost and various hidden human costs, but is also the most likely to produce high-quality data, and may in fact be the only viable path for certain tasks. In the grand scheme of things, however, the financial burden may not be so severe. As a point of reference, MedleyDB, a well-curated dataset of over 100 multi-track recordings for a number of MIR tasks, cost approximately $12 per track to annotate ($1240 total) [1]. [12] The ISMIR Society maintains a membership of roughly 250 individuals each year: a $5 increase in membership dues would cover the annotation cost of a new dataset like MedleyDB annually. Grants or industrial partnerships could also subsidize annotation costs.

Second, for tasks that require minimal annotator training, we can leverage either crowd-sourcing platforms, *e.g.*, Mechanical Turk, or seek out enthusiastic communities interested in voluntary *citizen science* for music. Websites such as Genius [13] (6M monthly active users) and Ultimate Guitar [14] (3M monthly active users) demonstrate the existence of these communities for lyrics and guitar tablature. [15] As witnessed by the success of eBird [16] or AcousticBrainz, [17] motivated people with the right tools can play a substantial role in scientific endeavors.

Finally, if no funding can be found to annotate data, we may solicit annotations directly from participants and the ISMIR community at large. This approach has been effectively used to collect judgements for the audio and symbolic music similarity tasks.

In each case, we will institute a ratification system so that annotations are independently validated before inclusion in the evaluation set. As mentioned in section 4.2, web-based annotation tools will enable volunteer contribution, which can supplement paid annotations.

5. NEXT STEPS: A TRIAL RUN

We conclude by advocating a large-scale instrument identification task for ISMIR2017. In this task, the presence of active instruments (under a pre-defined taxonomy) are estimated over an entire recording. The taxonomy may be readily adapted from WordNet [12], and refined by the community, starting at this year's ISMIR conference. There are a number of strong motivations for pursuing instrument identification. It is an important, classic problem in MIR, but is currently absent from MIREX. Researchers typically explore the topic with disparate datasets, and the problem remains unsolved to an unknown degree. Compared with other music perception tasks, annotation requires a simple interface, *e.g.*, "check all that apply", and the task definition itself is relatively unambiguous: a particular instrument is either present in the mix or not. Instrument occurrence is largely independent of popularity, which results in a fairly minimal bias due to the use of CC-licensed music. Finally, computer vision found fantastic success with a similar endeavor, known as ImageNet [5], in which algorithms detect objects in natural images.

The following steps are necessary to realize this goal within the coming year: (i) establish a robust instrument taxonomy; (ii) acquire a large sample of audio content; (iii) build a web-based annotation tool and storage system; (iv) construct a development set; (v) implement or collect a few simple algorithms to prioritize content for initial annotation; (vi) perform data collection, through some combination of paid annotation, crowd-sourcing, and community support; and finally, deploy an evaluation server and leader-board to accept and score submissions.

Each of these components, while requiring some engineering and organizational efforts, are achievable goals with the help of the ISMIR community.

Acknowledgments. B.M. is supported by the Moore-Sloan Data Science Environment at NYU. J.U. is supported by the Spanish Government: JdC postdoctoral fellowship, and projects TIN2015-70816-R and MDM-2015-0502.

[12] Figures provided via personal communication with the authors.
[13] http://genius.com/
[14] http://www.ultimate-guitar.com/
[15] Data gathered from http://compete.com, March 2016
[16] http://ebird.org/
[17] http://acousticbrainz.org/

6. REFERENCES

[1] R.M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J.P. Bello. Medleydb: A multitrack dataset for annotation-intensive mir research. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 155–160, 2014.

[2] J.A. Burgoyne, J. Wild, and I. Fujinaga. An expert ground truth set for audio chord recognition and music analysis. In *International Society for Music Information Retrieval Conference*, ISMIR, 2011.

[3] B. Carterette. Robust Test Collections for Retrieval Evaluation. In *SIGIR*, pages 55–62, 2007.

[4] B. Carterette and J. Allan. Incremental test collections. In *ACM International conference on Information and Knowledge Management*, pages 680–687. ACM, 2005.

[5] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, CVPR, pages 248–255. IEEE, 2009.

[6] J.S. Downie. The music information retrieval evaluation exchange (2005-2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[7] J.S. Downie, A.F. Ehmann, M. Bay, and M.C. Jones. The music information retrieval evaluation exchange: Some observations and insights. In *Advances in music information retrieval*, pages 93–115. Springer, 2010.

[8] A. Holzapfel, M.E.P. Davies, J.R. Zapata, J.L. Oliveira, and F. Gouyon. Selective sampling for beat tracking evaluation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(9):2539–2548, 2012.

[9] E.J. Humphrey and J.P. Bello. Four timely insights on automatic chord estimation. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 673–679, 2015.

[10] E.J. Humphrey, J. Salamon, O. Nieto, J. Forsyth, R.M. Bittner, and J.P. Bello. JAMS: A JSON annotated music specification for reproducible MIR research. In *ISMIR*, pages 591–596, 2014.

[11] B. McFee, T. Bertin-Mahieux, D. Ellis, and G. Lanckriet. The million song dataset challenge. In *4th International Workshop on Advances in Music Information Research*, AdMIRe, April 2012.

[12] G.A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[13] O. Nieto. *Discovering structure in music: Automatic approaches and perceptual evaluations*. PhD thesis, New York University, 2015.

[14] N. Orio, D. Rizo, R. Miotto, M. Schedl, N. Montecchio, and O. Lartillot. Musiclef: a benchmark activity in multimodal music information retrieval. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 603–608, 2011.

[15] P. Over, J. Fiscus, G. Sanders, D. Joy, M. Michel, G. Awad, A.F. Smeaton, W. Kraaij, and G. Quenot. TRECVID 2014–An Overview of the Goals, Tasks, Data, Evaluation Mechanisms, and Metrics. In *TREC Video Retrieval Evaluation Conference*, 2014.

[16] G. Peeters and K. Fort. Towards a (Better) Definition of the Description of Annotated MIR Corpora. In *International Society for Music Information Retrieval Conference*, pages 25–30, 2012.

[17] G. Peeters, J. Urbano, and G.J.F. Jones. Notes from the ISMIR 2012 Late-Breaking Session on Evaluation in Music Information Retrieval. In *International Society for Music Information Retrieval Conference*, 2012.

[18] C. Raffel, B. McFee, E.J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D.P.W. Ellis. mir_eval: A transparent implementation of common mir metrics. In *International Society for Music Information Retrieval Conference*, ISMIR, 2014.

[19] Y. Raimond and M.B. Sandler. A web of musical information. In *ISMIR*, pages 263–268, 2008.

[20] X. Serra, M. Magas, E. Benetos, M. Chudy, S. Dixon, A. Flexer, E. Gómez, F. Gouyon, P. Herrera, S. Jorda, O. Paytuvi, G. Peeters, J. Schlüter, H. Vinet, and G. Widmer. Roadmap for Music Information ReSearch, 2013.

[21] B.L. Sturm. The state of the art ten years after a state of the art: Future research in music information retrieval. *Journal of New Music Research*, 43(2):147–172, 2014.

[22] J. Urbano and M. Schedl. Minimal Test Collections for Low-Cost Evaluation of Audio Music Similarity and Retrieval Systems. *International Journal of Multimedia Information Retrieval*, 2(1):59–70, 2013.

[23] J. Urbano, M. Schedl, and X. Serra. Evaluation in Music Information Retrieval. *Journal of Intelligent Information Systems*, 41(3):345–369, 2013.

[24] E.M. Voorhees and D.K. Harman. *TREC: Experiment and Evaluation in Information Retrieval*. MIT Press, 2005.

[25] E. Yilmaz, E. Kanoulas, and J.A. Aslam. A Simple and Efficient Sampling Method for Estimating AP and NDCG. In *SIGIR*, pages 603–610, 2008.

GO WITH THE FLOW: WHEN LISTENERS USE MUSIC AS TECHNOLOGY

Andrew Demetriou [‡] **Martha Larson** [‡§] **Cynthia C. S. Liem** [‡]

[‡] Delft University of Technology, Delft, The Netherlands

[§] Radboud University, Nijmegen, The Netherlands

andrew.m.demetriou@gmail.com {m.a.larson, c.c.s.liem}@tudelft.nl

ABSTRACT

Music has been shown to have a profound effect on listeners' internal states as evidenced by neuroscience research. Listeners report selecting and listening to music with specific intent, thereby using music as a tool to achieve desired psychological effects within a given context. In light of these observations, we argue that music information retrieval research must revisit the dominant assumption that listening to music is only an end unto itself. Instead, researchers should embrace the idea that music is also a technology used by listeners to achieve a specific desired internal state, given a particular set of circumstances and a desired goal. This paper focuses on listening to music in isolation (i.e., when the user listens to music by themselves with headphones) and surveys research from the fields of social psychology and neuroscience to build a case for a new line of research in music information retrieval on the ability of music to produce flow states in listeners. We argue that interdisciplinary collaboration is necessary in order to develop the understanding and techniques necessary to allow listeners to exploit the full potential of music as psychological technology.

1. INTRODUCTION

When the word *technology* is used in the context of music, it generally relates to the development of new digital devices or algorithms that support the production, storage, and/or transmission of music. In this paper we break from the conventional use of the word *technology* in regards to music, reprising a conception of music as a technology *in and of itself.*

In order to understand precisely what *music as technology* means, it is helpful to take a closer look at the meaning of the word *technology*. Specifically, we use *technology* in the sense of *a manner of accomplishing a task especially using technical processes, methods, or knowledge*[1]. We do not contradict the generally accepted perspective that music may exist for its own sake. However, we do take the position that other considerations may also be at stake when listeners listen to music. Specifically, we hold that there are cases when listeners use music as a tool that is directed towards accomplishing a task. In these cases, music can be considered as part of a method applied by listeners to achieve a goal.

The notion of music as technology was already coined in the area of sociology by DeNora at the end of the millennium [8]. This work characterized music as part of the continuing process of self-development, and posited that individuals use it to maintain and develop a social identity as well as a means to self-regulate emotions, moods, energy levels, or for the purposes of 'self care'. In effect, it was suggested that people outsource various sorts of 'emotional work' to music, based on their goals within a given context.

We argue that the moment is now ripe for the music information retrieval (MIR) community to revisit this notion. In the intervening years, social psychology and neuroscience have considerably advanced our understanding of how music is used in everyday life, and how it effects the brain. Further, music recommender systems show signs that they are already reorienting themselves from music that users "like" to music that users find useful in a particular situation. This development is evident in the evolution of how the purpose of music recommender systems is described in the literature. A 2002 publication [36] characterized this purpose as recommending *music that the user will be interested in*, which contrasts with the statement of a 2011 publication [12] that *a good recommendation system should...maximize the user's satisfaction by playing (the) appropriate song at the right time*. Currently, the unprecedentedly large amount of music available online offers new possibilities of finding a tight fit with listener needs. Reflecting this focus, a 2015 publication [32] stated the purpose of music recommender systems to *provide guidance to users navigating large collections*. We draw on these contemporary findings and theory to understand how users may better use music as a tool in everyday life.

The contribution of this paper is to revisit and update the notion of music as technology, and to link it to a Call to Action for MIR and neighboring psychology-oriented communities. It should be noted that the socio-psychological concept of music preference as a potential indicator of personality, values and beliefs (and as a 'social badge') is relevant to music consumption behavior, fitting into the concept of considering music as a technology (to establish belonging), and not yet taken into account sufficiently in the context of music recommender

© Andrew Demetriou, Martha Larson, Cynthia C. S. Liem. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Andrew Demetriou, Martha Larson, Cynthia C. S. Liem. "Go with the Flow: When Listeners User Music As Technology", 17th International Society for Music Information Retrieval Conference, 2016.

[1] http://www.merriam-webster.com/dictionary/technology

systems [19]. However, in our paper the focus will not be on social listening, but rather on the complementary situation in which the listener consumes music on their own, in relation to achieving a personal goal.

In our consideration of a technological role of music, we go beyond 'self-care', and describe music as a tool that a listener may use to achieve the internal state necessary to accomplish their goal. We hypothesize that this connects to the concept of flow [24]: a desirable internal state that has been characterized by complete and total attention, a loss of a sense of self, a loss of a sense of the passing of time, and the experience that conducting the activity is, in and of itself, intrinsically rewarding. In other words, a listener in flow state is enjoying the feeling of being absorbed in their task to such a degree that the passing of time is not noticed, and is therefore able to push past obstacles to carry out activities and achieve goals. In later sections, we will elaborate on theories regarding the possible neurophysiological nature of flow states, the effects of music on the brain, and how it is that music may assist in achieving these internal states. As an initial indication of the growing importance of music that allows users to accomplish goals, we point to the growing number of artists[1] and services[2] on the Internet that are providing music to help people focus.

The idea of music as technology should not be considered a paradigm shift, but rather as the explicit identification of a common phenomenon. This phenomenon has thus far escaped the attention of the MIR community because the focus of music information retrieval research has been firmly set on *what music is*, rather than on *what music does*. However, there are many examples of work that illustrates the breadth of areas in which music is used as a tool to accomplish an end. Most widely known is perhaps the use of music as a meaning-creating element in storytelling, especially in film and video, e.g., [35]. Currently expanding is the use of music in branding, e.g., within the rise of the concept of corporate audio identity [2]. Less comfortable to contemplate is the use of music for torture e.g., as studied by Cusick [7]. Finally, we mention the therapeutic uses of music, as covered recently by Koelsch [17].

Our work differs in a subtle, but important way from these examples. We look at music as technology from the point of view of listeners who make a conscious decision to *expose themselves to the experience of music to alter their internal state in order to achieve a goal that they have set for themselves*. Later, we will return to the importance of listener control over the choice of music for the effectiveness of music as a tool.

Music as technology has serious implications for music information retrieval. If listeners may choose to use music as a psychological tool, then it is important for music search engines and recommender systems to be sensitive to the exact nature of the task that users wish to accomplish. It also is important for researchers to judge the success of these systems in terms of their ability to support users towards accomplishing tasks.

To understand music as technology more profoundly and fundamentally, collaborations between MIR and the neuro-, cognitive, and social psychological sciences, will be essential. Joint research lines involving collaborations between these fields will allow for the potential to determine when and how flow states occur, if they vary in any way based on context, and how exactly these states are aided by music.

In summary, this implies two places in which the MIR community should be active: i) learning and understanding what users need to put themselves into a flow state, and how this depends on what they are doing and on the surrounding circumstances, and ii) understanding how new music search engines and recommender systems can be designed to allow listeners to achieve flow states.

In the remainder of this paper, we first will review how music is used as part of daily life. After this, we consider the effects of music on the brain, subsequently connecting to insights in relation to achieving flow state. Based on our proposed viewpoint and the reviewed literature, we discuss how the MIR research agenda can be broadened in this light, and finish with a Call to Action for interdisciplinary work worth investigating.

2. LISTENERS USING MUSIC

2.1 Music as part of daily life

In the everyday life of the modern human, music has become a constant accompaniment to all manner of daily activities [27, 29, 34]. The advent of portable music devices capable of housing vast collections, the ubiquity of available musical data via streaming services, and the development of technology that allowed for greater ease of music production, have all lead to the consumption of music on an increasingly individual basis across an increasingly broad range of activities and contexts [10].

Music listening is a common occurrence in everyday life, yet rarely the sole focus of an activity. A number of studies have pointed to this conclusion, and we mention some key examples here. In an experience sampling study where participants completed brief surveys at random intervals throughout their day, 44% of the surveys were completed while music listening had taken place within any 2-hour period, yet less than 2% of episodes involved listening to music as a main activity [32]. A later study showed that 38.6% of text messages sent to participants randomly throughout the day occurred during music listening occasions; on occasions where the participants were not listening to music, 48.6% indicated that they had listened to music since the last text message, yet only 11.6% of these episodes occurred when music listening was the main activity [27]. A more recent survey study has shown similar results, with respondents indicating a mean of less than 1 hour of active music listening per day, yet 2-4 hours of passive music listening [15].

Along with an increase in music consumption accompanying other activities is the emergence of the belief that individual music selections function as a means to

[1] e.g., Delta Notch, https://www.youtube.com/user/DDRfrosh1
[2] e.g., Focus at Will https://www.focusatwill.com

achieve various emotional, motivational, or cognitive effects to the benefit of accomplishing various activities [27]. Individuals will report that music is expected to perform different functions based on different situations [26], an awareness of the specific songs expected to fulfill these functions, as well as the expected psychological benefits from listening [8]. As such, people have come to use music as a piece of technology in their daily lives, effectively attempting to outsource various psychological tasks to specific song selections. We now go on to discuss the factors that contribute to listeners successfully using music to achieve internal states that may be described as flow, which, in turn, support activities or goals.

2.2 Choosing music for a purpose

As mentioned above, our perspective on music as technology regards music as a tool in the hands of listeners themselves. In this section, we examine in more detail the importance of listener control of music. The perceived benefits of music listening have been shown to be more positive when the individuals had the ability to choose the desired music [27]. Participants indicate preferring playlists they created rather than automatically curated content [15], and those who chose the music they were listening to reported enjoying it more [27]. Furthermore, with greater control on the choice of music selection, individuals reported experiencing greater changes in mood along three bipolar factors: 1) positivity, 2) present mindedness, and 3) arousal [34].

Listeners' preference for control is consistent with the idea of music being a means to an end. A number of studies have shown that listeners use music as psychological tool to optimize emotion, mood, and arousal based on the very specific needs of a given situation and/or activity [8, 27, 34]. Interviews have shown that individuals have an awareness of specific songs they feel will assist in accomplishing various emotional tasks, such as decreasing or increasing their arousal, motivating them to take action, adjusting their moods, or assisting them to focus [8]. Reasons for listening to music have also been shown to vary by activity (e.g., doing housework, travelling, studying, dating, getting dressed to go out etc.) [8, 15, 29].

Along with the constant growth of the music corpus, a means to organize, retrieve and discover appropriate music selections is a growing challenge. Despite the prevalence of current playlist curation technologies, individuals report self-generated playlists to be the organizational method of choice [8, 15], an indication of the specificity of song selection requirements, above and beyond the specificity of individual preference. In the final section of the paper, we will return to discuss how, in order to use *music as technology*, users must have at their disposal appropriate music information retrieval technology. Next we turn to the neuroscience perspective on music as technology.

3. MUSIC AND THE BRAIN

Research in the field of music and emotion suggests there are multiple means for music to affect the individual, and that underlying physiological and neurological mechanisms should be researched [14]. We highlight two posited mechanisms relevant to our discussion: a) brain stem reflexes, and b) musical expectancy.

The degree and manner in which each mechanism results in a physiological or neurological response, and by extension arousal, may be key in understanding why listeners select specific songs given the tasks they have set out to accomplish. As the demands of each situation vary, the effect of acoustic stimuli on the brain of the listener may function to moderate arousal such that an optimal internal state is reached. In other words, listeners may be selecting songs, and by extension sequences of acoustic stimuli, to alter their internal state in order to best meet the needs of their situation.

3.1 Brain stem responses

The brain stem is believed to be a very old part of the brain, and has been shown to be sensitive to loud, low frequency, dissonant, suddenly changing sounds [5, 9, 22]. It is posited that sounds indicative of a sudden change, a strong force, or something of large size may coincide with an event that requires immediate, urgent and reflexive attention. These acoustic qualities shift attention to the stimulus, giving rise to muscular and cardiovascular responses as well; a by-product of this may be the reason bass drum sounds inspire people to dance in sync with the music, and why music with faster tempos is more arousing (see [14] and [17]). Furthermore, a greater number of brain regions have shown activation at the onset of musical samples as opposed to the middle or end of these samples [23].

As such, music that contains such acoustical stimuli, or dramatic changes in its acoustic features (e.g., dramatic build ups and “drops”), may shift attention to the music arousing the listener in the process. Conversely, music that is relatively constant may instead serve to 'drown out' distracting ambient sounds instead: for example, the difference between silence and the rustling of papers is far greater than the difference between the rustling of papers and background music. As such, music may provide a constant acoustic backdrop thereby reducing the amount of arousal and attentional shifts caused by distracting sounds in the listener’s environment.

3.2 Musical expectancy

Recently, an increasing amount of attention has been devoted to expectancy as it relates to music (e.g., as in Huron's recent work [11]). The ability of the human brain to predict events is thought to have been vital to survival, and thus plays a prominent role in all cognition. As such, meeting or violating expectations in music should result in physiological and neurological effects (see [30] and [31]). Given that music is essentially an organized pattern of sounds, our brains generate predictions as the music unfolds over time based on our knowledge of the specific musical piece, but also our knowledge of all music [31].

As only so much information may be encoded at a time, the more complex the piece, the greater the number of potential prediction errors, the more exposure is required to become familiar [31]. In fact, as far back as Berlyne's [3] studies, it has been shown that familiarity of a particular sequence of notes in relation to a corpus results in less physiological arousal than unfamiliar sequences of notes,

as does simplicity in the melody as opposed to complexity. These expectations may be used deliberately by composers of music to create a sense of musical tension, only to resolve the tension later on in the piece, resulting in relaxation and pleasure [18]. In addition, familiarity of a piece may lead to anticipation of the pleasure to be experienced at peak moments in the music, resulting in the activation of midbrain dopamine neurons causing attention to be paid to potential upcoming rewards [31].

Relevant to our topic, such arousal may divert attention from the task to the music [e.g., 13]. On one hand, music that adheres to expectations, such as a collection of very familiar pieces, may result in less overall arousal than pieces that are unfamiliar, very complex, or of an unfamiliar genre. On the other hand, familiar pieces that result in pleasure and anticipation may also be arousing, diverting attention from the task to the music as well.

4. MUSIC AND FLOW

Flow is characterized as a mental state in which one's complete attention is focused on a task, one has lost sense of self and of time, and one's perception of the experience is positive and rewarding [24]. In this research tradition, the definition of flow also includes a sense that one's subjective level of skill is balanced with the subjective challenge of the activity: a too-simple task evokes relaxation then boredom which in turn causes attention to drift, and a too-challenging task evokes vigilance then anxiety [24]. As with music use in everyday life, the concept of flow is also intertwined with context and activity.

More recently it has been theorized that flow states may emerge during media enjoyment, resulting in neural states where attentional and reward centers in the brain are activated synchronously [40]. Weber and colleagues [40] drew a theoretical link between engagement in linear media (e.g., books, films and video games) and flow states. They posit that linear media require mastery of mental models: video games require a level of skill that increases as one progresses, and films require an understanding of the characters and the narrative. It is suggested that these contribute the challenge, which in addition to pleasurable engagement, coincides with activations of the brain regions necessary to achieve flow. While music is not specifically discussed, it is a medium that can be consumed during various activities, and may function in conjunction with these activities to inspire flow states.

The dopaminergic pathway, which is involved in the experience of pleasure, is posited to be active during flow states [40], and has been shown to be active during experiences of pleasure while listening to music [31]. Of interest in this pathway is the nucleus accumbens, which is also thought to be involved in automatic consummatory behavior (e.g., drinking or eating), and the striatum which also has connections to the brain stem [40]: both also been observed in pleasurable responses to music [31]. In addition, regions thought to be involved in reward-seeking behaviors, such as the prefrontal and orbitofrontal cortices have also been implied in both [31] [40].

While it is not yet clear how specifically music and context may interact to produce a flow state, enough evidence has been accrued for us to suggest two aspects worthy of study. Firstly, during tasks in which boredom is likely, more arousing music may be selected to induce a flow state: by diverting attentional resources to the music the challenge of the task increases, as it now requires attention to be paid to both the activity and the music. As such, music that is more likely to be arousing either by a) resulting in responses from the brain stem (e.g., loud, frequently changing, or dissonant song selections) or b) causing prediction errors (e.g., less familiar, familiar and causing anticipation, or more complex) may be more suitable. Secondly, during tasks that are challenging or otherwise cognitively engaging (e.g., studying or reading) music that is likely to be less arousing either by a) resulting in less brain stem activation (e.g., relatively unchanging or consonant) or b) being predictable without anticipation (e.g., somewhat familiar and somewhat liked, more simple songs) may be more suitable.

5. NEW CHALLENGES FOR MIR

We now turn back to discuss how *music as technology* connects with MIR. The ability of listeners to successfully use music as technology depends on the effectiveness of music information retrieval and recommender systems in supporting them. We argue for the necessity of multidisciplinary research that brings together neuro-, cognitive, and social psychologists, and music information retrieval researchers. Such collaboration will allow us to understand what makes music helpful for users and what makes it appropriate for different tasks. In this section, we point to several areas in which the music information retrieval is on the right track, and several areas in which more effort is needed if users are to truly benefit from music as technology.

First, we return to the relation between the user choosing music, and music being perceived as having positive benefits. Taking this connection seriously means taking the position that for music to be used effectively as technology, it must truly be a tool in the users' hands (i.e., fully under the control of the user). Other work that points out the critical role of user control over music selection includes [38], who observe that the context and the intentions of the user impact which music features are important. Their music selection interface provides users with control over factors such as tempo, mood, and genre, and their experiments show that users prefer this control. The findings are not surprising given the role of control in the success of recommender systems from the user point of view [28]. In order to make music a useful tool, MIR must start with the choice of the listener to change their internal state in order to accomplish a goal. The choice may be semi-conscious, or may simply consist of going to a place where certain music is playing, or accepting to stay in that place. Listeners who are unwilling or who are not themselves in control are not using music as technology. In other words, piping in focus music during an exam can be predicted not to improve students' ability to concentrate. MIR systems can make music useful as technology by providing results and recommenda-

tions that are transparent. The importance of transparency for recommender systems has long been recognized [33]. They should also minimize the effort needed from the user to provide feedback.

Second, serving listeners who want to use music as a tool requires extending today's context-aware recommender systems, which are described, for example, in [32]. Particularly promising is the development of systems to recommend music for activities, e.g., [39]. In [25] the authors propose a context-aware music recommendation system the monitors heart rate and activity level, and recommends music that helps the user achieve a desired heart rate intensity. The challenge of such activity-based recommenders is to provide music that serves the common needs of people engaging in an activity, while taking personal taste into account. One aspect of using music as technology is blocking out background noise. Context-aware recommenders will need to develop to be sensitive to the acoustic environment, so that they can recommend music that will mask it.

A challenge that has yet to be faced is moving music recommendation and retrieval away from music that listeners "like" the first time that they hear it, towards music that allows them to meet their goals. Currently, the ground truth that is used to evaluate the success of recommender systems does not differentiate “love at first listen” from an appreciation that a listener develops over a longer period of time on the basis of utility given the context and activity.

We suggest that collaboration between MIR and psychology may be appropriate to best determine not only how music can better be organized to suit different tasks, but also which specific features make certain music helpful, or make one selection more suitable for a given activity than another.

Recent years have seen progress in content-based and hybrid music recommender systems [32]. These systems make use of timbral features (e.g., MFCCs), features related to the temporal domain, such as rhythmic properties, and tonal features such as pitch-based features. Our discussion revealed the importance of content features that might point to a sudden, unexpected event in the music that would shift the listener’s attention. We point out that recent approaches to exploiting music content may only use very short segments of the music, such as the deep learning approach in [37]. A future challenge is to determine how long a window must be considered in order to determine whether the song contains features that disrupt focus. Here again, task specific as well as user-specific aspects are important.

Further, the role of familiarity is critical. The importance of music freshness is well recognized. For example, Hu and Ogihara [12] relate it to a memory model. However, playing the same familiar music repeatedly does not promote focus if the user's sense of anticipation becomes too strong. With the vast amounts of music currently available online, the possibility is open to creating a music recommendation system that never repeats itself.

When music is used as technology, it is important to keep in mind that it is the stream and not the individual song that is important. Currently, an increasing amount of work is carried out in the area of playlist recommendation [4]. Whereas many playlists are played on shuffle, playlists that most effectively allow the user to achieve internal state transformation may have a particular order, calling for more work on the generation of ordered streams of content items.

Finally, we anticipate that when listeners use music as technology they will want the possibility to query the system, instead of relying on a recommendation. Such queries, even though context-based, may not be well fitted to the goal that they want to accomplish. Here, it is necessary to understand the type of language that users use to express the complexity of their task. To this end, the MIR community should further foster insights in information seeking and user studies. However, an important difference with the existing paradigms under which these studies are conducted (e.g., [6, 20]) is that under the ‘music as technology’ paradigm, a query would be expressed in the form of a (non-musical) task to be accomplished, rather than a directed query to an explicit song (e.g., similarly to what was done in [21] on music and narrative).

6. CALL TO ACTION

In this work, we pointed out the notion of music as technology, which we feel currently is overlooked in MIR solutions. Connecting this concept to existing literature from the psychological sciences, it is clear that pursuing a joint research roadmap will be beneficial in both gaining fundamental insights into processes and internal states of listeners, and finding ways to improve music search engines and recommender systems. To concretize this further, we conclude this paper with a Call to Action, formulating interdisciplinary research directions, which will be beneficial for realizing the full potential of music as technology.

First, research should contribute to a better understanding of flow states. The evidence brought together in this paper points to the conclusion that flow is a desirable overarching internal state, and is the target state underlying a wide range of activities. We further argued that listeners choose music that complements an activity to result in a net optimal level of cognitive engagement. Under this view, music is not an end unto itself, but rather an inextricable part of the activity. More research is needed to validate flow as an overarching mental state in practice, as well as its antecedents. In addition, how music leads to and moderates flow state should be investigated.

Second, on the basis of a deeper understanding of flow, research should work to define new relevance criteria for music. Such work will involve understanding which kinds of music fit which kinds of tasks, zeroing in on the relevant characteristics of the music. We expect this to be a formidable challenge, since it must cover perceptual, cognitive, and social aspects of music. The contribution of users’ personal music experiences and music tastes must also be understood. On the one hand, we anticipate a certain universal character in the type of music that will allow a person to achieve flow state for a given activity. On the other hand, we anticipate that a ‘one size fits all’ solution will not be optimal, and that relevance criteria

must also be flexible enough to capture individuals' needs and preferences.

Third, once we have defined relevance criteria, we should move from there to identify new features, new algorithms, and new system designs. We anticipate that features reflecting music complexity and unexpectedness will be important, as a few relatively isolated disruptive moments can potentially make an entire song unsuitable for an activity. This observation points to the need to consider the song as a whole, implying, in turn, new MIR algorithms. New system designs will be needed to help guide users' music choice without effort, and ideally without interrupting their flow state. System designs will need to take into account that users may not recognize the music that will make them most productive the first time they hear it. Further, even after listeners recognize the connection between certain music and their own productivity levels, they might not be able to express their music needs explicitly in music-technical terms. Systems must be able to accommodate the types of information and feedback that users are able to provide about the kind of music that will be most effective for them.

Finally, once new applications have been developed and deployed, they will provide an extremely valuable source of information about when listeners use music, allowing neuroscientists and psychologists to refine their theories of flow and how listeners achieve it in certain situations, against the backdrop of scalable and real-world use cases.

Our suggestion for MIR and the (neuro)psychological sciences to connect is not new; for example, it also was reflected upon in [1], and recently further interconnection possibilities between the disciplines were suggested in [16]. Both of these works rightfully point out that such collaborations are not trivial, particularly because of methodological differences and mismatches. However, we believe that the currently described possibilities offer fruitful research questions for all disciplines.

Ultimately, understanding music as technology has the potential to profoundly impact not only the MIR domain, but the whole ecosystem of music production, delivery and consumption. Currently, the success of music is judged by the number of downloads or the number of listens. The idea of music as technology opens up the possibility of evaluating the success of music also in terms of the goals that are achieved by listeners.

Besides considering music as technology, we believe that we also should continue to study and enjoy music for its own sake. However, the potential of music to help listeners achieve their ends opens the way for creative new uses of music, with respect to commercial business models, as well as promoting the well-being of listeners. We hope that ultimately, music as technology will support listeners in coming to a new understanding on how they can use music to reach their goals and improve their lives.

Acknowledgments: The research leading to these results was performed in the CrowdRec and the PHENICX projects, which have received funding from the European Commission's 7th Framework Program under grant agreement no. 610594 (CrowdRec) and no. 601166 (PHENICX).

7. REFERENCES

[1] Aucouturier, J., and Emmanuel B. "Seven problems that keep MIR from attracting the interest of cognition and neuroscience," *JIIS,* 41.3, 483-497. 2013.

[2] Bartholmé, R. H. & Melewar, T.C.: "The end of silence? Qualitative findings on corporate auditory identity from the UK," *Journal of Marketing Communications*, 29 Oct 2014, pp.1-18. 2014..

[3] Berlyne, D. E.: *Aesthetics and psychobiology,* New York: Appleton-Century-Crofts, Vol. 336, 1971.

[4] Bonnin, G. & Jannach, D.: "Automated Generation of Music Playlists: Survey and Experiments," *ACM Comput. Surv*. 47, 2, Article 26, 35 pages, 2014.

[5] Burt, J. L., Bartolime, D. S., Burdette, D. W., & Comstock, J. R: "A psychophysiological evaluation of the perceived urgency of auditory warning signals," *Ergonomics*, 38(11), 2327–2340, 1995.

[6] Cunningham, S. J. & Bainbridge, D: "A search engine log analysis of music-related web searching," In *N.T. Nguyen et al. (Eds.), Advances in Intelligent Information and Database Systems: Studies in Computational Intelligence*, Vol. 283, pp. 79-88, 2010.

[7] Cusick, S.: "You are in a place that is out of the world: Music in the Detention Camps of the 'Global War on Terror,'" *JSAM,* Vol. 2/1, pp. 1-26, 2008.

[8] DeNora, T.: "Music as a technology of the self." *Poetics*, 27(1), 31–56, 1999.

[9] Foss, J. A., Ison, J. R., Torre, J. P., & Wansack, S.: "The acoustic startle response and disruption of aiming: I. Effect of stimulus repetition, intensity, and intensity changes," *Human Factors*, 31(3), 307–318. 1989.

[10] Hargreaves, D. J., & North, A. C.: "The Functions of Music in Everyday Life: Redefining the Social in Music Psychology," *Psychology of Music*, 27(1), 71–83, 1999.

[11] Huron, D.: "Sweet Anticipation: Music and the Psychology of Expectation," *Music Perception*, 24(5), 511–514, 2007.

[12] Hu, Y. and Ogihara, M.: "NextOne Player: A music recommendation system based on user behavior," *12th Int. Society for Music Information Retrieval Conference (ISMIR'11),* pp. 103-108, 2011.

[13] Jung, H., Sontag, S., Park, Y. S., & Loui, P.: "Rhythmic effects of syntax processing in music and language," *Frontiers in Psychology*, 6(NOV), pp. 1–11, 2015.

[14] Juslin, P. N., & Västfjäll, D.: "Emotional responses to music: The need to consider underlying mechanisms," *BBS*, 31(06), 751, 2008.

[15] Kamalzadeh, M., Baur, D., & Möller, T.: "A Survey on Music Listening and Management Behaviours,"

13th Int. Society for Music Information Retrieval Conference (ISMIR'12), pp. 373–378, 2012.

[16] Kaneshiro, B., and J. P. Dmochowski. "Neuroimaging methods for music information retrieval: Current findings and future prospects," *16th Int. Society for Music Information Retrieval Conference (ISMIR'15),* pp. 538–544, 2015.

[17] Koelsch, S.: "Brain correlates of music-evoked emotions," *Nature Reviews Neuroscience*, 15(3), 170–180, 2014.

[18] Koelsch, S.: "Music-evoked emotions: principles, brain correlates, and implications for therapy." *Annals of the New York Academy of Sciences*, 1337(1), 193–201, 2015.

[19] Laplante, A. Improving music recommender systems: "What can we learn from research on music tastes?" *15th Int. Society for Music Information Retrieval Conference (ISMIR '14),* pp. 451-456, 2014.

[20] Lee, J. H.: "Analysis of user needs and information features in natural language queries seeking music information." *Journal of the Association for Information Science and Technology (JASIST)*, 61(5), 1532-2890, 2010.

[21] Liem, C. C. S., Larson, M. A., & Hanjalic A.: "When Music Makes a Scene-Characterizing Music in Multimedia Contexts via User Scene Descriptions," *Int. Journal of Multimedia Information Retrieval*, 2, pp.15-30, 2013.

[22] Masataka, N., & Perlovsky, L.: "Cognitive interference can be mitigated by consonant music and facilitated by dissonant music," *Scientific Reports*, 3, 2028, 2013.

[23] Mueller, K., Fritz, T., Mildner, T., Richter, M., Schulze, K., Lepsien, J., Möller, H. E.: "Investigating the dynamics of the brain response to music: A central role of the ventral striatum/nucleus accumbens." *NeuroImage*, 116, 68–79, 2015.

[24] Nakamura, J., & Csikszentmihalyi, M.: "The Concept of Flow," In *J. S. Snyder & S. J. Lopez (Eds.), Flow and the Foundations of Positive Psychology*, pp. 239–263. New York: Oxford University Press, 2014.

[25] Nirjon, S., Dickerson, R. F., Li, Q., Asare, P., Stankovic, J. A., Hong, D., Zhang, B., Jiang, X., Shen, G., and Zhao, F.: "MusicalHeart: a hearty way of listening to music," *10th ACM Conference on Embedded Network Sensor Systems (SenSys '12). ACM,* New York, NY, USA, 43-56, 2012.

[26] North, A. C., & Hargreaves, D. J.: "Situational influences on reported musical preference." *Psychomusicology: A Journal of Research in Music Cognition*, 15(1-2), 30–45, 1996.

[27] North, A. C., Hargreaves, D. J., & Hargreaves, J. J.: "Uses of Music in Everyday Life," *Music Perception: An Interdisciplinary Journal*, 22(1), 41–77, 2004.

[28] Pu, P., Chen, L., and Hu, R.: "A user-centric evaluation framework for recommender systems." *Proceedings of the 5th ACM conference on Recommender systems (RecSys '11)*, pp. 157-164, 2011.

[29] Rentfrow, P. J., & Gosling, S. D.: "The do re mi's of everyday life: The structure and personality correlates of music preferences," *JPSP,* 84(6), 1236–1256, 2003.

[30] Rohrmeier, M. A., & Koelsch, S.: "Predictive information processing in music cognition: A critical review." *Int. Journal of Psychophysiology,* 83(2), 164–175, 2012.

[31] Salimpoor, V. N., Zald, D. H., Zatorre, R. J., Dagher, A., & McIntosh, A. R.: "Predictions and the brain: How musical sounds become rewarding," *Trends in Cognitive Sciences,* 19(2), 86–91, 2015.

[32] Schedl, M., Knees, P., McFee, B., Bogdanov, D., & Kaminskas, M.: "Music Recommender Systems," *In F. Ricci, L. Rokach, B. Shapira, (eds.) Recommender Systems Handbook (2nd ed.),* pp. 453-492. Springer US., 2015.

[33] Sinha, R. and Swearingen, K.: "The role of transparency in recommender systems," *CHI '02 Extended Abstracts on Human Factors in Computing Systems (CHI EA '02). ACM*, New York, NY, USA, 830-831, 2002.

[34] Sloboda, J. A, O'Neill, S. A. & Ivaldi, A.: "Functions of music in everyday life: an exploratory study using the experience sampling method," *Musicae Scientiae*, 5(1), 9–32, 2001.

[35] Tagg, P. and Clarida, B.: "Ten Little Title Tunes: Towards a Musicology of the Mass Media," *The Mass Media Scholar's Press,* New York, USA and Montreal, Canada, 2001.

[36] Uitdenbogerd, A. and van Schnydel, R.: "A review of factors affecting music recommender success," *3rd Int. Conference on Music Information Retrieval (ISMIR '02)*, Paris, France, 2002.

[37] A. van den Oord, S. Dieleman, and B. Schrauwen: "Deep content-based music recommendation," in *NIPS*, 2013.

[38] Vignoli, P. and Pauws, S.: "A Music Retrieval System Based on User-Driven Similarity and its Evaluation," *6th Int. Conference on Music Information Retrieval (ISMIR '05),* pp. 272-279, 2005.

[39] Wang, X., Rosenblum, D. and Wang, Y.: "Context-aware mobile music recommendation for daily activities," *20th ACM Int. Conference on Multimedia (MM '12). ACM,* New York, NY, USA, 99-108, 2012.

[40] Weber, R., Tamborini, R., Westcott-Baker, A., & Kantor, B.: "Theorizing flow and media enjoyment as cognitive synchronization of attentional and reward networks," *Communication Theory,* 19(4), 397–422, 2009.

A LOOK AT THE CLOUD FROM BOTH SIDES NOW: AN ANALYSIS OF CLOUD MUSIC SERVICE USAGE

Jin Ha Lee
University of Washington
jinhalee@uw.edu

Yea-Seul Kim
University of Washington
yeaseul1@uw.edu

Chris Hubbles
University of Washington
chubbles@uw.edu

ABSTRACT

Despite the increasing popularity of cloud-based music services, few studies have examined how users select and utilize these services, how they manage and access their music collections in the cloud, and the issues or challenges they are facing within these services. In this paper, we present findings from an online survey with 198 responses collected from users of commercial cloud music services, exploring their selection criteria, use patterns, perceived limitations, and future predictions. We also investigate differences in these aspects by age and gender. Our results elucidate previously under-studied changes in music consumption, music listening behaviors, and music technology adoption. The findings also provide insights into how to improve the future design of cloud-based music services, and have broader implications for any cloud-based services designed for managing and accessing personal media collections.

1. INTRODUCTION

The last decade has been marked by significant and rapid change in the means by which people store and access music. New technologies, tools, and services have resulted in a plethora of choices for users. Mobile devices are becoming increasingly ubiquitous, and different access methods, including streaming and subscription models, have started to replace the traditional model of music ownership via personal collections [30]. Cloud-based music services are one of the more recently developed consumer options for storing and accessing music, and the use of cloud-based systems in general is expected to increase in the near future. As the popularity of cloud computing grows, a number of studies have been published regarding uses and attitudes of cloud-based systems (e.g., [21]). However, few studies specifically investigate cloud-based music services; many questions regarding the use of those services are virtually unexplored. For instance, what makes people choose cloud-based music services, given numerous streaming choices for accessing music? What works, and what does not work, in existing services, and how can user experiences be improved? What opinions do users hold about cloud-based services, especially regarding the longevity, privacy, and security of such systems? Answering these questions will help elucidate the challenges users are facing in today's complex music access environment, and will inform future music access and organization models.

In this paper, we aim to answer the following research questions: 1) How do people commonly use cloud music services and manage their cloud music collections, and how does streaming usage interact with, support, or supplant cloud music usage?; 2) How do users explain their preferences for particular cloud music services and functionalities?; 3) What do users perceive as limitations of current services, and what kinds of features do users want in a cloud-based music access and management system?; and 4) Are there significant differences in perceptions and usage of cloud music services which correlate to demographic differences, such as age or gender?

This study is part of a larger agenda seeking to empirically ground current understandings of music collecting and information-seeking behavior. The explosive growth of cloud services in the past five years has demonstrated a burgeoning, robust commercial market of products which will benefit from new empirical analyses. This work is critical in an age where technology and society undergo upheavals so frequently that previous models of human activity often prove to be oversimplified or obsolete when applied to new problems. Empirical work in this area has implications for device and software design and development, structuring of metadata, consumer behavior, and music industry planning, in addition to offering contributions to academic theory in multiple disciplines.

2. RELEVANT WORK

Cloud computing has exploded in popularity since the mid-2000s, and scholarly inquiry on the topic has correspondingly increased. User studies of cloud services have found a variety of factors influencing consumer adoption and retention of cloud services, including ease of use and on-demand ubiquity [24, 28], functionality and perceived usefulness [1, 28], accessibility across web-enabled devices [21], and support for collaborative projects [21, 24]. While online music discovery and consumption has also grown dramatically over the course of the nascent 21st century, cloud platforms designed specifically for music listening and storage are still relatively new; for instance, Apple iCloud and Google Play Music, two major competitors in the cloud music marketplace, both launched in 2011. A great deal of speculative and anecdotal literature has arisen around cloud music, including on the cloud's philosophical implications and its potential to disrupt so-

© Jin Ha Lee, Yea-Seul Kim, Chris Hubbles. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jin Ha Lee, Yea-Seul Kim, Chris Hubbles. "A Look at the Cloud from Both Sides Now: An Analysis of Cloud Music Service Usage". 17th International Society for Music Information Retrieval Conference, 2016.

cioeconomic and cultural notions of ownership [4, 22, 30]. However, actual user attitudes toward services and behavior within these services remain underexplored, reflecting a general lack of focus on user experience in MIR studies [27]. Furthermore, cloud services afford and facilitate functions such as transfer of files between devices, automated organization of files and metadata, sharing, and backup, which previously were cumbersome but common user tasks [3]. User behavior thus may have changed significantly, or be in transition, from that described in studies which are only a few years old.

Cloud music services also complement, or compete with, streaming services for listeners' ears. User behavior on streaming services has received more empirical attention as the popularity of platforms like Spotify and Pandora has swelled. Hagen [9] conducted a mixed-methods study to examine playlist-making behavior in music streaming services, finding a heterogeneous set of management and use strategies. Kamalzadeh et al. [14] investigated music listening and management both online and offline, and found that streaming service use was less frequent than offline listening to personal digital music collections. Lee et al. [15, 16] inquired into user needs for music information services and user experience within commercial music platforms, noting increased use of streaming services and exploring opinions about services and features in some depth. Zhang et al. [31] examined user behavior on Spotify through quantitative analysis of use logs, focusing on device switching habits and frequency and periodicity of listening sessions. Liikkanen and Aman [19] conducted a large-scale survey of digital music habits in Finland, finding that online streaming through Spotify and YouTube were predominant. Cesareo and Pastore [5] and Nguyen et al. [23] both executed large-scale surveys of streaming music use to assess consumer willingness to pay for services and streaming's effect on music purchasing and illegal downloading. However, detailed user-centered studies which examine both cloud and streaming services in concert are lacking in the extant literature.

Our study seeks to enrich understandings of online music listeners' needs, desires, attitudes, and behaviors through a large-scale survey of cloud music usage. We also seek to explore whether differences in behaviors and attitudes about cloud and streaming services correlate to demographic differences, particularly age and gender. Music sociology, music psychology, and music information studies researchers have noted gender differences in some aspects of music tastes [8], experiences [18], and listening habits [7, 8], but not others [6, 13, 26]. Technology use can also differ markedly by gender, e.g. in choice of smartphone applications [25], and in adoption and use of mobile phones [12] and social networking services [10]. Comparatively little attention has been paid to whether and how these differences are mirrored in online music service usage; exceptions include Berkers [2], who used Last.FM user data to examine differences in musical taste between genders, and Makkonen et al. [20] and Suki [29], both of whom found gender and age differences in online music purchasing intentions.

3. STUDY DESIGN AND METHOD

This study is a follow-up to an earlier project which investigated current cloud music usage and the future of cloud music practices through semi-structured interviews with 20 adult and 20 teen users [17]. This study seeks to validate findings from the interviews and surface new insights by surveying a larger number of cloud music service users.

The online survey consisted of 24 questions which asked about users' cloud music service usage, cloud music collection management, and general music listening behavior. Our question set was generated after the completion of the interview project, and so our choice of questions was partly informed by our interview findings. Participants were recruited via online venues such as e-mail lists, Facebook groups targeted for students attending the University of Washington, the first author's social network websites, Craigslist, and several online listservs and forums related to music (e.g., ISMIR community listserv, Allaccessplaylists reddit). We also distributed and mailed flyers to 50 physical venues including campus locations, record shops, businesses, libraries, and community centers. Participants were offered an opportunity to enter their names in a raffle to win Amazon.com gift cards.

The survey data included quantitative numerical responses, radio-button and check-all-that-apply multiple choice questions, and free response text boxes. Quantitative data was processed via SPSS and Microsoft Excel. Answers from open-ended questions were qualitatively coded by two coders, employing an iterative process. The codebook from [17] was adopted as an initial framework, and then was slightly expanded and revised after the first round of coding to fully represent the themes in all responses. Afterwards, we adopted a consensus model [11] where two coders compared their coded results and discussed instances where disagreements in code application occurred, aiming to reach a consensus.

Our recruitment methods, both online and real-world, often centered on areas populated by young adults in their twenties and thirties, and while it seems intuitively reasonable that this population would be more likely to patronize cloud services than other demographics, there may be significant cloud-using populations we did not reach. Our outreach efforts occurred mostly within the United States, especially the Puget Sound region, and while we allowed for worldwide access to the survey, the majority of our respondents were Americans. Of our survey respondents, over 70% were male, which may not necessarily be indicative of actual cloud usage patterns.

Despite employing a variety of recruitment tactics and publicizing the survey in several waves, we received a total of 371 responses, of which 198 were complete responses. Since cloud services are a relatively new service industry, we speculate that our recruitment difficulties may be due to a general lack of widespread adoption. Furthermore, many online music consumers are electing to use streaming rather than cloud platforms, making them ineligible for our study.

4. FINDINGS AND DISCUSSION

4.1 Participants' Demographics and Characteristics

The average age of participants was 29.7 (Stdev: 8.5). Most participants (80.8%) were from the United States, with the rest from Canada, the United Kingdom, and 16 other countries. 70.7% of respondents were male, 27.8% were female, and the rest selected 'other'. Participants listened to a wide variety of music as well as spoken-word audio (e.g., comedy, podcasts), with rock, pop, and electronic music being the most preferred genres.

4.2 Usage of Cloud Music Services

Of the three most commonly used cloud music services, Google Play was the predominant service (71.7%), with about a quarter of respondents using each of the other major services (Amazon Cloud, 25.8%; Apple iCloud, 23.7%). These services were primarily accessed by smartphone (91.9%), laptop (75.8%), desktop computer (60.1%), and tablet (51.5%). Devices designed specifically for music listening, such as cloud-enabled home stereo systems (e.g., Sonos) (10.6%) and portable music players (8.1%), were much less common. The average reported length of cloud music service use was 35.5 months (Stdev: 25.8). The frequency of service use tended to be high; 66.2% used them on a daily basis ('almost every day' or 'more than once a day'), and 20.7% on a weekly basis ('about once a week' or 'a few times a week').

Table 1 summarizes how participants reported using cloud music services. Easier access to music which users may or may not own was the primary reason for using services, followed by discovery, preservation, management, and sharing purposes. When they do use cloud services for discovery of new music, 59.6% reported using an automatically-generated playlist or using a cloud radio feature, 41.9% relied on new music suggestions by the service (e.g., advertisements or promotions), and 23.7% took suggestions from friends on the cloud. Approximately one out of four participants (25.3%) did not use cloud services for discovering new music. In the prior study, interviewees reported that they primarily rely on streaming services like Spotify and Pandora for music discovery [17].

Usage of cloud music services	Total (n=198)
To stream music from my collection which I do not have on my music playing devices	171 (86.4%)
To listen to music I do not have in my collection	138 (69.7%)
To discover new music or get recommendations about songs and artists	128 (64.6%)
To hold copies of my digital music files in case my hard drive dies	97 (49.0%)
To transfer digital music files between computers and/or mobile devices	89 (44.9%)
To share music with other people	38 (19.2%)

Table 1. Usage of cloud music services.

4.3 Management of Cloud Music Collections

The median value of the estimated size of participants' music collections was 2,908 songs (1Q: 300, 3Q: 10,000, max: 100,000) or 29.74 GB of disk space (1Q: 5.75, 3Q: 60, max: 2,500). While many participants had sizable collections, organization was not a pressing issue for most of them, as 72.2% stated they relied on automatic organization by the service, compared to 24.2% who manually organize their collections. 56.6% of participants responded that they have music that is not uploaded to the cloud. The reasons varied, from lack of time/resources to issues of limited access (presented in Table 2).

Reasons for having music not uploaded to the cloud	Total (n=112)
I have not had time to add all of them yet	63 (56.3%)
I have enough music in the cloud for my needs right now	40 (35.7%)
They are physical items that are hard to digitize	36 (32.1%)
My cloud storage is limited	30 (26.8%)
I prefer listening to physical items for some music and/or like to have physical copies of things as well	28 (25.0%)
They are physical items which are not readily accessible to me	15 (13.4%)

Table 2. Reasons for having music not uploaded to the cloud.

Although 55.1% of participants responded that they purchase or obtain music from cloud services, few did so frequently, with approximately three out of four participants (72.5%) doing it about once a month or less.

We also asked participants whether they back up their music collection in general, and if so, what kinds of strategies they use. Of all participants, 58.6% responded that they do back up their collection; of those answering yes, 48.3% keep local copies of music files as backup on a secondary storage device, and 11.2% keep copies on a computer. Some participants considered the cloud music services to be their backup (23.3%) or backed up their music in the cloud using another cloud service such as CrashPlan or Google Drive (8.6%). Most of the backup efforts were done in digital file formats; only 3.4% kept physical copies of CDs, vinyl, etc. as backup.

4.4 Music Listening Behavior

YouTube (65.8%), Spotify (57.8%) and Pandora (52.9%) were the most popular streaming services, followed by SoundCloud (40.6%) and Last.FM (23.5%). With the increasing availability of music streaming features offered by cloud and other online music services, we wanted to know how much of the music our participants listen to is actually owned by them (versus access via streaming). As shown in Table 3, the proportions of participants who almost always own or almost always stream the music they listen to were about equal. Approximately one out of four listen to owned music and stream music about the same amount. Overall, the distribution is fairly spread out

across the different categories, although there were slightly more participants who tend to stream more than own music rather than the vice versa.

Ownership vs. Streaming	Total (n=197)
I own almost all the music I listen to	29 (14.7%)
I mostly listen to the music I own, but sometimes stream music I don't own	36 (18.3%)
I listen to music I own and stream about the same amount	52 (26.4%)
I mostly stream music I don't own, but sometimes listen to the music I own	50 (25.4%)
I almost always stream music I don't own	27 (13.7%)
Other	3 (1.5%)

Table 3. Ownership versus Streaming.

89.4% of participants responded that they use playlists. Criteria for generating playlists included personal preference (72.9%), mood (59.9%), genre/style (55.4%), accompanying activity (e.g., working out, partying, traveling) (50.8%), artists (35.6%), and recent acquisition (33.3%). More than half of participants (53.1%) listen to playlists that are automatically generated by the services instead of (or in addition to) creating their own.

4.5 Selection Factors, Perceived Limitations, and Desired Features

We asked respondents how they came to use cloud music services, what they desired from the services, and what kinds of limitations or frustrations had surfaced in their usage of the services. When asked how they initially found services, respondents chose the option 'I sought out cloud services to fit my music listening needs' most frequently from a predetermined list of choices (47.0%). Others had cloud services preinstalled on devices (21.7%), found out from friends or family (21.7%), through advertising (20.7%), or were signed up automatically due to an existing connection with a cloud provider (12.6%). Free-form responses given via the 'other' option indicated that several users discovered their cloud service providers through Internet information sources, such as press coverage or blog posts (11 responses). 64.1% of respondents were paying for cloud music access.

We also asked users which service they preferred of those they had tried and why. 184 users responded to this open-ended question, though 15 of them noted that they only used one service. Qualitative coding of the responses indicated that the most popular reasons were device compatibility (29.9%), ease of upload and size of storage space (23.4%), brand loyalty (19.0%), price (18.5%), and variety and availability of desired music (16.3%). A representative user explained that he chose Google Play Music "because 1) I use an Android phone & tablet, 2) they uploaded my library to their cloud, 3) I jumped on early & have a discounted monthly price." (ID: 103)

51.0% of participants responded that there is something they would like to change about the service they use. From a predetermined bank of answers, users indicated that the most common factors hindering their use of services were lack of good sharing features (40.6%), clumsy or unappealing visual design (30.7%), poor general functionality or bugginess (30.7%), other missing features (26.7%), difficulties with transferring music (22.8%), high cost (11.9%), device compatibility issues (9.9%), and a lack of storage space (7.9%). Free-form responses to this question indicated that song access was also an issue for some users, due to services' incomplete artist libraries or problems uploading certain file formats. Other free-form responses from dissatisfied users related to suboptimal playlist or automated radio features, poor organizational or metadata-curating functionalities, streaming options (such as lack of support for simultaneous streaming from multiple devices), and sharing.

We also asked whether and why users would consider switching to another service. Of the 170 respondents who answered this question, 47.6% indicated they would consider switching, while 34.7% indicated they would not, and 17.6% answered that they might switch or were non-committal. Of those who said they would switch, pricing was by far the most common reason given (43 responses), with artist selection (21) and device compatibility (17) distant runners-up. For those who said they would not switch, the most common thread undergirding responses (11) was a sense of inertia. Moving collections from service to service is time-consuming and cumbersome, making it unappealing to users who have settled in with a cloud provider - especially if the user has bought into a full software/hardware combination (such as Google Play Music and Android devices, or iCloud and Apple devices). For instance, one user noted, "I would not consider switching at this time. It would be a hassle to move my personal music collection to a new service." (ID: 342), and another replied, "Only if I were to switch to another mobile ecosystem." (ID: 197) The need for compatibility across devices and services surfaced repeatedly in qualitative coding of the no-switch responses (9 codes, plus some inertia comments obliquely referenced this); other concerns include artist selection (8), upload/storage needs (7) and price (7). Pricing, artist selection, and device compatibility also surfaced in the replies of the maybe-switch respondents, making these common concerns.

4.6 Differences in Gender and Age

We initially speculated that there might be marked differences in cloud service usage by age based on the fact that cloud services were introduced recently, but our data indicate that age, overall, was a relatively minor factor in explaining cloud service usage variability. We divided the participants into three age groups of approximately equal size (25 and younger, 26-30, 31 and older) and ran chi-square analyses on the responses for most of the survey questions (excluding open-ended questions) to identify statistically significant differences. Significant differences between age groups were observed in questions regarding music purchase and paying behavior, as well as in choice of device for accessing cloud music services. Participants who were 31 or older were more likely to pay to use cloud services (X^2=11.34, df=2, p=0.003), though younger people more frequently purchased or obtained music from cloud services (X^2=21.06, df=8, p=0.006) (cf. Makkonen's [20] findings regarding age and willingness to pay for music downloads). Older par-

ticipants also tended to access cloud music via desktop computers (X^2=12.76, df=2, p=0.002) more than younger participants. Younger participants were more likely to use YouTube for streaming (X^2=7.17, df=2, p=0.028). Notably, no significant difference was observed by age for the question asking about listening to owned music versus streaming unowned music, challenging presumptions that younger listeners are less concerned with owning music.

Our survey results indicated that, rather than age, gender seemed to play a larger role in cloud music behavioral differences. Almost half of the respondents reported using cloud services more than once a day, but men tended toward daily usage (90.7% of male users reported using cloud services 'a few times a week' or more), while women's usage was much more evenly distributed between daily ('more than once a day' + 'almost every day': 36.4%), weekly ('a few times a week' + 'about once a week': 36.4%), or monthly ('2 or 3 times a month' + 'once a month or less': 27.3%) access and usage (X^2=42.13, df=5, p=0.000).

In general, we noted a trend across multiple questions indicating that women tended to listen to music within their collections and were less likely to listen to music they did not already know than men were. Nearly half of female participants noted that they 'mostly' (20.0%) or 'almost always' (27.3%) listened to music they owned, whereas almost half of male participants 'mostly' (30.7%) or 'almost always' (15.0%) streamed music (X^2=15.05, df=5, p=0.010). Women were far less likely to report that they used the services for listening to music they did not have in their collections (47.3% for women [W]; 79.3% for men [M]; X^2=19.37, df=1, p=0.000), and made far less use of cloud recommendation and discovery functions (36.4% for W; 77.1% for M; X^2=29.12, df=1, p=0.000), such as new music suggestions (29.1% for W; 47.1% for M; X^2=5.28, df=1, p=0.02), automatically generated playlists (38.2% for W; 69.3% for M; X^2=15.99, df=1, p=0.000), and suggestions from friends (12.7% for W; 28.6% for M; X^2=5.42, df=1, p=0.020), than men did. 38.2% of female respondents noted that they did not use cloud services for music discovery at all, compared with 19.3% of men (X^2=7.60, df=1, p=0.006). One possible caveat here is that women reported much higher usage of the Pandora streaming service alongside cloud services (70.4% for W; 45.4% for M; X^2=9.56, df=1, p=0.002). Pandora, an Internet radio service with personalization features, does not allow for collection building or search access to specific songs, and so may be a route to music discovery for some female users. However, it is possible that the heavier usage of Pandora among women may simply be an issue of convenience (Pandora requires no upkeep or maintenance once a station is chosen, unless the user decides to vote up or down songs she likes or dislikes). Women may also be using Pandora's playlists for listening to similar songs (generated based on already familiar and preferred songs/artists) rather than seeking out channels playing new and unfamiliar music, or for listening to more mainstream genres, which they prefer more than men, according to Berkers [2]. Lastly, Pandora's prominence among female users could merely be indicative of targeted advertising; it is mirrored in the site's general user demographics.[1]

Women reported using cloud services to purchase music more than men did (67.3% for W; 50.0% for M; X^2=4.76, df=1, p=0.029), but were much less likely to pay for the cloud service as a whole than men were (29.1% for W; 78.6% for M; X^2=42.28, df=1, p=0.000), both confirming and complicating Makkonen's [20] finding that women express a higher willingness to pay for music albums and tracks. When asked how they initially found out about cloud music services, more males chose the options 'I sought out cloud services to fit my music listening needs' (32.7% for W; 53.6% for M; X^2=6.877, df=1, p=0.009) or 'through an advertisement' (9.1% for W; 24.3% for M; X^2=5.70, df=1, p=0.017), while women were more likely to choose the responses 'the service was preinstalled on a device I obtained' (45.5% for W; 12.9% for M; X^2=24.41, df=1, p=0.000) or 'a company automatically signed me up for a cloud music service' (30.9% for W; 5.0% for M; X^2=24.56, df=1, p=0.000). Perhaps not coincidentally, men were far more likely than women to report using Google Play Music though many women also used this service (45.5% for W; 82.9% for M; X^2=27.59, df=1, p=0.000), while women were much more likely to use Apple iCloud and very few men were iCloud users (54.5% for W; 12.1% for M; X^2=38.81, df=1, p=0.000). Apple tends to focus on integration of software and hardware, and frequently bundles services together.

This seems to indicate that women are exercising less overt consumer choice in selecting a cloud provider, which may have implications for service fit and user satisfaction. For instance, women were much more likely than men to use the services for transfer between devices (70.9% for W; 34.3% for M; X^2=21.43, df=1, p=0.000), and they were more likely to report problems with transferring files (47.6% for W; 15.4% for M; X^2=9.95, df=1, p=0.002) and device compatibility issues (23.8% for W; 6.4% for M; X^2=5.52, df=1, p=0.019) when asked about service deficiencies. Suki [29] reports a similar tendency of men having a higher level of perceived ease of use than women when using online music. Women have more music not uploaded to the cloud (76.4% for W; 49.3% for M; X^2=11.50, df=1, p=0.001) which may reflect that they have enough music in the cloud for their needs now (45.2% for W; 30.4% for M, although not significant) and that they prefer to listen to physical copies (35.7% for W; 18.8% for M; X^2=3.941, df=1, p=0.047).

4.7 Thoughts on the Trend of Moving to the Cloud

Our survey concluded with an open-ended question asking respondents to express other thoughts or opinions they had about cloud computing and cloud music storage. 98 users responded with statements of length varying from a single sentence fragment to several paragraphs. These responses were qualitatively coded and examined for common patterns using a consensus code strategy [11]. We found that the codebook developed for our interview project [17] was useful as a starting point, and only a few codes were added to this preexisting frame-

[1] Alexa.com reports that Pandora's userbase skews strongly female. http://www.alexa.com/siteinfo/pandora.com

work during coding iterations. The most common topic which surfaced in these responses was the relationship between cloud and streaming music platforms and their relative benefits and drawbacks. Alongside this was an abiding concern over issues of ownership and access, present in nearly a quarter of responses. Users expressed keen and sometimes profuse opinions about ownership and access modes of listening, just as the interviewees did in our project's first phase [17] - but without explicit prompting, and with minimal addressing of the topic in earlier survey questions (only one question, discussed in Section 4.4, indirectly references this issue). As in [17], participants expressed a variety of positions: one uneasy user noted, "The entire system of 'owning music' is nearly obsolete. The legal as well as social ramifications of identity ties to cultural objects to which someone else controls all access is little understood and downright frightening" (ID: 36), and another cloud skeptic stated, "It's scary to think of everything being online without a physical copy anywhere. I still purchase CDs and import them to my online service because I enjoy having a real CD, but appreciate the probabilities of cloud streaming." (ID: 110) Still others saw cloud-based access models as an nigh-unstoppable new wave: "These [record] labels need to wake up the internet/cloud is not a fad it is the future[. S]ure it will be improved upon but I have not bought a physical album in years and eventually no one will." (ID: 311) Once again, age was not a reliable predictor of opinion on ownership/access matters; many under-26 users favored owning files, and several over-30 users favored access-only streaming systems. Concerns over service cost (22 responses), praise or circumspection regarding service convenience (20), opinions about artist and genre availability (15), and fears or experiences of network and data issues (20) and storage caps (15) also factored prominently into responses to this call for opinions.

One topic which was more prominent in our survey than the interviews was artist royalties, perhaps influenced by recent news coverage of court cases involving streaming royalty payments, as well as the weighing-in of high-profile musicians (such as country/pop superstar Taylor Swift) on the subject. Some wrote approvingly of service handling of royalty payments, such as the user who wrote, "I like the fact that the music is now more available to more people and that it can be accessed more globally while still generating revenue for the artist." (ID: 101) Others had more ambivalent reactions: "While as a musician I recognize the damage st[r]eaming services [have done] to the industry, as a listener the convenience is absolutely incredible and has introduced me to so much new music." (ID: 192) Also more prominent in survey responses than in the interviews were comments regarding audio quality of services; one user replied, "I would never consider going all-streaming, unless I (and the infrastructure) were able to do this with full-quality uncompressed audio... I'm interested in services like PONO and TIDAL with 'high-quality' audio streaming, but, they are too expensive for me to opt in." (ID: 103)

5. CONCLUSION AND FUTURE WORK

Our survey results show that cloud music services are primarily used to improve music access by overcoming limitations imposed by device storage or lack of ownership. While listening from participants' own music collections was the top usage of cloud services, streaming music they do not own was important as well. This seems to signal a desire for merged systems with both cloud and streaming features. The services are also used for music discovery and management, though less so for sharing music. Exploring and implementing better ways to share listening experiences may help improve users' experiences with cloud services. Collection-building and streaming approaches divide online music usage, although there is a slight preference toward streaming.

Approximately half of participants reported choosing services to fit their needs, although a substantial number were influenced by preinstalled options, word of mouth, and advertising. Major contributing factors in user service choice included device compatibility, ease of upload, storage space, brand loyalty, price, and music availability. Over half of the participants indicated the desire to change something about the services they use. Again, the lack of good sharing features was the most commonly mentioned factor, followed by dissatisfaction regarding the design and functioning of the service. Difficulty transferring music was also mentioned by about a quarter of participants. Nearly half of respondents indicated they would consider switching to another service based on price, artist selection, and device compatibility.

Differences regarding use of cloud music services were much more prominent by gender rather than age. Women reported listening to music they owned more than men, sought out new music less than men, paid for services less often, and asserted less consumer choice in selecting services than men did. This warrants future investigation of the underlying reasons for these differences, and also suggests opportunities for developing music services tailored to gender-specific usage.

In future work, we plan to continue our investigation of music users, focusing on two aspects: 1) the meaning of personal collections in an increasingly streaming-dominated environment, and 2) investigation of reasons for the differences observed in music selection, listening, and sharing between genders.

6. ACKNOWLEDGEMENTS

The authors extend special thanks to Lara Aase and Rachel Wishkoski for their contributions to survey design, and Rebecca Fronczak for assisting in recruiting survey participants. This research is supported by the University of Washington Office of Research.

7. REFERENCES

[1] P. Ambrose and A. Chiravuri: An empirical investigation of cloud computing for personal use. *MWAIS 2010 Proceedings*, Paper 24, 2010.

[2] P. Berkers: Gendered scrobbling: Listening behavior of young adults on Last.fm. *Interactions: Studies in Communication & Culture*, 2(3), pp. 279-296, 2010.

[3] J. Brinegar and R. Capra: Managing music across multiple devices and computers. *Proceedings of the iConference*, pp. 489-495, 2011.

[4] P. Burkart: Music in the cloud and the digital sublime. *Popular Music and Society*, 37(4), pp. 393-407, 2014.

[5] L. Cesareo and A. Pastore: Consumers' attitude and behavior towards online music piracy and subscription-based services. *J. Consumer Marketing,* 31(6/7), pp. 515-525, 2014.

[6] T. Chamorro-Premuzic and A. Furnham: Personality and music: Can traits explain how people use music in everyday life? *Brit. J. Psychol.*, 98, pp. 175-185, 2007.

[7] T. Chamorro-Premuzic, V. Swami and B. Cermakova: Individual differences in music consumption are predicted by uses of music and age rather than emotional intelligence, neuroticism, extraversion or openness. *Psychology of Music*, 40(3), pp. 285-300, 2012.

[8] T. DeNora: *Music in everyday life*. Cambridge University Press, Cambridge, UK, 2000.

[9] A. Hagen: The playlist experience: Personal playlists in music streaming services. *Popular Music and Society*, 38(5), pp. 625-645, 2015.

[10] E. Hargittai: Whose space? Differences among users and non-users of social network sites. *J. Comp.-Mediated Comm.*, 13, pp. 276-297, 2008.

[11] C. Hill et al.: Consensual qualitative research: an update. *J. Couns. Psych.* 52(2), pp. 196-205, 2005.

[12] R. Junco, D. Merson and D. Salter: The effect of gender, ethnicity and income on college students' use of communication technologies. *Cyberpsychology, Behavior, and Social Networking*, 13(6), pp. 619-627, 2010.

[13] P. Juslin et al.: An experience sampling study of emotional reactions to music: Listener, music, and situation. *Emotion*, 8(5), pp. 668-683, 2008.

[14] M. Kamalzadeh, D. Baur and T. Möller: A survey on music listening and management behaviours. *Proceedings of the ISMIR,* pp. 373-378, 2012.

[15] J. H. Lee and N. Waterman: Understanding user requirements for music information services. *Proceedings of the ISMIR,* pp. 253-258, 2012.

[16] J. H. Lee and R. Price: User experience with commercial music services: an empirical exploration. *JASIST*, 2015. DOI: 10.1002/asi.23433

[17] J. H. Lee, R. Wishkoski, L. Aase, P. Meas and C. Hubbles: Understanding users of cloud music services: selection factors, management and access behavior, and perceptions. *JASIST*, 2016. In press.

[18] M. Lesaffre, L. De Voogt and M. Leman: How potential users of music search and retrieval systems describe the semantic quality of music. *JASIST*, 59(5), pp. 695-707, 2008.

[19] L. Liikkanen and P. Aman: Shuffling services: Current trends in interacting with digital music. *Interacting with Comp.*, 2015. http://iwc.oxfordjournals.org/content/early/2015/03/27/iwc.iwv004.full

[20] M. Makkonen, V. Halttunen and L. Frank: The effects of gender, age, and income on the willingness to pay for music downloads. *Bled eConference Proceedings*, paper 39, 2011.

[21] C. Marshall and J. Tang: That syncing feeling: Early user experiences with the cloud. *Proceedings of DIS*, pp. 544-553, 2012.

[22] J. Morris: Sounds in the cloud: Cloud computing and the digital music commodity. *First Monday, 16*(5), 2011. http://firstmonday.org/article/view/3391/2917

[23] G. Nguyen, D. Dejean and S. Moreau: On the complementarity between online and offline music consumption: The case of free streaming. *J. Cultural Economics, 38*(4), pp. 315-330, 2014.

[24] S. Park and S. Ryoo: An empirical investigation of end-users' switching toward cloud computing: A two factor theory perspective. *Comp. in Human Behavior*, 29(1), pp. 160-170, 2013.

[25] K. Purcell, R. Entner and N. Henderson: The rise of apps culture. Pew Research, 2010. http://www.pewinternet.org/2010/09/14/the-rise-of-apps-culture/

[26] P. Rentfrow and S. Gosling: The do re mi's of everyday life: The structure and personality correlates of music preferences. *J. Psychology and Social Psychology*, 84(6), pp. 1236-1256, 2003.

[27] M. Schedl and A. Flexer: Putting the user in the center of music information retrieval. *Proceedings of the ISMIR,* pp. 385-390, 2012.

[28] V. Stantchev et al.: Learning management systems and cloud file hosting services: A study on students' acceptance. *Comp. in Human Behavior*, 31, pp. 612-619, 2014.

[29] N. Suki: Gender, age, and education: Do they really moderate online music acceptance? *Communications of the IBIMA*, 2011.

[30] P. Wikström: *Music industry: Music in the cloud,* 2nd edition. Polity Press, Cambridge, UK, 2013.

[31] B. Zhang et al.: Understanding user behavior in Spotify. *Proceedings of IEEE INFOCOM*, pp. 220-224, 2013.

Poster Session 2

A HIERARCHICAL BAYESIAN MODEL OF CHORDS, PITCHES, AND SPECTROGRAMS FOR MULTIPITCH ANALYSIS

Yuta Ojima[1] **Eita Nakamura**[1] **Katsutoshi Itoyama**[1] **Kazuyoshi Yoshii**[1]
[1] Graduate School of Informatics, Kyoto University , Japan
{ojima, enakamura}@sap.ist.i.kyoto-u.ac.jp, {itoyama, yoshii}@kuis.kyoto-u.ac.jp

ABSTRACT

This paper presents a statistical multipitch analyzer that can simultaneously estimate pitches and chords (typical pitch combinations) from music audio signals in an unsupervised manner. A popular approach to multipitch analysis is to perform nonnegative matrix factorization (NMF) for estimating the temporal activations of semitone-level pitches and then execute thresholding for making a piano-roll representation. The major problems of this cascading approach are that an optimal threshold is hard to determine for each musical piece and that musically inappropriate pitch combinations are allowed to appear. To solve these problems, we propose a probabilistic generative model that fuses an acoustic model (NMF) for a music spectrogram with a language model (hidden Markov model; HMM) for pitch locations in a hierarchical Bayesian manner. More specifically, binary variables indicating the existences of pitches are introduced into the framework of NMF. The latent grammatical structures of those variables are regulated by an HMM that encodes chord progressions and pitch co-occurrences (chord components). Given a music spectrogram, all the latent variables (pitches and chords) are estimated jointly by using Gibbs sampling. The experimental results showed the great potential of the proposed method for unified music transcription and grammar induction.

1. INTRODUCTION

The goal of automatic music transcription is to estimate the *pitches*, *onsets*, and *durations* of musical notes contained in polyphonic music audio signals. These estimated values must be directly linked with the elements of music scores. More specifically, in this paper, a pitch means a discrete fundamental frequency (F0) quantized in a semitone level, an onset means a discrete time point quantized on a regular grid (*e.g.*, eighth-note-level grid), and a duration means a discrete note value (integer multiple of the grid interval).

In this study we tackle multipitch estimation (subtask of automatic music transcription) that aims to make a binary piano-roll representation from a music audio signal, where only the existences of pitches are estimated at each frame. A popular approach to this task is to use non-negative matrix factorization (NMF) [1–7]. It approximates the magnitude spectrogram of an observed mixture signal as the product of a basis matrix (a set of basis spectra corresponding to different pitches) and an activation matrix (a set of temporal activations corresponding to those pitches). The existence of each pitch is then determined by executing thresholding or Viterbi decoding based a hidden Markov model (HMM) for the estimated activations [7, 8].

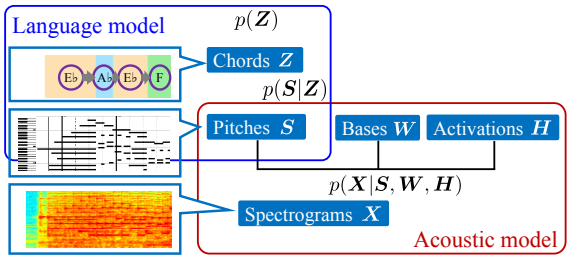


Figure 1. Overview of the proposed model consisting of language and acoustic models that are linked through binary variables $\boldsymbol{S}$ representing the existences of pitches.

This NMF-based cascading approach, however, has two major problems. First, it is hard to optimize a threshold for each musical piece. Second, the estimated results are allowed to be musically inappropriate because the relationships between different pitches are not taken into account. In fact, music has simultaneous and temporal structures; certain kinds of pitches (*e.g.*, C, G, and E) tend to simultaneously occur to form chords (*e.g.*, C major), which vary over time to form typical progressions. If such structural information is unavailable for multipitch analysis, we need to tackle the chicken-and-egg problem that chords are determined by pitch combinations, and vice versa.

To solve these problems, we propose a statistical method that can discover chords and pitches from music audio signals in an unsupervised manner while taking into account their interdependence (Fig.1). More specifically, we formulate a hierarchical Bayesian model that represents the generative process of an observed music spectrogram by unifying an *acoustic model* (probabilistic model underlying NMF) that represents how the spectrogram is generated from pitches and a *language model* (HMM) that represents how the pitches are generated from chords. A key feature of the unified model is that binary variables indicating the existences of pitches are introduced into the framework of NMF. This enables the HMM to represent both chord

© Yuta Ojima, Eita Nakamura, Katsutoshi Itoyama, Kazuyoshi Yoshii. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Yuta Ojima, Eita Nakamura, Katsutoshi Itoyama, Kazuyoshi Yoshii. "A Hierarchical Bayesian Model of Chords, Pitches, and Spectrograms for Multipitch Analysis", 17th International Society for Music Information Retrieval Conference, 2016.

transitions and pitch combinations using only discrete variables forming a piano-roll representation with chord labels. Given a music spectrogram, all the latent variables (pitches and chords) are estimated jointly by using Gibbs sampling.

The major contribution of this study is to realize unsupervised induction of musical grammars from music audio signals by unifying acoustic and language models. This approach is formally similar to, but essentially different from that to automatic speech recognition (ASR) because both the models are jointly learned in an unsupervised manner. In addition, our unified model has a three-level hierarchy (chord–pitch–spectrogram) while ASR is usually based on a two-level hierarchy (word–spectrogram). The additional layer is introduced by using an HMM instead of a Markov model (n-gram model) as a language model.

2. RELATED WORK

This section reviews related work on multipitch estimation (acoustic modeling) and on music theory implementation and musical grammar induction (language modeling).

2.1 Acoustic Modeling

The major approach to music signal analysis is to use nonnegative matrix factorization (NMF) [1–6, 9]. Cemgil *et al.* [9] developed a Bayesian inference scheme for NMF, which enabled the introduction of various hierarchical prior structures. Hoffman *et al.* [3] proposed a Bayesian nonparametric extension of NMF called gamma process NMF for estimating the number of bases. Liang *et al.* [6] proposed beta process NMF, in which binary variables are introduced to indicate the needs of individual bases at each frame. Another extension is source-filter NMF [4], which further decomposes the bases into sources (corresponding to pitches) and filters (corresponding to timbres).

2.2 Language Modeling

The implementation and estimation of music theory behind musical pieces are composed have been studied [10–12]. For example, some attempts have been made to computationally formulate the Generative Theory of Tonal Music (GTTM) [13], which represents the multiple aspects of music in a single framework. Hamanaka *et al.* [10] reformalized GTTM through a computational implementation and developed a method for automatically estimating a tree that represents the structure of music, called a time-span tree. Nakamura *et al.* [11] also re-formalized GTTM using a probabilistic context-free grammar model and proposed inference algorithms. These methods enabled automatic analysis of music. On the other hand, induction of music theory in an unsupervised manner has also been studied. Hu *et al.* [12] extended latent Dirichlet allocation and proposed a method for determining the key of a musical piece from symbolic and audio music based on the fact that the likelihood of appearance of each note tends to be similar among musical pieces in the same key. This method enabled the distribution of notes in a certain key to be obtained without using labeled training data.

Assuming that the concept of chords is a kind of music grammar, statistical methods of supervised chord recognition [14–17] are deeply related with unsupervised musical grammar induction. Rocher *et al.* [14] attempted chord recognition from symbolic music by constructing a directed graph of possible chords and then calculating the optimal path. Sheh *et al.* [15] used acoustic features called chroma vectors to estimate chords from music audio signals. They constructed an HMM whose latent variables are chord labels and whose observations are chroma vectors. Maruo *et al.* [16] proposed a method that uses NMF for extracting reliable chroma features. Since these methods need labeled training data, the concept of chords is required in advance. Approaches to make use of a sequence of chords in estimating pitches has also been proposed [18, 19]. This method estimates chord progressions and multiple pitches simultaneously by using a dynamic Bayesian network and shows better performance even with a simple acoustic model. Recent works employ recurrent neural networks as a language model to describe the relations between pitch combinations [20, 21].

3. PROPOSED METHOD

This section explains the proposed method of multipitch analysis that simultaneously estimates pitches and chords at the frame level from music audio signals. Our approach is to formulate a probabilistic generative model for observed music spectrograms and then solve the "inverse" problem, *i.e.*, given a music spectrogram, estimate unknown random variables involved in the model. The proposed model has a hierarchical structure consisting of acoustic and language models that are connected through a piano roll, *i.e.*, a set of binary variables indicating the existences of pitches (Fig. 1). The acoustic model represents the generative process of a music spectrogram from the piano roll, basis spectra, and temporal activations of individual pitches. The language model represents the generative process of chord progressions and pitch locations from chords.

3.1 Problem Specification

The goal of multipitch estimation is to make a piano roll from a music audio signal. Let $\boldsymbol{X} \in \mathbb{R}_+^{F \times T}$ be the magnitude spectrogram of a target signal, where F is the number of frequency bins and T is the number of time frames. We aim to convert $\boldsymbol{X}$ into a piano roll $\boldsymbol{S} \in \{0,1\}^{K \times T}$, which represents the existences of K kinds of pitches over T frames. In addition, we attempt to estimate a sequence of chords $\boldsymbol{Z} = \{z_t\}_{t=1}^{T}$.

3.2 Acoustic Modeling

The acoustic model is formulated in a similar way to beta-process NMF having binary masks [6] (Fig. 2). The given spectrogram $\boldsymbol{X} \in \mathbb{R}_+^{F \times T}$ is factorized into bases $\boldsymbol{W} \in \mathbb{R}_+^{F \times K}$, activations $\boldsymbol{H} \in \mathbb{R}_+^{K \times T}$, and binary variables $\boldsymbol{S} \in \{0,1\}^{K \times T}$ as follows:

$$X_{ft} | \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{S} \sim \mathrm{Poisson}\left(\sum_{k=1}^{K} W_{fk} H_{kt} S_{kt}\right), \quad (1)$$

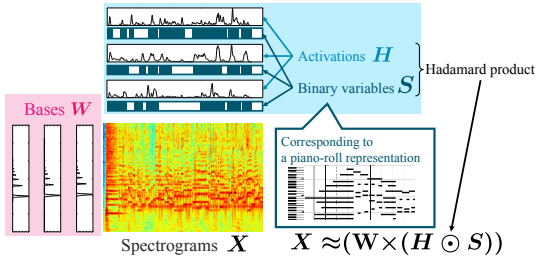


Figure 2. The overview of the acoustic model based on a variant of NMF having binary variables (masks).

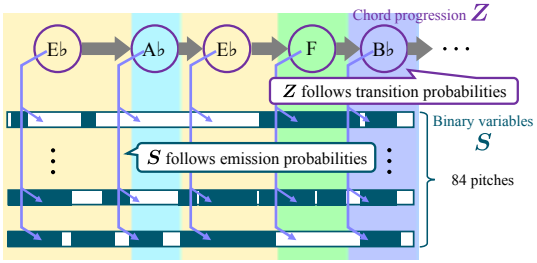


Figure 3. The overview of the language model based on an HMM that stochastically emits binary variables.

where $\{W_{fk}\}_{f=1}^{F}$ is the k-th basis spectrum, H_{kt} is the volume of basis k at frame t, and S_{kt} is a binary variable indicating whether or not basis k is used at frame t.

A set of basis spectra $\boldsymbol{W}$ is divided into two parts: harmonic spectra and noise spectra. In this study we prepare K_h harmonic basis spectra corresponding to K_h different pitches and one noise basis spectrum ($K = K_h + 1$). Assuming that the harmonic structures of the same instrument have the shift-invariant relationships, the harmonic part of $\boldsymbol{W}$ are given by

$$\{W_{fk}\}_{f=1}^{F} = \mathrm{shift}\left(\{W_f^{\mathrm{h}}\}_{f=1}^{F},\ \zeta(k-1)\right), \qquad (2)$$

for $k = 1, \ldots K_h$, where $\{W_f^{\mathrm{h}}\}_{f=1}^{F}$ is a harmonic template structure common to harmonic basis spectra used for NMF, $\mathrm{shift}\,(\boldsymbol{x}, a)$ is an operator that shifts $\boldsymbol{x} = [x_1, \ldots, x_n]^T$ to $[0, \ldots, 0, x_1, \ldots, x_{n-a}]^T$, and ζ is the number of frequency bins corresponding to the semitone interval.

We put two kinds of priors on the harmonic template spectrum $\{W_f^{\mathrm{h}}\}_{f=1}^{F}$ and a noise basis spectrum $\{W_f^{\mathrm{n}}\}_{f=1}^{F}$. To make the harmonic spectrum sparse, we put a gamma prior on $\{W_f^{\mathrm{h}}\}_{f=1}^{F}$ as follows:

$$W_f^{\mathrm{h}} \sim \mathcal{G}\left(a^{\mathrm{h}}, b^{\mathrm{h}}\right) \qquad (3)$$

where a^{h} and b^{h} are hyperparameters. On the other hand, we put an inverse-gamma chain prior [22] on $\{W_f^{\mathrm{n}}\}_{f=1}^{F}$ to induce the spectral smoothness as follows:

$$G_f^W | W_{f-1}^{\mathrm{n}} \sim \mathcal{IG}\left(\eta^W, \frac{\eta^W}{W_{f-1}^{\mathrm{n}}}\right),$$
$$W_f^{\mathrm{n}} | G_f^W \sim \mathcal{IG}\left(\eta^W, \frac{\eta^W}{G_f^W}\right), \qquad (4)$$

where η^W is a hyperparameter that determines the strength of smoothness and G_f^W is an auxiliary variable that induces positive correlation between W_{f-1}^{n} and W_f^{n}.

A set of activations $\boldsymbol{H}$ is represented in the same way as $\boldsymbol{W}$. If H_{kt} takes almost zero, S_{kt} has no impact on NMF. This allows S_{kt} to take one (the corresponding pitch is judged to be activated) even though the activation H_{kt} is almost zero. We can avoid this problem by putting an inverse-gamma prior for H_{kt} to induce non-zero values. To induce the temporal smoothness in addition, we put the following inverse-gamma chain prior on $\boldsymbol{H}$:

$$G_{kt}^H | H_{k(t-1)} \sim \mathcal{IG}\left(\eta_H, \frac{\eta_H}{H_{k(t-1)}}\right),$$
$$H_{kt} | G_{kt}^H \sim \mathcal{IG}\left(\eta_H, \frac{\eta_H}{G_{kt}^H}\right), \qquad (5)$$

where η_H is a hyperparameter that determines the strength of smoothness and G_{kt}^H is an auxiliary variable that induces positive correlation between $H_{k(t-1)}$ and H_{kt}.

3.3 Language Modeling

The language model is an HMM that has a Markov chain of latent variables $\boldsymbol{Z} = \{z_1, \ldots, z_T\}$ ($z_t \in \{1, \ldots, I\}$) and emits binary variables $\boldsymbol{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T\}$ ($\boldsymbol{s}_t \in \{0,1\}^{K_h}$), where I represents the number of states (chords) and K_h represents the number of possible pitches. Note that $\boldsymbol{S}$ is actually a set of latent variables in the proposed unified model. The HMM is defined as:

$$z_1 | \boldsymbol{\phi} \sim \mathrm{Categorial}(\boldsymbol{\phi}), \qquad (6)$$
$$z_t | z_{t-1}, \boldsymbol{\psi}_{z_{t-1}} \sim \mathrm{Categorical}(\boldsymbol{\psi}_{z_{t-1}}), \qquad (7)$$
$$S_{kt} | z_t, \pi_{z_t k} \sim \mathrm{Bernoulli}(\pi_{z_t k}) \qquad (8)$$

where $\boldsymbol{\psi}_i \in \mathbb{R}^I$ is a set of transition probabilities of chord i, $\boldsymbol{\phi} \in \mathbb{R}^I$ is a set of initial probabilities, and $\pi_{z_t k}$ indicates the probability that the k-th pitch is emitted under a chord z_t, We put conjugate priors on these parameters as:

$$\boldsymbol{\psi}_i \sim \mathrm{Dir}(\mathbf{1}_I), \quad \boldsymbol{\phi} \sim \mathrm{Dir}(\mathbf{1}_I), \quad \pi_{z_t k} \sim \mathrm{Beta}(e, f), \qquad (9)$$

where $\mathbf{1}_I$ is the I-dimensional all-one vector and e and f are hyperparameters.

In practice, we represent only the emission probabilities of 12 pitch classes (C, C#, . . ., B) in one octave. Those probabilities are copied and pasted to recover the emission probabilities of K_h kinds of pitches. In addition, the emission probabilities $\{\pi_{ik}\}_{k=1}^{K_h}$ of chord i are forced to have circular-shifting relationships with those of other chords of the same type. In this paper, we consider only major and minor chords as chord types ($I = 2 \times 12$) for simplicity.

3.4 Posterior Inference

Given the observed data $\boldsymbol{X}$, our goal is to calculate the posterior distribution $p(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{S}, \boldsymbol{z}, \boldsymbol{\pi}, \boldsymbol{\psi} | \boldsymbol{X})$. Since analytic calculation is intractable, we use Markov chain Monte Carlo (MCMC) methods as in [23]. Since the acoustic and language models share only the binary variables, each model can be updated independently when the binary variables are given. These models and binary variables are iteratively sampled. Finally, the latent variables (chord progressions) of the language model are estimated by using the Viterbi algorithm and the binary variables (pitch locations) are determined by using parameters having the maximum likelihood.

3.4.1 Sampling Binary Variables

The binary variables $\boldsymbol{S}$ are sampled from a posterior distribution that is calculated by integrating the acoustic model

as a likelihood function and the language model as a prior distribution according the Bayes' rule. Note that as shown in Fig. 1, the binary variables $\boldsymbol{S}$ are involved in both acoustic and language models (*i.e.*, the probability of each pitch being used is determined by a chord, and whether or not each pitch is used affects the reconstructed spectrogram). The conditional posterior distribution of S_{kt} is given by

$$S_{kt} \sim \text{ Bernoulli}\left(\frac{P_1}{P_1+P_0}\right), \tag{10}$$

where P_1 and P_0 are given by

$$\begin{aligned} P_1 &= p(S_{kt}=1|S_{\neg k,t}, \boldsymbol{x}_t, \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{\pi}, \boldsymbol{z}, \alpha) \qquad (11)\\ &\propto \pi_{z_k}^{\alpha}\prod_f \left(\hat{X}_{ft}^{\neg k} + W_{fk}H_{kt}\right)^{X_{ft}} \exp\{-W_{fk}H_{kt}\},\\ P_0 &= p(S_{kt}=0|S_{\neg k,t}, \boldsymbol{x}_t, \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{\pi}, \alpha)\\ &\propto (1-\pi_{z_k})^{\alpha}\prod_f \left(\hat{X}_{ft}^{\neg k}\right)^{X_{ft}}, \qquad (12)\end{aligned}$$

where $\hat{X}_{ft}^{\neg k} \equiv \sum_{l\neq k} W_{fl}H_{lt}S_{lt}$ denotes the magnitude at frame t reconstructed without using the k-th basis and α is a parameter that determines the weight of the language model relative to that of the acoustic model. Such a weighting factor is also needed in ASR. If α is not equal to one, Gibbs sampling cannot be used because the normalization factor cannot be analytically calculated. Instead, the Metropolis-Hastings (MH) algorithm is used by regarding Eq. (10) is used as a proposal distribution

3.4.2 Updating the Acoustic Model

The parameters of the acoustic model $\boldsymbol{W}^{\mathrm{h}}$, $\boldsymbol{W}^{\mathrm{n}}$, and $\boldsymbol{H}$ can be sampled using Gibbs sampling. These parameters are categorized into those having gamma priors ($\boldsymbol{W}^{\mathrm{h}}$) and those inverse-gamma chain priors ($\boldsymbol{W}^{\mathrm{n}}$ and $\boldsymbol{H}$).

Using the Bayes' rule, the conditional posterior distribution of $\boldsymbol{W}^{\mathrm{h}}$ is given by

$$W_{fk}^{\mathrm{h}} \sim \mathcal{G}\left(\textstyle\sum_t X_{ft}\lambda_{ftk} + a^{\mathrm{h}},\ \sum_t H_{kt}S_{kt} + b^{\mathrm{h}}\right), \tag{13}$$

where λ_{ftk} is a normalized auxiliary variable that is calculated with the latest sampled variables $\hat{\boldsymbol{W}}$, $\hat{\boldsymbol{H}}$, and $\hat{\boldsymbol{S}}$, as:

$$\lambda_{ftk} = \frac{\hat{W}_{fk}\hat{H}_{kt}\hat{S_{kt}}}{\sum_l \hat{W}_{fl}\hat{H}_{lt}\hat{S_{lt}}}. \tag{14}$$

The other parameters are sampled through auxiliary variables. Since $\boldsymbol{H}$ and $\boldsymbol{G}^H$ are interdependent in Eq. (5) and cannot be sampled jointly, $\boldsymbol{G}^H$ and $\boldsymbol{H}$ are sampled alternately. The conditional posterior of $\boldsymbol{G}^H$ is given by

$$G_{kt}^H \sim \mathcal{IG}\left(2\eta_H, \eta_H\left(\frac{1}{H_{kt}} + \frac{1}{H_{k(t-1)}}\right)\right). \tag{15}$$

Similarly, the conditional posteriors of $\boldsymbol{H}$, $\boldsymbol{G}^W$, and $\boldsymbol{W}^{\mathrm{n}}$ are given by

$$H_{kt} \sim \mathcal{IG}\left(2\eta_H, \eta_H\left(\frac{1}{G_{k(t+1)}^H} + \frac{1}{G_{kt}^H}\right)\right), \tag{16}$$

$$G_f^W \sim \mathcal{IG}\left(2\eta_W, \eta_W\left(\frac{1}{W_f^{\mathrm{n}}} + \frac{1}{W_{f-1}^{\mathrm{n}}}\right)\right), \tag{17}$$

$$W_f^{\mathrm{n}} \sim \mathcal{IG}\left(2\eta_W, \eta_W\left(\frac{1}{G_{f+1}^W} + \frac{1}{G_f^W}\right)\right), \tag{18}$$

if the observation $\boldsymbol{X}$ is not taken into account. Using the Bayes' rule and Jensen's inequality as in Eq. (13) and regarding Eq. (16) as a prior, the conditional posterior considering the observation $\boldsymbol{X}$ is written as follows: [1]

$$H_{kt} \sim \text{GIG}\left(2S_{kt}\textstyle\sum_f W_{fk}, \delta_H, \sum_f X_{ft}\lambda_{ftk} - \gamma_H\right),$$

where $\gamma_H = 2\eta_H$ and $\delta_H = \eta_H(\frac{1}{G_{k(t+1)}^H} + \frac{1}{G_{kt}^H})$. The conditional posterior of $\boldsymbol{W}^{\mathrm{n}}$ can be derived in the same manner as follows:

$$W_{fk}^{\mathrm{n}} \sim \text{GIG}\left(2\textstyle\sum_t H_{kt}S_{kt}, \delta_W, \sum_t X_{ft}\lambda_{ftk} - \gamma_W\right),$$

where $\gamma_W = 2\eta_W$ and $\delta_W = \eta_W(\frac{1}{G_{f+1}^W} + \frac{1}{G_f^W})$

3.4.3 Updating the Language Model

The latent variables $\boldsymbol{Z}$ are sampled from the following conditional posterior distribution:

$$p(z_t|\boldsymbol{S}, \boldsymbol{\pi}, \boldsymbol{\phi}, \Psi) \propto p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t, z_t), \tag{19}$$

where $\boldsymbol{\pi}$ is the emission probabilities, $\boldsymbol{\phi}$ is the initial probabilities, and $\Psi = \{\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_I\}$ is a set of the transition probabilities from each state. The right-hand side of Eq. (19) is further factorized using the conditional independence over $\boldsymbol{Z}$ and $\boldsymbol{S}$ as follows:

$$\begin{aligned} &p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t, z_t)\\ &= p(\boldsymbol{s}_t|z_t)\textstyle\sum_{z_{t-1}} p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{t-1}, z_{t-1})p(z_t|z_{t-1}), \qquad (20)\\ &p(\boldsymbol{s}_1, z_1) = p(z_1)p(\boldsymbol{s}_1|z_1) = \phi_{z_1}p(\boldsymbol{s}_1|\pi_{z_1}). \qquad (21)\end{aligned}$$

Using Eqs. (20) and (21) recursively, $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T|z_T)$ can be efficiently calculated via forward filtering and the last variable z_T is sampled according to $z_T \sim p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T|z_T)$. If the latent variables $z_{t+1}, \ldots, z_T$ are given, z_t is sampled from a posterior given by

$$p(z_t|\boldsymbol{S}, z_{t+1}, \ldots, z_T) \propto p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t, z_t)p(z_{t+1}|z_t). \tag{22}$$

Since $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t, z_t)$ can be calculated in Eq. (20), z_t is recursively sampled from $z_t \sim p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t, z_t)p(z_{t+1}|z_t)$ via backward sampling.

The posterior distribution of the emission probabilities $\boldsymbol{\pi}$ is given by using the Bayes' rule as follows:

$$p\left(\boldsymbol{\pi}|\boldsymbol{S}, \boldsymbol{z}, \boldsymbol{\phi}, \Psi\right) \propto p\left(\boldsymbol{S}|\boldsymbol{\pi}, \boldsymbol{z}, \boldsymbol{\phi}, \Psi\right) p\left(\boldsymbol{\pi}\right). \tag{23}$$

This is analytically calculable because $p\left(\boldsymbol{\pi}\right)$ is a conjugate prior of $p\left(\boldsymbol{S}|\boldsymbol{\pi}, \boldsymbol{z}, \boldsymbol{\phi}, \Psi\right)$. Let C_i be the number of occurrences of chord $i \in \{1 \ldots I\}$ in $\boldsymbol{Z}$ and $\boldsymbol{c}_i \equiv \sum_{t\in\{t|z_t=i\}} \boldsymbol{s}_t$ be a K-dimensional vector that denotes the sum of $\boldsymbol{s}_t$ under the condition $z_t = i$. The parameters $\boldsymbol{\pi}$ are sampled according to a conditional posterior given by

$$\boldsymbol{\pi} \sim \text{Beta}\left(e + c_{ik}, f + C_i - c_{ik}\right). \tag{24}$$

The posterior distributions of the transition probabilities $\boldsymbol{\psi}$ and the initial probabilities $\boldsymbol{\pi}$ are given similarly as follows:

$$\begin{aligned} &p(\boldsymbol{\phi}|\boldsymbol{S}, \boldsymbol{z}, \boldsymbol{\pi}, \Psi) \propto p(z_1|\boldsymbol{\phi})\, p(\boldsymbol{\phi}) \qquad (25)\\ &p(\boldsymbol{\psi}|\boldsymbol{S}, \boldsymbol{z}, \boldsymbol{\pi}, \boldsymbol{\phi}) \propto \textstyle\prod_t p(z_t|z_{t-1}, \boldsymbol{\psi}_{z_{t-1}})\, p(\boldsymbol{\psi}_{z_{t-1}}). \qquad (26)\end{aligned}$$

Since $p(\boldsymbol{\phi})$ and $p\left(\boldsymbol{\psi}_i\right)$ are conjugate priors of $p(z_1|\boldsymbol{\phi})$ and $p(z_t|z_{t-1}, \boldsymbol{\psi}_{z_{t-1}})$, respectively, these posteriors can be easily calculated. Let $\boldsymbol{e}_i$ be the unit vector whose i-th element

[1] $\text{GIG}(a, b, p) \equiv \frac{(a/b)^{\frac{p}{2}}}{2K_p(\sqrt{ab})} x^{p-1}\exp(-\frac{ax+\frac{b}{x}}{2})$ denotes a generalized inverse Gaussian distribution.

is 1 and $\boldsymbol{a}_i$ be the I-dimensional vector whose j-th element denotes the number of transition from state i to state j. The parameters $\boldsymbol{\phi}$ and $\boldsymbol{\psi}_i$ are sampled according to conditional posteriors given by

$$\boldsymbol{\phi} \sim \mathrm{Dir}\left(\mathbf{1}_I + \boldsymbol{e}_{z_1}\right), \quad \boldsymbol{\psi}_i \sim \mathrm{Dir}\left(\mathbf{1}_I + \boldsymbol{a}_i\right). \tag{27}$$

4. EVALUATION

We report comparative experiments we conducted to evaluate the performance of our proposal model in pitch estimation. First, we confirmed in a preliminary experiment that correct chord progressions and emission probabilities were estimated from the piano-roll by the language model. Then, we estimated the piano-roll representation from acoustic audio signals by using the hierarchical model and the acoustic model.

4.1 Experimental Conditions

We used 30 pieces (labeled as "ENSTDkCl") selected from the MAPS database [24]. We converted them into monaural signals and truncated each of them to 30 seconds from the beginning. The magnitude spectrogram was made by using the variable-Q transform [25]. The 926×10075 spectrogram thus obtained was resampled to 926×3000 by using MATLAB's resample function. Moreover, we used harmonic and percussive source separation (HPSS) [26] as a preprocessing. Unlike the original study, HPSS was performed in the log-frequency domain. Median filter is applied over 50 time frames and 40 frequency bins each. Hyperparameters were empirically determined as $I = 24, a^{\mathrm{h}} = 1, b^{\mathrm{h}} = 1, a^{\mathrm{n}} = 2, b^{\mathrm{n}} = 1, c = 2, d = 1, e = 5, f = 80, \alpha = 1300, \eta_W = 800000$ and $\eta_H = 15000$. The emission probabilities are obtained for 12 notes, which are expanded to cover 84 pitches. In practice, we fixed the probability of internal transition (*i.e.* $p(z_{t+1} = z_t | z_t)$) to a large value ($1 - 8.0 \times 10^{-8}$) and assumed that the probabilities of transition to a different state follow Dirichlet distribution as shown in section 3.4.3 We implemented the proposed method by using C++ and a linear algebra library called Eigen3. The estimation was conducted with a standard desktop computer with an Intel Core i7-4770 CPU (8-core, 3.4 GHz) and 8.0 GB of memory. The processing time for the proposed method with one music piece (30 seconds as mentioned above) was 15.5 minutes .

4.2 Chord Estimation for Piano Rolls

We first verified that the language model properly estimated the emission probabilities and a chord progression. As an input, we combined correct binary piano-roll representations for 84 pitches (MIDI numbers 21–104) of the pieces we used. Since each representation has 3000 time-frames and we used 30 pieces, the input was 84×90000 matrix. We evaluated the precision of chord estimation as the ratio of the number of frames whose chords were estimated correctly to the total number of frames. Since we prepared two chord types for each root note, we treated "major" and "7th" in the ground-truth chords as "major" in the estimated chords, and "minor" and "minor 7th" in the ground-truth chords as "minor" in the estimated chords. In evaluation, other chord types were not used in evaluation and chord labels were estimated to maximize the precision since we estimated chords in an unsupervised manner. Since original MAPS database doesn't contain chord information, one of the authors labeled chord information for each music piece by hand [2].

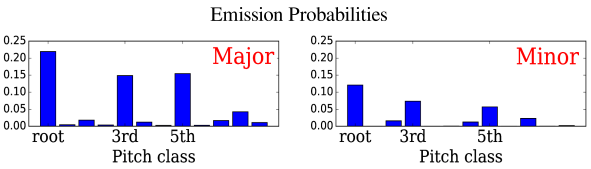


Figure 4. Emission probabilities estimated in the preliminary experiment. The left corresponds to major chords and the right corresponds to minor chords.

The experimental results shown in Fig. 4 shows that major chords and minor chords, which are typical chord types in tonal music, were obtained as emission probabilities. This implies that we can obtain the concept of chord from piano-roll data without any prior knowledge. The precision was 61.33%, which indicates our model estimates chords correctly to some extent even in an unsupervised manner. On the other hand, other studies on chord estimation have reported higher score [15, 16]. This is because that they used labeled training data and that they evaluated their method with popular music, which has clearer chord structure than classical music we used.

4.3 Multipitch Estimation for Music Audio Signals

We then evaluated our model in terms of the frame-level recall/precision rates and F-measure:

$$\mathcal{R} = \frac{\sum_t c_t}{\sum_t r_t}, \ \mathcal{P} = \frac{\sum_t c_t}{\sum_t e_t}, \ \mathcal{F} = \frac{2\mathcal{R}\mathcal{P}}{\mathcal{R}+\mathcal{P}}, \tag{28}$$

where r_t, e_t, and c_t are respectively the numbers of ground truth, estimated and correct pitches at the t-th time-frame. To cope with the arbitrariness in octaves of the obtained bases, estimated results for the whole piece were shifted by octaves and the most accurate one was used for the evaluation. We conducted a few comparative experiments under the following conditions: 1) Chords were fixed and unchanged during a piece (the acoustic model), 2) the language model was pre-trained using the correct chord labels and a correct piano-roll, and the learned emission probabilities were used in estimation (pre-trained with chord), 3) the language model was pre-trained using only a correct piano-roll, and the learned emission probabilities were used in estimation (pre-trained without chord). we evaluated the performances under the second and the third conditions by using cross-validation.

As shown in Table 1, the performance of the proposed method in the unsupervised setting (65.0%) was better than that of the acoustic model (64.7%). As shown in Fig. 5, the F-measure improvement due to integrating the language model for each piece correlated positively with the preci-

[2] The annotation data used for evaluation is available on http://sap.ist.i.kyoto-u.ac.jp/members/ojima/mapschord.zip

Condition	$\mathcal{F}$	$\mathcal{R}$	$\mathcal{P}$
The integrated model	65.0	67.3	62.8
The acoustic model	64.7	64.7	64.7
Pre-trained w/ chord	65.5	65.3	65.6
Pre-trained w/o chord	65.0	65.5	64.6

Table 1. Experimental results of multipitch analysis for 30 piano pieces labeled as ENSTDkCl.

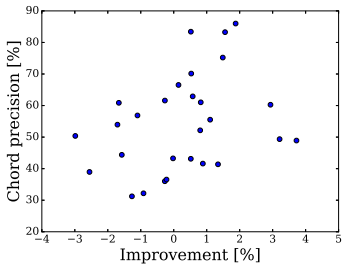


Figure 5. Correlation between estimated chord precision and the improvement of F-measure.

sion of chord estimation for each piece (correlation coefficient $r = 0.33$). This indicates that refining the language model also improves the pitch estimation.

Moreover, as shown in Fig. 6, major and minor chords like those in Fig. 4 were obtained as emission probabilities directly from music audio signals without any prior knowledge. This implies that frequently used chord types can be inferred from music audio signals automatically, which would be useful in music classification or similarity analysis. The performance in the supervised setting (65.5%) was better than the performance obtained in the unsupervised settings. Since there exist published piano scores with chord labels, this setting is considered to be practical. Although this difference was statistically insignificant (standard error was about 1.5%), F-measures were improved for 25 pieces out of 30. Moreover, the improvement exceeded 1% for 15 pieces. The example of pitch estimation shown in Fig. 7 indicates that insertion errors at low pitches are reduced by integrating the language model. On the other hand, insertion errors in total increased in the integrated model. This is because the constraint on harmonic partials (shift-invariant) is too strong to appropriately estimate the spectrum of each pitch. As a result, the overtones that should be expressed by a single pitch are expressed by multiple inappropriate pitches that do not exist in the ground-truth.

There would be much room for improving the performance. The acoustic model has the strong constraint on harmonic partials as mentioned above. This constraint can be relaxed by introducing source-filter NMF [4], which further decomposes the bases into sources corresponding to pitches and filters corresponding to timbres. Our model corresponds the case the number of filters is one, and increment of the number of filters would contribute to express difference in timbres (*e.g.*, difference between the timbre of high pitches and that of low pitches). The language model, on the other hand, can be refined by introducing other music theory such as keys. Some methods that treat the relationship between keys and chords [27], or keys and notes [12], have been studied. Moreover, the language model focus on reducing unmusical errors such as insertion errors in adjacent pitches, and is difficult to cope with errors in octaves or overtones. Modeling transitions between notes (horizontal relations) will contribute to solve this problem and to improve the accuracy.

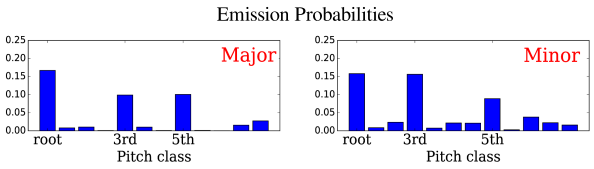


Figure 6. Emission probabilities learned from estimated piano-roll. Chord structures like those in Fig. 4 were obtained.

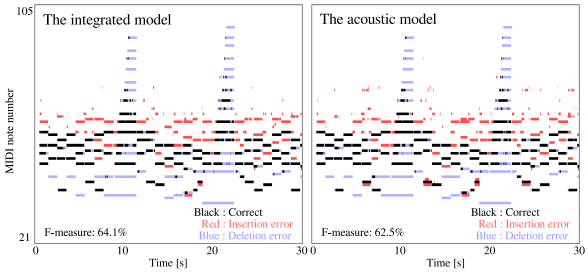


Figure 7. Estimated piano-rolls for MUS-bk_xmas5_ENSTDkCl. Integrating the language model redeuced Insertion errors at low pitches.

5. CONCLUSION

We presented a new statistical multipitch analyzer that can simultaneously estimate pitches and chords from music audio signals. The proposed model consists of an acoustic model (a variant of Bayesian NMF) and a language model (Bayesian HMM), and each model can make use of each other's information. The experimental results showed the potential of the proposed method for unified music transcription and grammar induction from music audio signals. On the other hand, each model has much room for performance improvement: the acoustic model has a strong constraint, and the language model is insufficient to express music theory. Therefore, we plan to introduce a source-filter model as the acoustic model and to introduce the concept of key in the language model.

Our approach has a deep connection to language acquisition. In the field of natural language processing (NLP), unsupervised grammar induction from a sequence of words and unsupervised word segmentation for a sequence of characters have actively been studied [28, 29]. Since our model can directly infer musical grammars (*e.g.*, concept of chords) from either music scores (discrete symbols) or music audio signals, the proposed technique is expected to be useful for an emerging topic of language acquisition from continuous speech signals [30].

Acknowledgement: This study was partially supported by JST OngaCREST Project, JSPS KAKENHI 24220006, 26700020, 26280089, 16H01744, and 15K16054, and Kayamori Foundation.''

6. REFERENCES

[1] P. Smaragdis and J. C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *IEEE WASPAA*, pages 177–180, 2003.

[2] K. Ohanlon, H. Nagano, N. Keriven, and M. Plumbley. An iterative thresholding approach to L0 sparse hellinger NMF. In *ICASSP*, pages 4737–4741, 2016.

[3] M. Hoffman, D. M. Blei, and P. R. Cook. Bayesian nonparametric matrix factorization for recorded music. In *ICML*, pages 439–446, 2010.

[4] T. Virtanen and A. Klapuri. Analysis of polyphonic audio using source-filter model and non-negative matrix factorization. In *Advances in models for acoustic processing, neural information processing systems workshop*. Citeseer, 2006.

[5] J. L. Durrieu, G. Richard, B. David, and C. Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE TASLP*, 18(3):564–575, 2010.

[6] D. Liang and M. Hoffman. Beta process non-negative matrix factorization with stochastic structured mean-field variational inference. *arXiv*, 1411.1804, 2014.

[7] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE TASLP*, 18(3):528–537, 2010.

[8] G. E. Poliner and D. P. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Applied Signal Processing*, 2007.

[9] A. T. Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009.

[10] M. Hamanaka, K. Hirata, and S. Tojo. Implementing a generative theory of tonal music. *Journal of New Music Research*, 35(4):249–277, 2006.

[11] E. Nakamura, M. Hamanaka, K. Hirata, and K. Yoshii. Tree-structured probabilistic model of monophonic written music based on the generative theory of tonal music. In *ICASSP*, 2016.

[12] D. Hu and L. K. Saul. A probabilistic topic model for unsupervised learning of musical key-profiles. In *ISMIR*, pages 441–446, 2009.

[13] R. Jackendoff and F. Lerdahl. *A generative theory of tonal music*. MIT Press, 1985.

[14] M. Rocher, T.and Robine, P. Hanna, and R. Strandh. *Dynamic Chord Analysis for Symbolic Music*. Ann Arbor, MI: MPublishing, University of Michigan Library, 2009.

[15] A. Sheh and D. P. Ellis. Chord segmentation and recognition using EM-trained hidden Markov models. In *ISMIR*, pages 185–191, 2003.

[16] S. Maruo, K. Yoshii, K. Itoyama, M. Mauch, and M. Goto. A feedback framework for improved chord recognition based on NMF-based approximate note transcription. In *ICASSP*, pages 196–200, 2015.

[17] Y. Ueda, Y. Uchiyama, T. Nishimoto, N. Ono, and S. Sagayama. HMM-based approach for automatic chord detection using refined acoustic features. In *ICASSP*, pages 5518–5521, 2010.

[18] S. Raczynski, E. Vincent, F. Bimbot, and S. Sagayama. Multiple pitch transcription using DBN-based musicological models. In *ISMIR*, pages 363–368, 2010.

[19] S. A. Raczynski, E. Vincent, and S. Sagayama. Dynamic bayesian networks for symbolic polyphonic pitch modeling. *IEEE TASLP*, 21(9):1830–1840, 2013.

[20] S. Sigtia, E. Benetos, and S. Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE TASLP*, 24(5):927–939, 2016.

[21] S. Sigtia, E. Benetos, S. Cherla, T. Weyde, A. Garcez, and S. Dixon. An RNN-based music language model for improving automatic music transcription. In *ISMIR*, pages 53–58, 2014.

[22] A. T. Cemgil and O. Dikmen. Conjugate Gamma Markov random fields for modelling nonstationary sources. In *Independent Component Analysis and Signal Separation*, pages 697–705. Springer, 2007.

[23] M. Davy and S. J. Godsill. Bayesian harmonic models for musical signal analysis. *Bayesian Statistics*, (7):105–124, 2003.

[24] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE TASLP*, 18(6):1643–1654, 2010.

[25] C. Schörkhuber, A. Klapuri, N. Holighaus, and M. Dörfler. A Matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution. In *Audio Engineering Society Conference*, 2014.

[26] D. Fitzgerald. Harmonic/percussive separation using median filtering. In *DAFx*, pages 1–4, 2010.

[27] K. Lee and M. Slaney. Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio. *IEEE TASLP*, 16(2):291–301, 2008.

[28] M. Johnson. Using adaptor grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*, pages 398–406, 2008.

[29] D. Mochihashi, T. Yamada, and N. Ueda. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *ACL*, pages 100–108. Association for Computational Linguistics, 2009.

[30] T. Taniguchi and S. Nagasaka. Double articulation analyzer for unsegmented human motion using Pitman-Yor language model and infinite hidden markov model. In *IEEE/SICE International Symposium on System Integration*, pages 250–255. IEEE, 2011.

A HIGHER-DIMENSIONAL EXPANSION OF AFFECTIVE NORMS FOR ENGLISH TERMS FOR MUSIC TAGGING

Michele Buccoli[1], **Massimiliano Zanoni**[1], **György Fazekas**[2]
Augusto Sarti[1], **Mark Sandler**[2]
[1] Dipartimento di Elettronica, Informatica e Bioingegneria, Politecnico di Milano, Italy
[2] Centre For Digital Music, Queen Mary, University of London, UK
michele.buccoli@polimi.it, g.fazekas@qmul.ac.uk

ABSTRACT

The Valence, Arousal and Dominance (VAD) model for emotion representation is widely used in music analysis. The ANEW dataset is composed of more than 2000 emotion related descriptors annotated in the VAD space. However, due to the low number of dimensions of the VAD model, the distribution of terms of the ANEW dataset tends to be compact and cluttered. In this work, we aim at finding a possibly higher-dimensional transformation of the VAD space, where the terms of the ANEW dataset are better organised conceptually and bear more relevance to music tagging. Our approach involves the use of a kernel expansion of the ANEW dataset to exploit a higher number of dimensions, and the application of distance learning techniques to find a distance metric that is consistent with the semantic similarity among terms. In order to train the distance learning algorithms, we collect information on the semantic similarity from human annotation and editorial tags. We evaluate the quality of the method by clustering the terms in the found high-dimensional domain. Our approach exhibits promising results with objective and subjective performance metrics, showing that a higher dimensional space could be useful to model semantic similarity among terms of the ANEW dataset.

1. INTRODUCTION

One of the fundamental properties of music is the ability to convey emotions [27]. Consequently, there is a great interest in representing and classifying music according to emotions in areas like music information retrieval, music recommendation and personalisation [11–13, 27]. It has been proven that listeners enjoy looking for and discovering music using mood-based queries, which represents 33% of music queries according to [13]. This is an important reason that urged psychologist and musicologist to investigate different paradigms for the representation and modelling of emotion related descriptors [11–13].

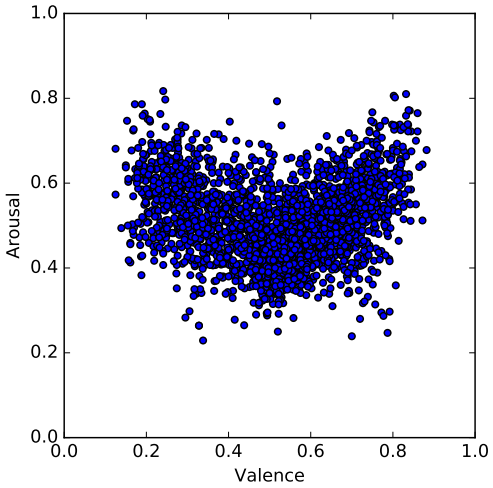


Figure 1: Distribution of the ANEW dataset in the VA space

Dimensional approaches to emotion conceptualisation focus on describing emotional states in a continuous space, where emotion states are represented as points or distributions in an N-dimensional space. Specific emotion terms can be localised in such continuous space and it is possible to define a metric in a way that the distance between points is proportional to the semantic distance between emotions. The most influential dimensional models so far [1, 25] are proposed by Russell [19] and Thayer [24]. Russell devised a *circumplex model of affect* which consists of a two-dimensional, circular structure involving the dimensions of *arousal* (A), linked to the degree of activation or excitement, and *valence* (V), linked to the degree of pleasantness. A third dimension related to *dominance* (D) was later proposed to express the degree of control and to possibly distinguish different and overlapping moods. [8, 21].

The increasing need of a continuous affective model led to the creation of the ANEW dataset [4] for Psychological research. This dataset is composed of 2,476 English words with positions in the VAD space. Despite it is a generic dataset, the large amount of terms in ANEW makes it a powerful resource also for the Music Emotion Recognition (MER) community, including applications for

© Michele Buccoli, Massimiliano Zanoni, György Fazekas, Augusto Sarti, Mark Sandler. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Michele Buccoli, Massimiliano Zanoni, György Fazekas, Augusto Sarti, Mark Sandler. "A higher-dimensional expansion of affective norms for English terms for music tagging", 17th International Society for Music Information Retrieval Conference, 2016.

Type	Symbol	Num Terms	Details
Terms Dataset	$\mathcal{D}_{ANEW}$	2,476	Mean in V, A, D dimensions
Implicit Distance	$\mathbf{M}_{ILM10K}$	240, 450	Tag compact representation from an LSA with $k = 10, 20, 50, 100$ components
Explicit Distance	$\mathbf{M}_{HD}$	180	Human annotation of semantic similarity between terms from ANEW
Explicit Clustering	$\mathbf{M}_{HC}$	100	Human clustering of a subset of ANEW

Table 1: Summary of the collected data

automatic music annotation and retrieval [5, 20]. Unfortunately, ANEW terms in the VAD space tend to have a very uniform and compact distribution concentrated around the centre of the space as shown in Figure 1 for the VA subspace. Although the use of a substantial set of terms provides a very representative model of a large variety of emotions, to deal with a compact and cluttered distribution can be problematic in many musical applications. For this reason, typically only a subset of the terms is used, leading to a loss in the exhaustiveness of the model. A higher-dimensional mood space drawn from the ANEW dataset, where the terms are distributed following some conceptual organisation can benefit several applications. These include semantic music annotation, recommendation and mood-based music retrieval [2].

In this study, we investigate the construction of higher-dimensional emotional spaces from ANEW by means of kernel expansion techniques applied to the VAD space. Although the transformation has the effect to produce a more sparse distribution of terms, it is not clear if the semantic distance between concepts is well represented by the metric in the new space. To solve this problem, we first aim to find a distance reflecting conceptual organisation of terms by applying distance learning techniques [3, 10, 26]. Given some constraints between terms, these methods search for a linear transformation of the space that is semantically relevant. That is, the ideal learnt distance closely correlates with semantic differentiae given by users in a specific task for a subset of the ANEW terms. We generate the set of constraints using "a priori" information collected through a subjective test where participants were asked to specify the semantic similarity between pairs of terms in the context of music. We then perform Latent Semantic Analysis on emotion related annotations for a large set of music pieces.

Under the hypothesis that the terms may be improved by this transformation, we validate the approach by clustering in the new high-dimensional space. We subsequently evaluate the resulting clusters with objective and subjective metrics. Tests show promising results given these new learnt distance metrics and provide an insight into how a better configuration in the high-dimensional space can be achieved.

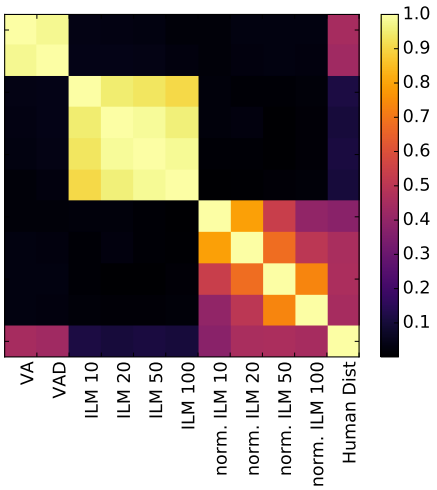


Figure 2: Absolute Pearson Correlation of the self-similarity matrices computed or collected from different sources of data.

2. DATASET CONSTRUCTION

In this section, we describe three datasets created to provide constraints for semi-supervised learning as well as to validate our approach. A summary of the collected data is provided in Table 1.

2.1 Terms in the ANEW Dataset

The terms in the ANEW dataset [4] is already annotated for the Valence, Arousal and Dominance dimensions with a value between 0 and 10 by Psychology class students. For each term, we consider the normalised average (between 0 and 1) of the annotations.

2.2 Implicit Distance Annotation

We compute a similarity distance matrix among emotion related descriptors by performing Latent Semantic Analysis (LSA) on the I-Like-Music [1] (ILM) dataset denoted ILM10K. This is composed of 10,199 tracks from commercial music annotated with crowed sourced editorial and social tags [2, 20] with weights corresponding to the prevalence of each tag.

From the tags in ILM10K, we first discard the tags that are not included in the ANEW dataset. We then filter the tags that are rarely used by means of two thresholds on the number of times the tag is used in ILM10K: 15 times, leading to a set of $T = 240$ terms and 5 times leading to $T = 450$ terms. We build a track-tag matrix using the remaining tags and compute a k-component approximation of this matrix via LSA, keeping different numbers of components k to test different degrees of approximations. We set $k = 10, 20, 50, 100$ components to produce the set of matrices $\mathbf{D}_{ILM10K} \in \mathbb{R}^{T \times k}$.

From $\mathbf{D}_{ILM10K}$, we compute the similarity between tags as the Euclidean distance between the correspondent

[1] http://www.ilikemusic.com/

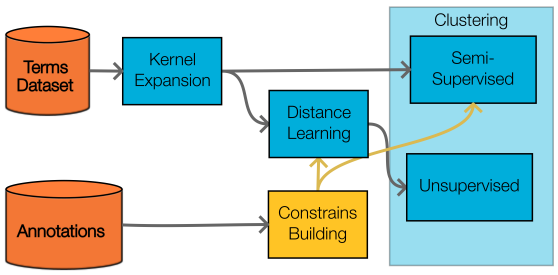


Figure 3: Block diagram of the clustering approach

rows, producing the matrix $\mathbf{M}_{ILM10K}$. We also compute the normalised matrix $\hat{\mathbf{D}}_{ILM10K}$, composed by the L^2 normalised rows of $\mathbf{D}_{ILM10K}$, so that the Euclidean distance of the former is equal to the cosine distance of the latter and compute the (cosine) distance matrix $\hat{\mathbf{M}}_{ILM10K}$.

2.3 Human Distance Annotation

We conducted an online survey and asked annotators to define the perceived mood similarity in the context of music between pairs of descriptors with a value between 0 (very similar) and 1 (not similar at all). 504 people participated in the survey. From the data we kept only the terms that received at least 2 annotations leading to $T = 180$ and we compose the sparse matrix $\mathbf{M}_{HD}$.

2.4 Human Clustering Annotation

We conducted a second online survey and asked annotators to group the set of top 100 descriptors in the dataset. 15 people participated in the survey, leading to the matrix $\mathbf{M}_{HC}$ with $T = 100$ terms, where each entry (i, j) indicates the number of people that grouped together the i-th and j-th terms.

2.5 Further Considerations

In Figure 2 we show the absolute Pearson correlations between the similarity matrices from the different sources of data we have mentioned so far. The human distance annotation exhibits a modest correlation with the Euclidean distance defined in the VA or VAD space. The annotations on distance between terms is also somewhat correlated with the distance that can be inferred from editorial tags. In the rest of this study, we aim to find a space where the defined distance metric provides a better representation of the perceived similarity in musical emotion.

3. HIGH-DIMENSIONAL SPACE LEARNING

The block diagram of our approach is shown in Figure 3. The VA or VAD coordinates for the ANEW dataset are processed using kernel expansion. In order to better represent the perceived similarity, we first rotate and translate the expanded dataset by means of distance learning algorithms.We then use the collected annotations to build constraints for learning the metric distance.

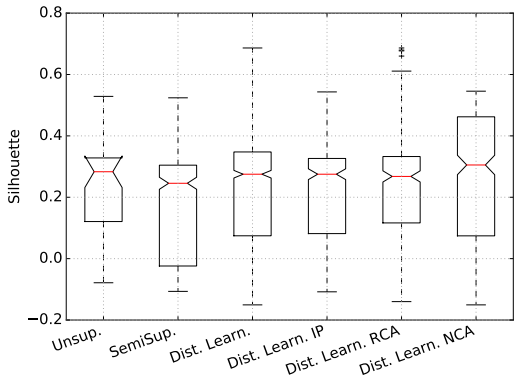


Figure 4: Boxplot of the Silhouette indices for the different scenarios

3.1 Kernel Expansion

Given a generic vector $\mathbf{x} \in \mathbb{R}^N$, we define its kernel expansion the vector $\phi_{\mathbf{x}} = \Phi(\mathbf{x}) \in \mathbb{R}^P$, with $P \geq N$, that is the result of the mapping function Φ. We expand our original dataset (both VA and VAD coordinates) with the following mapping functions: normalisation with respect to the L^2 norm, to include the cosine distance, i.e., $\Phi(\mathbf{x}) = \mathbf{x}/|\mathbf{x}|_2$; polynomial expansion with degrees two and three, to include nonlinear distance, i.e., $\Phi(x)_2 = [x_1, ..., x_N, x_1^2, ..., x_N^2, x_1x_2, ..., x_1x_N, ..., x_{N-1}, x_N]$ for the polynomial expansion of degree two, where $x_1, ..., x_N$ are the components of $\mathbf{x}$ (the third degree polynomial expansion is computed accordingly); approximate feature map of a Radial Basis Function (RBF) kernel by Monte Carlo expansion of its Fourier Transform [17].

3.2 Constrains Building

In order to facilitate the training of distance learning algorithms, we use similarity matrices obtained from human annotations introduced in Sections 2.2, 2.3, 2.4. Our method relies on the definition of a set of Must-Link (ML) and Cannot-Link (CL) constraints. Treating the similarity as a tendency of terms to be grouped together, ML constraints force a pair of terms to be in the same group, whereas CL constraints force a pair of terms to be in different groups. In the distance learning techniques, the space is transformed such that the pairs of points of the ML set are closer together while the pairs of points in the CL set are far apart.

As far as the ILM10K dataset is concerned, we compute the ML and CL constraints by computing the mean value μ and standard deviation σ of the matrices $\mathbf{M}_{ILM10K}$ and empirically defining two thresholds $th_l = \mu - 2\sigma$, $th_h = \mu + 2\sigma$. We compose the set of ML constraints by considering those pairs of terms which are close in the space defined by LSA, hence whose distance is lower than th_l. Similarly, we compose the CL set with those pairs of terms that are far from each other, i.e., whose distance is higher than th_h. We compose another set of constrains in the same way from the matrices $\hat{\mathbf{M}}_{ILM10K}$.

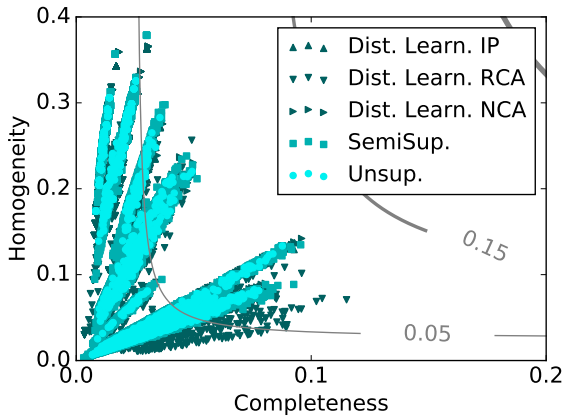


Figure 5: Completeness and Homogeneity results for the comparison of clusters with those obtained from the ILM10K dataset. Gray contour lines indicate the V-score

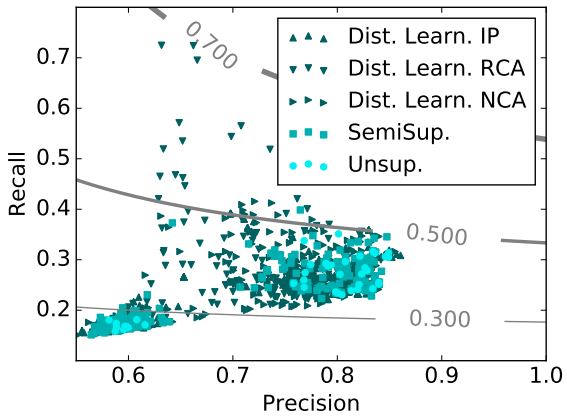


Figure 6: Precision and Recall results for the comparison of clusters with those obtained from human annotation. Gray contour lines indicate the F-measure.

We use a similar approach to compose the ML and CL constraints from the Human Distance annotations. We compute the mean value μ and standard deviation σ of the annotations in the matrix $\mathbf{M}_{HD}$ and we empirically define the thresholds $th_l = \mu - \sigma,\ th_h = \mu + \sigma$.

As far as the Human Clustering annotations are concerned, we compute the mean value μ and standard deviation σ of the non-zero entries of the matrix $\mathbf{M}_{HC}$, i.e., of the number of people who annotated two terms as belonging to the same clusters, and define the soft threshold $th = \{\mu - \sigma\}$. We compose ML with the pair of terms that have been grouped together by more than th people and we compose CL with the zero entries of $\mathbf{M}_{HC}$.

3.3 Distance Learning

Given $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$ two generic vectors, we can weight the contribution of each components and the inter-correlation among them by defining a Mahalanobis (symmetric, square) matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ and computing:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^\intercal \mathbf{A} (\mathbf{x} - \mathbf{y})} = \sqrt{(\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{y})^\intercal (\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{y})}, \quad (1)$$

that is the Euclidean distance between the projected vectors over the subspace defined by $\mathbf{L}$, with $\mathbf{L}^\intercal \mathbf{L} = \mathbf{A}$.

Distance metric learning is the sub-field of machine learning that aims at finding the best subspace $\mathbf{L}$, from a set of constraints. Using the constraints as described in Section 3.2, we compute a set of subspaces for each combination of input, kernel expansions and constraints. It is worth remembering that with the Mahalanobis distance, only translation and rotation operations are performed. However, the application of kernels on the sample data allows nonlinear transformation of the original space of data.

We employ the following distance metric learning algorithms [6]:

- Iterative Projection [26] (IP), computes the Mahalanobis matrix by means of an iterative minimisation of the distance of the ML data with the constraint to keep CL data far apart
- Relative Components Analysis [3] (RCA), learns a Mahalanobis matrix that assigns large weights to more relevant components and low weights to irrelevant ones, by using chunklets, i.e., subset of data that belong to the same group (i.e., have been defined in the ML set)
- Neighbourhood Components Analysis [10] (NCA) is a component analysis that computes an optimal Mahalanobis matrix for the K-nearest neighbour clustering algorithm.

4. EXPERIMENTAL SETUP AND EVALUATION

As previously discussed, our aim is to find a transformation of the ANEW space with improved conceptual organisation of terms that is relevant in a musical context. We expect terms that are semantically similar in this context to be close and dissimilar terms to be far apart. For this reason, we validate our approach by clustering in the transformed space.

Specifically, we perform three evaluations: i) we evaluate the resulting clusters by mean of the Silhouette quality index as objective metric; ii) we compare the resulting clusters with those obtained by clustering the tags of the ILM10K dataset; iii) we compare the resulting clusters with those obtained from manual annotation.

In the ILM10K dataset we consider the results obtained with both $T = 240$ and 450 tags. This is because the former is less noisy, while the latter provides more information and constrains. We will then specify in the single cases which dataset achieved the mentioned results.

4.1 Unsupervised and Semi-Supervised Clustering

In order to provide a robust evaluation, we apply several clustering techniques. We experimentally choose to retrieve the 6 best representative clusters, given the large

Scenario	Features	Algorithm	Constraints	Silhouette
Unsupervised	Norm. VA	K-Means		0.5287
Semi-Supervised	Norm. VA	SC	$\mathbf{M}_{HC}, th = \mu - \sigma$	0.5240
IP Distance Learning	VAD	kmeans	$\mathbf{M}_{HD}$	0.5432
RCA Distance Learning	VAD, poly 3 degree	AHC	$\mathbf{M}_{HC}, th = \mu - \sigma$	0.6864
NCA Distance Learning	VA, poly 2 degree	kmeans	$\hat{\mathbf{M}}_{ILM10K}, T = 240, k = 10$	0.5456

Table 2: Best results for the Silhouette metric

Scenario	Features	Algorithm	Constraints	ILM10K	Compl.	Hom.	V-score
Unsup.	VA, poly 3 degree	SC		AHC $\hat{\mathbf{M}}_{ILM10K}, k = 100$	0.0877	0.1334	0.1058
Semi-Sup.	VAD poly 2 degree	AHC	$\mathbf{M}_{HD}$	AHC $\hat{\mathbf{M}}_{ILM10K}, k = 100$	0.0955	0.1347	0.1118
IP Dist.	VA	SC	$\mathbf{M}_{HD}$	AHC $\hat{\mathbf{M}}_{ILM10K}, k = 100$	0.0929	0.1371	0.1107
RCA Dist.	VAD	kmeans	$\mathbf{M}_{HD}$	AHC $\hat{\mathbf{M}}_{ILM10K}, k = 100$	0.0869	0.1290	0.1038
NCA Dist.	VA, poly 2 degree	AHC	$\mathbf{M}_{HD}$	AHC $\hat{\mathbf{M}}_{ILM10K}, k = 100$	0.0959	0.1421	0.1145

Table 3: Best results of the Homogeneity, Separation and V-measure metrics for the comparison with the clusters generated by ILM10K dataset (240 terms)

number of configurations we need to test. We employ the following algorithms resulted to be effective in the context of music tag analysis and aggregation (from [16]):

- K-Means [14] is the common unsupervised clustering algorithm;
- Semi-Supervised Non-negative Matrix Factorisation [7] (SS-NMF) applies NMF techniques to the self-distance matrix of the samples;
- Spectral Clustering [22] (SC) employs a low-dimension reduction of the similarity matrix between samples before applying the K-means algorithm;
- Agglomerative Hierarchical Clustering [23] (AHC) performs a bottom-up clustering: initially every sample is a cluster, then the closest clusters merge together until the final number of clusters is reached.

In this study we use the SS-NMF, SC and AHC algorithms in both unsupervised and semi-supervised fashion. In semi-supervised algorithms, the clustering can be guided by constraints. Here we use the ML and CL constraints computed in Section 3.2.

In order to validate the actual contribution of the distance learning techniques, we compare the results of our approach with the results obtained with both unsupervised and semi-supervised clustering of the ANEW dataset. Hence, we have three scenarios: i) unsupervised clustering of the transformed (but not learned) space (Unsup.) ; ii) the semi-supervised clustering of the transformed (but not learned) space (SemiSup.); iii) our approach, that is the unsupervised clustering of the learned space (Dist. Learn). Please note that the first scenario also includes the clustering of non-transformed space using the Euclidean distance.

4.2 Objective metrics

We evaluate the objective quality of clustering by using the Silhouette index that is defined as [16]:

$$\text{silhouette} = \frac{1}{|\mathcal{D}|} \sum_{s \in \mathcal{D}} \frac{b - a}{\max(a, b)} \in [-1, 1], \qquad (2)$$

where a and b are the mean distance between the s-th sample and all other points in the same cluster and in the next nearest cluster, respectively and $\mathcal{D}$ is the dataset of samples. High (positive) values of Silhouette indicate dense well-separated clusters, values around 0 indicate overlapping clusters and low (or negative) values indicate incorrect clusters. In Figure 4 we show the boxplots of Silhouette metric for the different scenarios, while in Table 2 we show the configurations that generate the best results for each scenario.

It is clear that the application of Distance Learning techniques outperforms unsupervised and semi-supervised techniques, even with different configurations of kernels, algorithms and constraints. In particular, the best performance is obtained with the AHC over the third degree polynomial expansion of the VAD dataset, with a translation learned using the RCA technique with Human Clustering.

We can also notice that the semi-supervised scenario performs on average worse than the unsupervised scenario. This is because the Silhouette index evaluates the resulting clusters with respect of the position of the input data, that is not moved from the original position. The estimated clusters are therefore more noisy, which confirms the advantage of distance learning techniques to transform the space.

4.3 Subjective metrics - Comparison with Clustering of ILM10K

We compare the clusters obtained with the different approaches with the clusters obtained from ILM10K data. We consider the homogeneity and completeness metrics [18]. The former evaluates whether each estimated cluster contains only members of a group in the ground truth, while the latter estimates whether the samples of a given group belongs to the same estimated cluster. We also consider the V-measure, i.e., the harmonic mean of homogeneity and completeness. To avoid overfitting issues, we evaluate only the configurations that are not trained with the constraints generated from the ILM10K dataset.

Scenario	Features	Algorithm	Constraints	P	R	F
Unsup.	VAD, RBF	kmeans		0.8012	0.3512	0.4883
Semi-Sup.	VAD, RBF	AHC	$\mathbf{M}_{HD}$	0.7646	0.3986	0.5240
IP Dist.	VAD, RBF	AHC	$\hat{\mathbf{M}}_{ILM10K}, T = 240, k = 100$	0.6979	0.3915	05016
RCA Dist.	VA, poly 3 degree	AHC	$\hat{\mathbf{M}}_{ILM10K}, T = 240, k = 50$	0.6622	0.7241	0.6918
NCA Dist.	VAD, RBF	kmeans	$\hat{\mathbf{M}}_{ILM10K}, T = 240, k = 10$	0.7971	0.4124	0.5289

Table 4: Best results for the Precision, Recall and F-measure metric for the comparison with the human annotated clusters

We show the distribution of results for the different scenarios in Figure 5. We notice the results are fairly low, showing that the organisation given by the expanded and possibly learnt ANEW dataset is very different from that obtained from editorial tags on a real music annotation application. This confirms the necessity to find a space for the ANEW dataset which is more useful for MIR applications. However, it is clear that our approach can only slightly improve the task. In Table 3 we list the configurations that lead to the best performance. These are all obtained with the 240-term subset of the ILM10K dataset. The AHC over the normalised ILM10K dataset with 100 components is the one that best matches all scenarios, from which we can infer that the clustering of the ANEW dataset resembles a clustering based on cosine distance with a large number of components. The best results is reached using NCA technique with a AHC over the polynomial- expanded dataset, by means of annotated distance again.

4.4 Subjective metrics - Comparison with Human Annotations

We finally compare the obtained clusters with those collected from human annotations. Since the correctly grouped terms, i.e., the amount of True Positive (TP) examples are more specific and relevant than the correctly non-grouped ones (True Negative, TN), we consider the Precision (P) and Recall (R) metrics that focus on the number of TP examples. The Precision indicates the ratios of True Positive over the total estimated assignments, i.e., $P = |TP|/(|TP| + |FP|)$, while the Recall defines the number of TP over the total assignments in the ground truth, i.e., $R = |TP|/(|TP| + |FN|)$. We also consider the F-measure (F) as the harmonic mean of the two metrics [15]. To avoid overfitting, we evaluate only configurations that are not trained with the constraints generated from the human annotations on clustering.

In Figure 6 we show the distribution of results for the different scenarios and configurations. In general, Precision is higher than Recall, showing that the estimated clustering is not capable of retrieving all the corrected groups. The F-measures are mostly distributed between 0.3 and 0.5, which is an average result. We can clearly see that some configuration from RCA Distance Learning are able to improve the average Recall and improve the F-measure reaching almost at 0.7. In Table 4 we list the best configurations for each scenarios. The RCA Distance Learning technique clearly outperforms the other approaches and matches very closely the human annotations, i.e., the human way to organise the emotional-related descriptors. We can notice that once again, the best results are obtained by using the AHC over some polynomial expansion of the dataset. However, the constraints based on distance annotations are less helpful for emulating the human organisation of terms than the ones based on editorial and social tags (ILM10K).

5. CONCLUSIONS

We introduced a novel approach consisting of kernel expansion, constraint building from music task specific human annotation and finally distance learning to transform the ANEW dataset and obtain a distribution of terms with better conceptual organisation compared to the conventional VA or VAD space. This facilitates applications that require computer representation of music emotions, including music emotion recognition and music tagging, music recommendation or interactive musical applications such as [9] and [2].

Average and maximum results presented in Section 4 prove that distance learning techniques are effective in the improvement of the organisation of concepts of the ANEW dataset. In particular, hierarchical clustering of the RCA-learned space based on the polynomial expansion of VA/VAD space outperforms the other configurations. The paper also introduces a new task along with a set of techniques that proved successful as a first attempt. This provides useful insight into improving the organisation of mood terms to better reflect application specific semantic spaces. Future work involves the collection of additional "a priori" information for improving the robustness of evaluation, as well as further assessment of the constructed high-dimensional space within other MIR applications.

6. ACKNOWLEDGEMENTS

This work was part funded by the FAST IMPACt EPSRC Grant EP/L019981/1 and the EC H2020 research and innovation grant AudioCommons (688382). Sandler acknowledges the support of the Royal Society as a recipient of a Wolfson Research Merit Award.

7. REFERENCES

[1] A. Aljanaki, Y.-H. Yang, and M. Soleymani. Emotion in music task at mediaeval 2015. In *Working Notes Proceedings of the MediaEval 2015 Workshop*, 2015.

[2] A. Allik, G. Fazekas, M. Barthet, and M. Sandler. mymoodplay: an interactive mood-based music discovery app. In *proc. 2nd Web Audio Conference (WAC)*, 2016.

[3] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proc. of the 20th International Conference on Machine Learning (ICML)*, 2003.

[4] M. M. Bradley and P. J. Lang. Affective norms for english words (anew): Stimuli, instruction manual and affective ratings. Technical report, C-1, The Center for Research in Psychophysiology, University of Florida., Gainesville, FL, USA, 1999.

[5] M. Buccoli, M. Zanoni, A. Sarti, and S. Tubaro. A music search engine based on semantic text-based query. In *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2013.

[6] C.J. Carey and Y. Tang. metric-learn python package. http://github.com/all-umass/metric-learn, 2015.

[7] Y. Chen, M. Rege, M. Dong, and J. Hua. Non-negative matrix factorization for semi-supervised data clustering. *Knowledge and Info. Systems*, 17(3):355–379, 2008.

[8] R. Cowie, G. McKeown, and E. Douglas-Cowie. Tracing emotion: an overview. *International Journal of Synthetic Emotions, Special Issue on Benefits and Limitations of Continuous Representations of Emotions*, pages 1–17, 2012.

[9] G. Fazekas, M. Barthet, and M. Sandler. Novel methods in facilitating audience and performer interaction using the mood conductor framework. *Lecture Notes In Computer Science (LNCS): Sound, Music, and Motion*, 8905:122–147, 2014.

[10] J. Goldberger, G. E. Hinton, S. T. Roweis, and R. Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, 2005.

[11] P. N. Juslin and J. A. Sloboda, editors. *Music and Emotion Theory and Research*. Series in Affective Science. Oxford University Press, 2001.

[12] P. N. Juslin and J. A. Sloboda. *Handbook of Music and Emotion: Theory, Research, Applications*. OUP Oxford, 2011.

[13] J. A. Lee and J. S. Downie. Survey of music information needs, uses, and seeking behaviors: preliminary findings. In *Proc. of the 5th International Society for Music Information Retrieval (ISMIR)*, 2004.

[14] J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability*. Oakland, CA, USA., 1967.

[15] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems 22*, 2009.

[18] A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proc. of the Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[19] J. A. Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161–1178, 1980.

[20] P. Saari, G. Fazekas, T. Eerola, M. Barthet, O. Lartillot, and M. Sandler. Genre-adaptive semantic computing and audio-based modelling for music mood annotation. *IEEE Transactions on Affective Computing*, (99):1–1, 2015.

[21] K. R. Scherer. Which emotion can be induced by music? what are the underlying mechanisms? and how can we measure them? *Journal of New Music Research*, 33(5):239–251, 2004.

[22] J. Shi and J. Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[23] R. Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[24] J.F. Thayer. Multiple indicators of affective responses to music. *Dissertation Abst. Int.*, 47(12), 1986.

[25] F. Weninger, F. Eyben, and B. Schuller. On-line continuous-time music mood regression with deep recurrent neural networks. In *proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[26] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. *Advances in neural information processing systems*, 15:505–512, 2003.

[27] Y.-H. Yang and H. H. Chen. *Music Emotion Recognition*. CRC Press, 2011.

A LATENT REPRESENTATION OF USERS, SESSIONS, AND SONGS FOR LISTENING BEHAVIOR ANALYSIS

Chia-Hao Chung
National Taiwan University
b99505003@ntu.edu.tw

Jing-Kai Lou
KKBOX Inc.
kaelou@kkbox.com

Homer Chen
National Taiwan University
homer@ntu.edu.tw

ABSTRACT

Understanding user listening behaviors is important to the personalization of music recommendation. In this paper, we present an approach that discovers user behavior from a large-scale, real-world listening record. The proposed approach generates a latent representation of users, listening sessions, and songs, where each of these objects is represented as a point in the multi-dimensional latent space. Since the distance between two points is an indication of the similarity of the two corresponding objects, it becomes extremely simple to evaluate the similarity between songs or the matching of songs with the user preference. By exploiting this feature, we provide a two-dimensional user behavior analysis framework for music recommendation. Exploring the relationships between user preference and the contextual or temporal information in the session data through this framework significantly facilitates personalized music recommendation. We provide experimental results to illustrate the strengths of the proposed approach for user behavior analysis.

1. INTRODUCTION

Analyzing the listening behavior of users involves identifying and representing the music preferences of users. However, the music preference of a user is dynamic and varies with the listening context [1], such as time, location, user's mood, etc. How to take the contextual information into consideration for music recommendation is an important research issue [1]–[3]. In this work, we focus on the analysis of dynamic listening behavior and use the obtained information to personalize music recommendation.

Music and user preference are commonly represented using a taxonomy of musical genres, such as hip-hop, rock, jazz, etc. In this approach, the preference of a user is represented as a probability distribution of the genres that the user listened to. Although simple and easy to implement, the main drawback of this approach is that there is not a uniform taxonomy for music, making genre identification ambiguous and subjective [4]. In addition, it often lacks the kind of granularity needed to distinguish between songs of the same genre.

© Chia-Hao Chung, Jing-Kai Lou, Homer Chen.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Chia-Hao Chung, Jing-Kai Lou, Homer Chen. "A Latent Representation of Users, Sessions, and Songs for Listening Behavior Analysis", 17th International Society for Music Information Retrieval Conference, 2016.

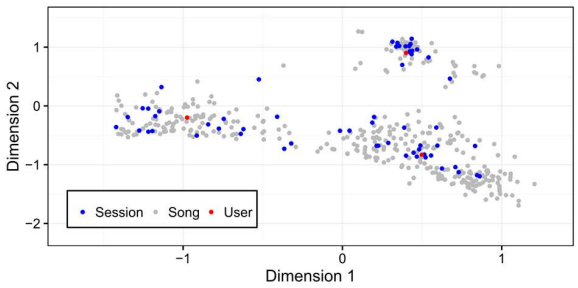


Fig. 1. A two-dimensional latent space representation of three users listening to 403 songs in 60 sessions. Each user, session, or song is represented by a point in the latent space, and each user is surrounded by the songs played by the user and the listening sessions during which the songs are played.

In practice, it is often required to describe the music preference of a user with a fine precision. This requires a good metric to measure the similarity between songs. Therefore, besides the uniformity and granularity requirements, a music preference representation scheme has to provide an effective similarity measurement. Many approaches based on latent representation have been proposed [5], [7] to meet all three requirements. These approaches represent songs and user preferences by one single scheme. The unified representation, which is a multi-dimensional vector, is learned from a listening record or a rating record. Each dimension of the vector represents a latent feature of songs and user preferences. Therefore, each song or user is an object represented by a vector in a latent space, making the evaluation of similarity between songs or the matching between songs and users a simple matter of distance measure between vectors.

The music preference of a user may change with the listening session [3], [8]. A listening session here refers to a sequence of songs (and the associated time code) which a user continuously listened to. It contains information related to the listening experience of the user. To account for the dynamic nature of music preference, we incorporate the notion of session into the learning stage of a latent representation. In our approach, each session of the listening record of a user is also represented as an object in the latent space. The contextual information, such as the time of day and the device used for music listening, associated with each session enables the analysis of user preference at a fine level.

In addition, each user, session, or song can be plotted as a point in a two-dimensional latent space, as illustrated in Fig. 1. Clearly, this provides an intuitive way to visualize or analyze the relationship between songs and user preferences. We exploit this feature of latent space for listening behavior analysis.

To the best of our knowledge, this paper is among the first that introduce the notion of session to the representation learning for music recommendation and proposes an approach that generates the latent representation of users, sessions, and songs (Sections 3 and 4). The proposed latent representation is a powerful basis for visual analysis of user preference in a two-dimensional space and enables the discovery of a user's listening behavior that would otherwise be difficult to do with conventional representations (Section 5). In addition, we propose an effective method to evaluate the performance of a latent representation for listening behavior discovery (Section 6).

2. REVIEW

Statistical approaches that analyze the dynamic nature of music preference have been reported in the literature. Herrera *et al.* [2] adopted a circular statistic method to identify the temporal pattern of music listening. Zheleva *et al.* [3] proposed a session-based probabilistic graphical model to characterize music preference and showed the usefulness of session to capture the dynamic preference of a user. The importance of these two pieces of work is that they show users' listening behaviors (patterns) can be discovered and used to predict future user behaviors.

Approaches that generate a latent representation of users and songs for music recommendation have been reported. Dror *et al.* [5] adopted a matrix factorization method to characterize each user or song as a low-dimensional latent vector and to approximate the user preference (i.e. a rating) as the inner product of a user vector and a song vector. The temporal dynamics of music preference and the taxonomy of musical genres are considered jointly to improve the performance of music recommendation. Moore *et al.* [7] proposed a dynamic probabilistic embedding method to generate a representation of users and songs for music preference analysis. Each user or song is represented as a point in a two-dimensional space, and the position of each point is allowed to gradually change over time. The trajectory of a point shows the long-term variation in music preference. Recently, Chen *et al.* [9] introduced a network embedding method to enhance music recommendation. The social relationship between users is exploited to learn a latent representation of social listening, which is fed to a factorization machine to improve the performance of music recommendation.

3. LATENT REPRESENTATION LEARNING

In our latent space approach, a network that describes the relationship between users, sessions, and songs stored in a listening record is first constructed, then a network embedding method is applied to learn the latent representation from the network. The details are described in this section.

3.1 Network Construction

The basic idea to construct a network that describes the relationship between users and songs is to consider each user or song as an object in the network and connect each user with the songs the user listened to [9]. To account for the dynamic nature of music preference, we further incorporate listening sessions into the network construction and consider each user, session, or song as an object in the network. A user is connected with all sessions of the user, and a session is connected with all songs appearing in the session. This makes the network capture the dynamic music preferences of users.

3.2 Network Embedding

A network embedding method aims at learning the latent representation of objects in a network. Such representation captures the relationship between the objects in the network. Objects having a similar neighborhood in the network are represented by similar vectors. In our approach, the DeepWalk algorithm [10] is applied to learn the latent representation. This algorithm consists of a random walk procedure and an update procedure.

A network consists of a set of vertices representing the objects and a set of edges connecting related vertices. The random walk procedure uniformly samples a random vertex as the root of a random walk, and then uniformly samples a random vertex from the neighbors of the current vertex as the next vertex until the maximum walking length L is reached. The procedure repeats until each vertex serves as the root of R random walks, where R is a predetermined number. The total number of random walks generated in this procedure is thus equal to the number of vertices in the network multiplied by R. The vertices visited in a random walk are processed next.

The latent representation of each vertex of is initially a d-vector of random variables, where d denotes the dimension of the latent space. The update procedure [11], [12] takes the random walks one by one as input and progressively refines the latent representation of objects in two steps. The first step creates a probability formula for each vertex in a random walk, starting from the first one. Specifically, for a vertex v_i in a random walk and a neighborhood window w, the conditional probability that the set of vertices $\{v_{i-w}, \ldots, v_{i-1}\}$ appears in the backward window of v_i and $\{v_{i+1}, \ldots, v_{i+w}\}$ appears in the forward window of v_i is expressed as

$$P(\{v_{i-w},\cdots,v_{i-1},v_{i+1},\cdots,v_{i+w}\} \mid v_i),$$

which is called the co-occurrence probability because it indicates the likelihood that these two sets of vertices are in the neighborhood of v_i in a random walk. If the order of the vertices in each window is ignored, the co-occurrence probability can be rewritten as

$$P(\{v_{i-w},\cdots,v_{i-1},v_{i+1},\cdots,v_{i+w}\} \mid v_i) = \prod_{j=i-w, j\neq i}^{i+w} P(v_j \mid v_i). \quad (1)$$

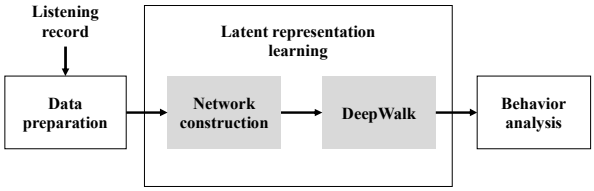


Fig. 2. Overview of our approach.

In the second step, the vector of v_i is optimized by maximizing $P(v_j \mid v_i)$. To enhance computational efficiency, a binary tree with all vertices of the network as the leaves is constructed to convert the maximization to a tree traversal process [13]. A path of the tree traversal is specified by a sequence of nodes $\{b_1, b_2, \ldots, b_k\}$ in the binary tree, where k is the length of the path, b_1 is the root of the binary tree and b_k is the leaf node representing v_j. Then the conditional probability $P(v_j \mid v_i)$ can be rewritten as

$$P(v_j \mid v_i) = \prod_{l=1}^{k-1} P(b_{l+1} \mid v_i, b_l). \quad (2)$$

Each conditional probability $P(b_{l+1} \mid v_i, b_l)$ is modeled by a logistic function and can be rewritten as

$$P(b_{l+1} \mid v_i, b_l) = 1/(1 + e^{-\Phi(v_i) \cdot \Psi(b_l)}), \quad (3)$$

where $\Phi(v_i)$ maps v_i to its vector, and $\Psi(b_l)$ maps b_l to its vector. Then, a stochastic gradient descent method [14] and a back-propagation algorithm [15] are applied to optimize Φ and Ψ. The update procedure repeats until the optimization for each vertex in each random walk is processed. The optimized Φ is the latent representation of each vertex in the network.

4. OUR APPROACH

The three basic steps of the proposed approach are shown in Fig. 2. The first step involves the preparation of a listening record, the second step involves the construction of a network and the learning of a latent representation from the network, and the third step involves the analysis of user behavior. The first and second steps are described in this section, and the third step is described in Section 5.

4.1 Preparation

We obtain a listening record of one hundred thousand users from a leading online music service provider [18] and use it in this work. The listening record contains every listening event of these users from January 1, 2015 to June 30, 2015, and each event contains seven fields: timestamp, user, session, listening device, song title, artist(s) of the song, and music tag(s) of the song. All users are anonymized to maintain privacy.

A session, which indicates the listening experience of a user, is defined as a sequence of events of the user with the following constraints: The gap between any two neighboring events in a session is shorter than 10 minutes, and the listening device stays the same in a session.

The music tags are used for visual analysis in Section 5, showing either genre or language information of a song. A genre tag indicates the musical style of the song, and a language tag indicates the language of the song. We also obtain the popularity of each song by taking the logarithm of its playcount for the visual analysis.

	#events	#users	#songs
Training set	33,790,690	33,292	441,796
Testing set	8,797,016	19,831	219,377

Table. 1. Data statistics.

The listening record is split in to training set and testing set. We adopt the real-life split strategy [16] and split the listening record into two parts: before and after 00:00:00, June 1, 2015. The last 80 sessions of each user in the first part are selected as training data, and the first 20 sessions of each user in the second part are selected as the testing data. Users with insufficient sessions or sessions with less than 5 songs in either set are discarded. In addition, because a cold start problem [16] may occur if songs in the testing set are not in the training set, we discard such events from the testing set. Table 1 shows the data statistics. The testing set is used for performance evaluation in Section 6.

4.2 Representation Learning

Using the training set, a user-session-song network is constructed as described in Section 3.1, and the DeepWalk algorithm is applied to generate the latent representation of users, sessions, and songs from the network. We set the parameters of the learning algorithm as the following: The length of a random walk L is 40, the number of random walks starting from a vertex R is 20, and the window size w is 6. A two-dimensional latent representation is generated for visual analysis (Section 5), and a 128-dimensional latent representation is generated for performance evaluation (Section 6).

5. VISUALIZATION AND ANALYSIS

A two-dimension latent space provides an intuitive way to visualize the relationship between songs and user preferences. We perform a visual analysis of the general trend of listening preference and then the individual listening behavior in such space. The details of these operations are described in this section.

5.1 General Trend of Listening Preference

We can get an idea of the general trend of listening preference with respect to the song properties (the language of songs, the musical genre, and the song popularity) by examining the distribution of songs in the latent space. The two-dimensional latent space learned from the training set is plotted in Fig. 3. In each plot, 25,000 songs are plotted, and each song is marked with a specific color according to the property of the song.

In Fig. 3 (a), songs are colored according to the language of songs. There are songs in six different languages in our dataset where Western songs take a large proportion. We can see that songs of the same language are close and form a cluster in the latent space. This suggests that the user listening preference is strongly related to the song language and that a user tends to listen to songs of the languages that the user is familiar with.

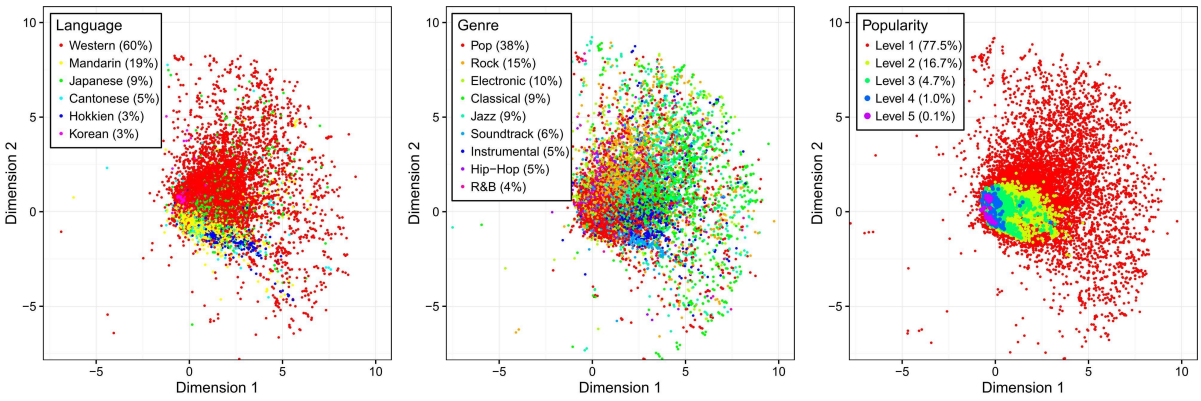


Fig. 3. The two-dimensional latent space learned from the training set is shown in three plots. In each plot, 25,000 songs are randomly selected, and each song is specified by a point and marked with a specific color according to the property of the song. The proportion of songs with each property is showed in the legend. (a) Each song is marked according to the song language. (b) Each song is marked according to the genre, and (c) Each song is marked according to the popularity (Level 5 indicates the most popular songs), and songs with high popularity are overlaid over those with low popularity.

We can also see that some clusters are located closely. For example, Mandarin songs and Hokkien songs are located closely or even mixed together. This indicates that a part of users who listen to Mandarin songs are likely to listen to Hokkien songs as well.

In Fig. 3 (b), songs are colored according to the musical genre. Clearly, songs of the same genre are located in the same area in the latent space. The similarity between genres is reflected on their distance. For example, Jazz songs are close to classical songs, and electronic songs are close to hip-hop songs. Combining Figs. 3 (a) and (b), we can see that Westerns songs contain many genres, such as hip-hop, rock, electronic, jazz, and classical songs, and Mandarin songs are mostly pop songs.

In Fig. 3 (c), songs are colored according to the popularity. The popularity of each song is obtained by taking the logarithm of the playcount of the song. For easy visualization, all songs are divided in to five levels in terms of popularity. Clearly, the most popular songs are near the origin of the latent space, and unpopular songs are far from the origin. It indicates that most users listen to the most popular songs, a typical long tail phenomenon of music listening [17]. Combining Figs 3 (a), (b), and (c), we can see that Western, Mandarin and Korean pop songs are more popular than other songs in our dataset.

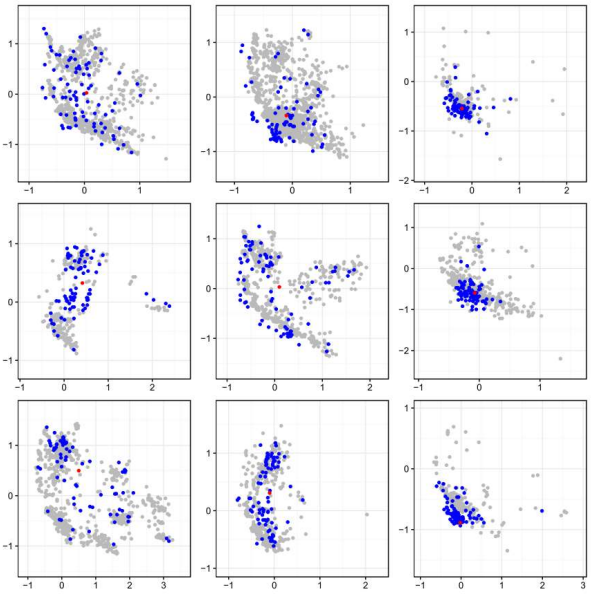


Fig. 4. Listening behavior analysis for nine example users. The sessions and songs associated with a user are plotted in each plot, where a red point represents a user, a blue point represents a session, and a gray point represents a song.

5.2 Individual Listening Behavior

The individual behavior is analyzed through the distribution of sessions and songs associated with a user in latent space. A session is close to the songs that appear in the session, and sessions form a cluster if they contain similar songs. Fig. 4 shows the analyses for nine example users. In each plot, the sessions and songs associated with a user are plotted.

One important discovery is that users can be divided into two types, one with a single preference and the other with dynamic preference. For some users, the sessions are mostly located in one small area in the latent space. This means that the users listens to the same songs most of the time and hence belong to the first type. For other users, the sessions have a wide distribution and form several clusters. This indicates that the users belong to the second type.

In order to analyze the dynamic preference of an individual, we distinguish sessions by the context information (the device used for listening and the time of day). In Fig. 5, we color each session according to the device used for listening, and we can clearly see that the sessions form clusters according to the listening device. This indicates there is a strong relevance between the music preference and the listening device. For example, a

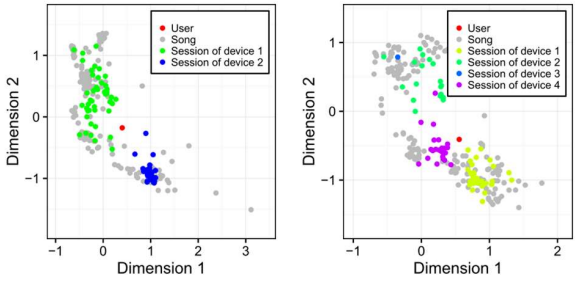


Fig. 5. The sessions are distinguished by the device used for music listening. Each plot illustrates a user whose music preference is related to the listening device.

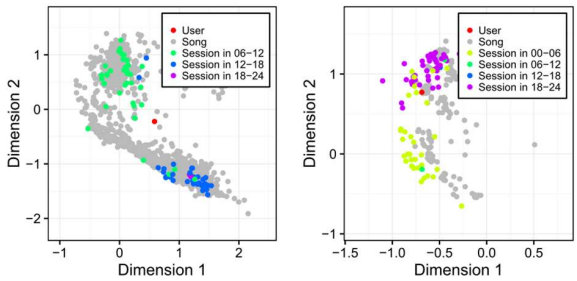


Fig. 6. Each plot shows a user whose music preference is related to the time for music listening. The sessions tend to form clusters according to the time (hour of day).

user may listen to rock songs through computer and listen to pop songs through mobile phone. This kind of listening behavior can be found on many users in our dataset. In Fig. 6, we color each session according to the time of day, and each plot shows a user whose music preference is related to the time for music listening. However, this kind of listening behavior is not easy to be observed on users. Probably it is because the relationship between time and music preference is too complex to be explained.

An interactive system for individual behavior analysis can be designed based on the latent space. For example, when we select a session, the system would highlight the songs that appear in the session so that we can further discover the user preference in each session.

6. PERFORMANCE EVALUATOIN

Two experiments are designed to evaluate the effectiveness of a latent representation for listening behavior prediction. The first one involves retrieving similar songs (i.e. songs that appear in the same session), and the second one involves recommending songs that match a user's preference in a session. Each experiment is considered a retrieval problem whose goal is to retrieve songs relevant to a query object (user, session, or song) as many as possible. Specifically, a query object is selected according to a given testing session, and songs that appear in the testing session are considered relevant to the query object. For a query object, its k nearest neighboring songs in the latent space are retrieved, and the performance is evaluated in terms of how well the retrieved songs match the relevant songs. In each experiment, 200,000 sessions in the testing set are randomly selected for testing. Because the average length of the testing sessions is 20 songs, k (the number of retrieved songs) is set to $\{10, 15, \cdots, 50\}$.

Two standard evaluation metrics for retrieval problem are applied here: Recall and precision:

$$Recal \quad \frac{|S_t \cap S_r|}{|S_t|} \tag{4}$$

$$Precision = \frac{|S_t \cap S_r|}{|S_r|} \tag{5}$$

where S_t is the set of songs that appear in a testing session and S_r is the set of retrieved songs. A high recall means that most of relevant songs are retrieved, and a high precision means that most of the retrieved songs are relevant songs. The average recall and precision for all testing sessions are reported.

The performances of the following four methods are reported for comparison.

- **Random**: k songs are randomly selected from the dataset as the retrieved songs.
- **Popularity**: The popularity of each song is obtained by calculating its playcount in the training set, and the k most popular songs are considered the retrieved songs.
- **Matrix factorization (MF)** [6]: The vector $\boldsymbol{p}_u$ for user u and the vector $\boldsymbol{q}_i$ for song i are learned by solving the optimization problem

$$\min_{q_*,p_*} \sum_{u,i}(c_{ui} - \boldsymbol{q}_i^T \boldsymbol{p}_u)^2 + \lambda(\|\boldsymbol{q}_i\|^2 + \|\boldsymbol{p}_u\|^2), \tag{6}$$

  where $c_{ui} = 1 + \log(1 + y_{ui})$ is the confidence value of the playcount y_{ui} of user u for song i, and λ is a regularization parameter. Songs are retrieved according to the inner product between vectors.
- **User-song network (U-S)**: A simplified network, where a user directly connects to all songs the user listened to, is constructed. The vectors for users and songs are learned from the network using the DeepWalk algorithm. Songs are retrieved according to the Euclidean distance between vectors.

For fair comparison, the dimension of the latent representations (vectors) learned by MF, U-S, and the proposed method is fixed to 128.

6.1 Experiment for Retrieval of Similar Songs

This experiment evaluates the ability of the representation of songs to capture the similarity between songs. Because songs in a session usually have similar properties, this experiment is to find songs that appear in the same testing session. Specifically, the first song in each testing session is selected as a query object to retrieve its k nearest neighbor songs, and the remaining songs in the testing session are considered relevant to the query object.

The performances of various methods are compared in in Fig. 7. We can see that the proposed approach outperforms MF and U-S, showing the effectiveness of adding session objects into the learning stage of a latent representation. We can also see that the popularity-based method works slightly better than the random method, showing that many users tend to listen to the popular songs. This is consistent with our observation (i.e. the long tail phenomenon) in Section 5.1.

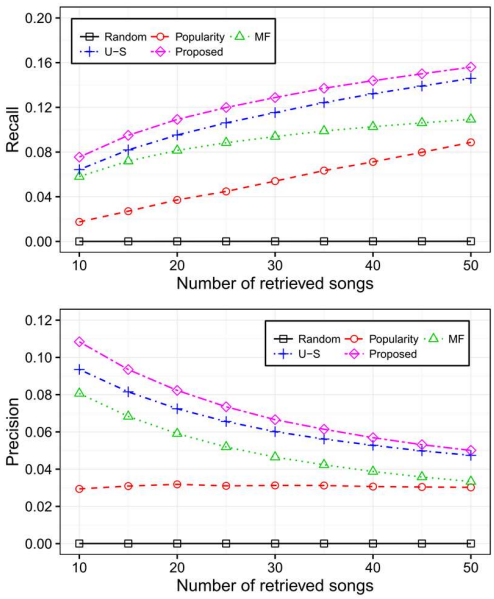


Fig. 7. Performance comparison of various methods for the retrieval of similar songs.

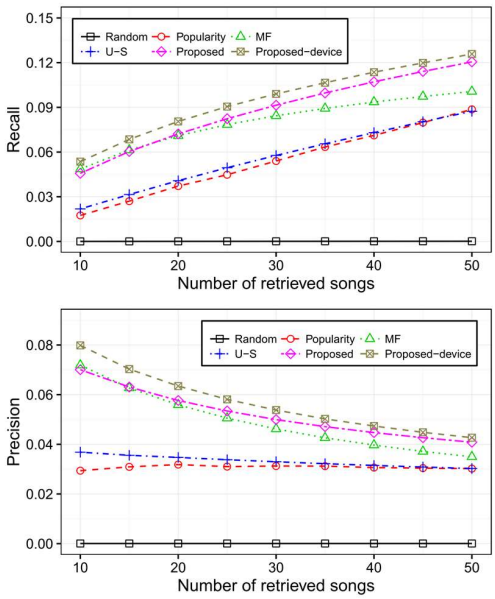


Fig. 8. Performance comparison of various methods for music recommendation. Proposed-device indicates that the device information of each testing session is exploited.

6.2 Experiment for Music Recommendation

This experiment evaluates the effectiveness of a latent representation to recommend songs to a user in a testing session. The vector for the user is used to retrieve its k nearest neighbor songs, and the songs in the testing session are considered relevant songs.

As discussed in Section 5.2, a user may have dynamic preference, so only considering the user vector for every session of the user is not enough. With our observation that there are many users whose preferences are related to the listening device, we use the session vector to recommend songs. For each testing session, a reference session that belongs to the same user of the testing session and that is through the same listening device is selected as a query object, and the vector for the reference session is used to retrieve its k nearest neighbor songs.

The performance comparison is shown in Fig. 8. We can see that if the user vector is used, the proposed approach outperforms MF when k is higher than 20. If the session vector is considered, the performance of proposed approach is significantly improved. It shows the benefit of exploiting the contextual information (device information) to capture the dynamic music preference of a user.

7. DISCUSSION

Besides the network embedding method adopted in this work, factorization is another approach that can be applied to generate the latent representation of objects. A factorization method approximates the interactions (e.g. ratings or counts) between objects as the inner product of the vectors representing the objects. In contrast, as we can see in Eqn. (3), the network embedding method maps all objects into the same latent space (Φ) and makes similar objects close to each other in the latent space. This enables us to visually analyze the relationship between objects in the latent space. Moreover, the network embedding method learns the relationship between objects that do not have explicit links between them, such as two users who listen to the same songs. Such relationship cannot be learned by the factorization method. A comprehensive comparison between the network embedding and factorization methods (e.g. user-session-song factorization) is an interesting topic for future work.

8. CONCLUSION

Knowledge of the behavior of music listeners is important to music recommendation. In this paper, we have described an approach to address this issue. The proposed approach generates the latent representation of users, sessions, and songs from a listening record. Such representation makes the relationship between these objects easy to analyze. We have performed a visual analysis of user behavior and preference in a two-dimensional latent space and illustrated the strengths of the proposed approach by comparing its music recommendation and retrieval performances with various methods. We have also shown that the information obtained from the two-dimensional analysis is useful for personalized music recommendation. The contextual information associated with each session enables both user preference analysis and music recommendation at a fine level.

9. REFERENCES

[1] M. Kaminskas and F. Ricci, "Contextual music information retrieval and recommendation: State of the art and challenges," *Comput. Sci. Review*, vol. 6, no. 2, pp. 89–119, 2012.

[2] P. Herrera, Z. Resa, and M. Sordo, “Rocking around the clock eight days a week: An exploration of temporal patterns of music listening,” in *1st Workshop Music Recommendation Discovery*, 2010.

[3] E. Zheleva, J. Guiver, E. M. Rodrigues, and N. Milic-Frayling, “Statistical models of music-listening sessions in social media,” in *Proc. 19th ACM Int. Conf. World Wide Web*, pp. 1019–1028, 2010.

[4] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Tran. Speech Audio Process.,* vol. 10, no. 5, pp. 293–302, 2002.

[5] G. Dror, N. Koenigstein, and Y. Koren, “Yahoo! music recommendations: Modeling music ratings with temporal dynamics and item taxonomy,” in *Proc. Fifth ACM Int. Conf. Recommender Syst.,* pp. 165–172, 2011.

[6] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Proc. 8th IEEE Int. Conf. Data Mining*, pp. 263–272, 2008.

[7] J. L. Moore, S. Chen, T. Joachims, and D. Turnbull, “Taste over time: The temporal dynamics of user preferences,” in *Proc. 14th Int. Soc. Music Inform. Retrieval Conf.*, pp. 401–406, 2013

[8] O. Carlsson, *Cluster User Music Sessions*, Master of Science Thesis, Department of Computer Science and Engineering, Chalmers University of Technology, Gothenburg, Sweden, 2014.

[9] C.-M. Chen, P.-C. Chien, Y.-C. Lin, M. F. Tsai, and Y.-H. Yang, “Exploiting Latent Social Listening Representations for Music Recommendations,” in *Proc Ninth ACM Int. Conf. Recommender Syst. Poster*, 2015.

[10] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” in *Proc. 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pp. 701–710, 2014.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv: 1301.3781*, 2013.

[12] T. Mikolov, I. Sutskever, K. Chen,G. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Proc. Advances Neural Inf. Process. Syst.*, pp. 3111–3119, 2013.

[13] F. Morin and Y. Bengio, “Hierarchical probabilistic neural network language model,” in *Proc. Int. Workshop Artificial Intell. Stat.*, pp. 246–252, 2005.

[14] L. Bottou, “Large-scale machine learning with stochastic gradient descent,” in *Proc. 19th Int. Conf. Comput. Stat.*, pp. 177–186, 2010.

[15] Y. LeCun, D. Touresky, and G. Hinton, “A theoretical framework for back-propagation,” in *Proc. Connectionist Models Summer School*, pp. 21–28, 1988.

[16] S.-Y. Chou, Y.-H. Yang, and Y.C. Lin, “Evaluating music recommendation in a real-world setting: On data splitting and evaluation metrics,” in *Proc. IEEE Int. Conf. Multimedia Expo*, pp. 1–6, 2015

[17] O. Celma, *Music Recommendation and Discovery*, Springer Berlin Heidelberg, 2010.

[18] KKBOX Inc., https://www.kkbox.com.

A METHODOLOGY FOR QUALITY ASSESSMENT IN COLLABORATIVE SCORE LIBRARIES

Vincent Besson[1] **Marco Gurrieri**[1] **Philippe Rigaux**[2]
Alice Tacaille[3] **Virginie Thion**[4]
[1] CESR, Univ. Tours, France
[2] CEDRIC/CNAM, Paris, France
[3] IReMus, Sorbonne Universités, Paris, France
[3] IRISA, Univ. Rennes 1, Lannion, France
{vincent.besson,marco.gurrieri}@univ-tours.fr, philippe.rigaux@cnam.fr,
alice.tacaille@paris-sorbonne.fr, virginie.thion@irisa.fr

ABSTRACT

We examine quality issues raised by the development of XML-based Digital Score Libraries. Based on the authors' practical experience, the paper exposes the quality shortcomings inherent to the complexity of music encoding, and the lack of support from state-of-the-art formats. We also identify the various facets of the "quality" concept with respect to usages and motivations. We finally propose a general methodology to introduce quality management as a first-level concern in the management of score collections.

1. INTRODUCTION

There is a growing availability of music scores in digital format, made possible by the combination of two factors: mature, easy-to-use music editors, including open-source ones like MuseScore [10], and sophisticated music notation encodings. Leading formats today are those which rely on XML to represent music notation as structured documents. MusicXML [7] is probably the most widespread one, due to its acceptance by major engraver softwares (Finale, Sibelius, and MuseScore) as an exchange format. The MEI initiative [13, 9], inspired by the TEI, attempts to address the needs of scholars and music analysts with an extensible format [8]. Recently, the launch of the W3C Music Notation Community Group [15] confirms that the field tends towards its maturity, with the promise to build and preserve large collections of scores encoded with robust and well-established standards. We are therefore facing emerging needs regarding the storage, organization and access to potentially very large Digital Libraries of Scores (DSL). It turns out that building such a DSL, particularly when the acquisition process is collaborative in nature, gives rise to severe quality issues. In short, we are likely to face problems related to *validity* (measure durations, voices and parts synchronization), *consistency* (heterogeneous notations, high variability in the precision of metadata, undetermined or inconsistent editorial rules), *completeness* (missing notes, directives, ornamentation, slurs or ties), and *accuracy* (music, lyrics).

There are many reasons for this situation. First, encoding formats have changed a lot during the last decades. We successively went through HumDrum and MIDI to finally come up with modern XML formats such as MusicXML and MEI [14]. A lot of legacy collections have been converted from one encoding to the other, losing information along the way. Given the cost and time to edit scores, incorporating these collections in a modern repository is a strong temptation, but requires to accept, measure, and keep track of their quality shortcomings.

Second, the flexibility of music notation is such that it is extremely difficult to express and check quality constraints on the representation. Many of the formats we are aware of for instance do not impose that the sequence of events in a measure exactly covers the measure duration defined by the metrics. As another example, in polyphonic music, nothing guarantees that the parts share the same metric and same duration. So, even with the most sophisticated encoding, we may obtain a score presentation which does not correspond to a meaningful content (the definition of which is context-dependent), and will lead to an incorrect layout (if not a crash) with one of the possible renderers.

Third, scores are being produced by individuals and institutions with highly variables motivations and skills. By "motivation", we denote here the purpose of creating and editing a score in digital format. A first one is obviously the production of material for performers, with various levels of demands. Some users may content themselves with schematic notation of simple songs, whereas others will aim at professional editing with high quality standards. The focus here is on rendering, readability and manageability of the score sheets in performance situation. Another category of users (with, probably, some overlap) are scientific editors, whose purpose is rather an accurate and


© Vincent Besson, Marco Gurrieri, Philippe Rigaux, Alice Tacaille, Virginie Thion. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Vincent Besson, Marco Gurrieri, Philippe Rigaux, Alice Tacaille, Virginie Thion. "A METHODOLOGY FOR QUALITY ASSESSMENT IN COLLABORATIVE SCORE LIBRARIES", 17th International Society for Music Information Retrieval Conference, 2016.

long-term preservation of the source content (including variants and composer's annotations). The focus will be put on completeness: all variants are represented, editor's corrections are fully documented, links are provided to other resources if relevant, and collections are constrained by carefully crafted editorial rules. Overall, the quality of such projects is estimated by the ability of a document to convey as respectfully as possible the composer's intent as it can be perceived through the available sources. Librarians are particularly interested by the searchability of their collections, with rich annotations linked to taxonomies [12]. We finally mention analysts, teachers and musicologists: their focus is put on the core music material, minoring rendering concerns. In such a context, part of the content may be missing without harm; accuracy, accessibility and clarity of the features investigated by the analytic process are the main quality factors.

Finally, even with modern editors, qualified authors, and strong guidelines, mistakes are unavoidable. Editing music is a creative process, sometimes akin to a free drawing of some graphic features whose interpretation is beyond the software constraint checking capacities. A same result may also be achieved with different options (e.g., the layer feature of Finale), sometimes yielding a weird and convoluted encoding, with unpredictable rendering when submitted to another renderer.

The authors of the present paper are in charge of the production, maintenance and dissemination of digital libraries of scores encoded in XML (mostly, MEI). NEUMA is an open repository of scores in various formats, managed by the IReMus [1] , and publicly accessible at `http://neuma.huma-nm.fr`. The CESR [2] publishes rare collections of Renaissance music for scholars and musicians (see, e.g., the "Lost voices" project, `http://digitalduchemin.org`). Both institutions have been confronted with the need to address issues related to the consistent production of high-level quality corpora, and had to deal with the poor support offered by existing tools. The current, ad-hoc, solution adopted so far takes the form of editorial rules. The approach is clearly unsatisfying and unable to solve the above challenges. Even though we assume that the scores are edited by experts keen to comply with the recommendations, nothing guarantees that they are not misinterpreted, or that the guidelines indeed result in a satisfying encoding. Moreover, rules that are not backed up by automatic validation safeguards are clearly non applicable in a collaborative context where un-controlled users are invited to contribute to the collections.

In the rest of the paper we position our work with respect to the field of quality management in databases and Digital Libraries (Section 2) and propose a general methodology to cope with quality issues in the specific area of digital score management. Section 3 exposes a quality management model. We apply this model to represent data quality metrics, usages and goals, as explained in Section 4, which includes our initial taxonomy of data quality metrics for score libraries. Finally, Section 5 recalls the contributions and outlines our perspectives.

[1] *Institut de Recherche en Musicologie*, `http://iremus.cnrs.fr`.

[2] *Centre d'Etudes Supérieures de la Renaissance*, `http://cesr.univ-tours.fr`.

2. QUALITY MANAGEMENT IN DATABASES AND DIGITAL LIBRARIES

Much published data suffers from endemic quality problems. It is now well-recognized that these problems may lead to severe consequences, and that managing the quality of data conditions the success of most existing information systems [5]. Data quality is a complex concept, which embraces different semantics depending on the context [11]. It is described through a set of quality *dimensions* aiming to categorize criteria of interest. Classical quality dimensions are *completeness* (the degree to which needed information is present in the collection), *accuracy* (the degree to which data are correct), *consistency* (the degree to which data respect integrity constraints and business rules) and *freshness* (the degree to which data are up-to-date). Data quality over a dimension is measured according to a set of *metrics* that allow a quantitative definition and evaluation of the dimension. Examples of metrics are "the number of missing meta-data" for the evaluation of the *completeness*, and "the number of conflicting duplicates" for *consistency*. These are simple examples but the literature proposes a large range of dimensions and metrics, conceptualized in quality models [4]. Of course, not all the existing dimensions and metrics may be used for evaluating data quality in a given operational context. An important property concerning data quality is that it is defined according to *fitness for use* of data, meaning that quality measurement involves dimensions and metrics that are relevant to a given user for a given *usage*. User u_1 may be concerned by some quality metrics for a specific usage, by some other metrics for another one, and they can be completely different than those needed by user u_2.

The literature proposes general methodologies for managing data quality [3]. We focus here on its *assessment*. Roughly speaking, each assessment methodology includes a *quality definition* stage and a *quality measurement* one. In the first stage, the quality definition consists in eliciting data quality requirements. Concretely, this means choosing quality dimensions and metrics of interest, and eventually thresholds associated with. Because data quality is *fitness for use* (depends on the context), defining data quality is not trivial. Dedicated methodological guidelines may be followed like the *Goal Question Metric* [2], which proposes to define quality metrics according to a top-down analysis of quality requirements. For each user (or each user role) and for each of his/her usages of data, conceptual *goals* are identified. Goals specify the intent of measurement according to a usage of data. Each goal is then refined into a set of operational *quality questions*. Each such question is itself expressed in terms of a set of quantitative quality metrics with possible associated thresholds (expected values). Measuring the quality metrics enables to (partly) answer to the quality questions, and consequently enables to decide whether data satisfy the requirements for the given goal (and each usage by extension).

Data quality methodologies are designed at a generic level, leading to difficulties for their implementation in a specific context (operational context and available information system and data). Additional context-dependent quality methodologies are then needed. We propose such a methodology for an explicit and systematic data quality assessment in DSL. To our knowledge, such a methodology has never been proposed in the MIR literature so far.

3. OUR QUALITY MANAGEMENT MODEL

We assume a very general organization of a DSL, where atomic objects are *scores*, organized in *collections*. We further assume that scores are encoded as structured documents (typically in MusicXML or MEI) that supply a fine-grained representation of all their structural, content, and rendering aspects.

The main components of the model are (i) modelization of metrics at the score level and collection levels, and of their relationships, (ii) definition of usages and goals, expressed with respect to these metrics, and (iii) computation of quality metrics. We present these concepts in order.

3.1 Quality schema

The initial step to address quality issues is to determine the set of relevant indicators, or *metrics*, that support the quality evaluation, and how they are related to each other.

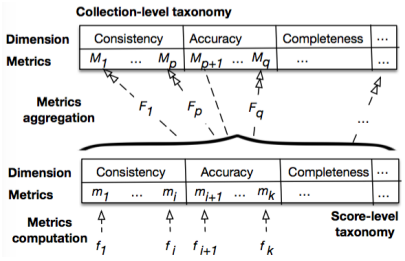


Figure 1. Quality schema: score-level and collection-level metrics

In our context, we consider *score-level metrics*, computed from individual scores, and *collection-level metrics*, essentially computed by aggregation from the score level. We use lowercase/uppercase symbols (e.g., m or M to specifically represent, resp., score-level and collection-level elements (metrics, values, or functions), and small capitals (e.g., M) when they do not need to be distinguished. We denote by $\mathcal{M}_{sc}$ the set of score-level quality metrics, $\mathcal{M}_{coll}$ the set collection-level metrics, and as $\mathcal{M}$ their union $\mathcal{M}_{sc} \cup \mathcal{M}_{coll}$.

Metrics are clustered in *quality dimensions*. For simplification reasons, we suppose that (i) a metric belongs to exactly one quality dimension and that (ii) each metric is relevant for every score/collection of the library (the model can easily be extended if these restrictions are too strong). For each metric $m \in \mathcal{M}$, we denote by $dom(m)$ the domain of the metric.

Each DSL has therefore to determine a two-levels organization of dimensions and metrics that constitutes the *quality schema*. Fig. 1 shows its general form. The value of each (score-level) metric m_i is computed from an atomic object (a score) by some function f_i. The domain of a metric can be a Boolean (*"the tempo is/is not missing"*), an integer (*"n measures are complete"*), a rational (*"position is given for x notes out of y"*), etc. In the case of numeric domains, for convenience, we map each value to a predefined scale $\mathcal{S}$ of the form $\{very\_poor(1), poor(2), borderline(3), good(4), very\_good(5), not\_relevant(\perp)\}$ easily adaptable if needed.

The value of a (collection-level) metric M_j is obtained by an aggregation function F_j which operates over the score-level metric vectors. As an illustration, imagine that we aim at representing the syntactic consistency M_s of a collection, defined as a standard variation from the following score-level values: presence of bars m_b, presence of directives m_d, presence of ornamentation m_o. Then the aggregation function F_s takes as input a set of triplets (v_b, v_d, v_o), which denotes values for m_b, m_d and m_o resp., one for each score of the collection. In the general case, an aggregation function F might take into account the whole set of score-level values.

3.2 Usages, goals, and profiles

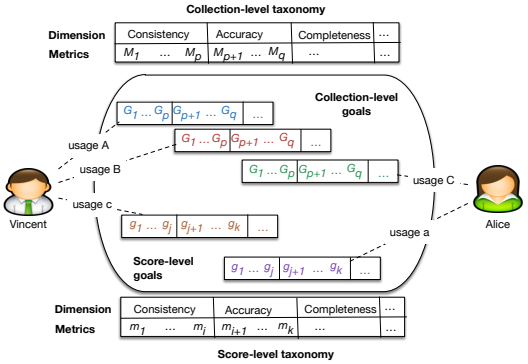


Figure 2. Usages, metric goals, and profiles

Assume now that a quality schema is defined for a DSL managing a set of collections. We are then able to propose to the DSL users a support to express their quality requirements. The main concepts at this level are *usages*, *metric goals* and *profiles* (Fig. 2). A *quality metric goal* assigns an expected quality level for a metric as a threshold $th \in \mathcal{S}$.

A user can express requirements as a set of goals relative to a subset of the metrics, ignoring those which she deems irrelevant in the context of a specific *usage*. For instance a melodic analyst can, *for this usage*, choose to safely ignore the quality metrics that pertain to directives or lyrics. Conversely, for publication purposes, directives and lyrics quality will be required to match a high-quality threshold, whereas analytic annotations are irrelevant. Requirements are therefore usage-related, and take for each usage the form of a set of goals on the metrics specifically relevant for this usage. The specification of such a requirement constitutes what we call a *quality profile*.

From a methodological point of view, each quality profile, including embedded quality dimensions and metrics, is defined by following the *Goal Question Metric* approach (see Section 2). Each metric and dimension appearing in a profile is intrinsically added to the general quality schema. In other words, the set $\mathcal{M}$ of metrics may be defined by union of all metrics appearing in the profiles, which are the relevant metrics identified by the users of the DSL.

3.3 Measurements

A specific module of the DSL is in charge of computing the metrics measurements. As summarized by Fig. 3, this requires to synchronize each score s with the vector $f_1(s), f_2(s), \cdots, f_k(s), \cdots$ representing its quality measurements with respect to the schema.

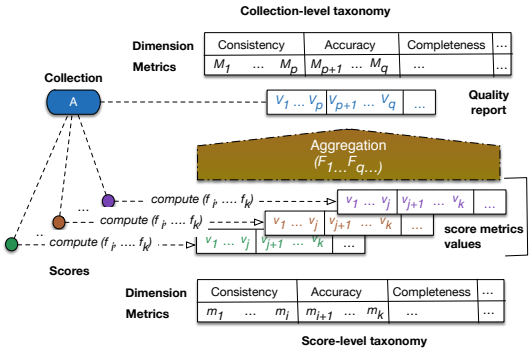


Figure 3. Metrics computation

Likewise, the set of measurements for some collection C must be computed from the quality measurements of the scores that belong to C. One obtains a summary of quality indicators relative to C that we call *quality report*. More formally, the *quality report* of a collection C (resp. of a score s) is a function $QR_C : \mathcal{M}_{coll} \rightarrow \cup_{M \in \mathcal{M}_{coll}} dom(M)$ (resp. $QR_s : \mathcal{M}_{sc} \rightarrow \cup_{m \in \mathcal{M}_{sc}} dom(m)$) that assigns a value for C (resp. score s) to each metric of $\mathcal{M}_{coll}$ (resp. $\mathcal{M}_{sc}$).

Technically, the main issue is to maintain a quality report that faithfully reflects the content of the collection. If the collection is created once for all and never updated, a single batch computation is enough. In general, though, collections are extended, scores are modified, and we have to take those changes into account. At the score level, a trigger seems an appropriate solution: any change of the score results in the execution of the metrics functions $f_1, \cdots, f_k$. Things are more complicated at the collection level. First, a change in some score does necessarily impact the collections metrics, or at least all of them. Second, one has to carefully consider aggregation functions which can be computed incrementally (e.g., *count()*) from those that require a brand new computation from their full input (e.g., *avg()*). Our current implementation adopts a simple recompute-everything strategy, but more sophisticated approaches need to be investigated in the future.

3.4 Matching goals and measurements

Let us now consider how we exploit our two main artifacts: user goals on one side, measurements and reports on the other side. This actually depends on the user's role: *publishers* are responsible for inserting and updating scores and collections, whereas *consumers* can only browse and read. Both a them can define a profile, but for different purposes in the system.

Consumer role and information retrieval. A data consumer may define a profile specifying the expected quality level of data that is needed for a given usage. This profile may be used as a filter by retrieving only appropriate data of the DSL, at the collection or score level, or for recommendation functionalities by suggesting collection or scores of the DSL that respects the profile.

As an example of this filtering facility, one of our DSL supplies a Web front-end interface to browse the collections, some of which exhibit a poor rendering on average, yet are useful for teaching purposes. To limit their access, we can define a usage *browse* with high rendering metric goals, assigned to the *anonymous* user. Anyone accessing to the Web UI without logging in will automatically adopt this default usage and will access only to high-level graphic scores. Connected users can be given access to the teaching collections via another specific usage.

Publisher role and data creation/update. A publisher may define a profile that specifies the needed quality level of data to be achieved before publishing, which may suspend the publishing of the collections or scores that do not respect the profile. This kind of practice goes beyond the control of the quality by the publisher as it also makes it possible to expose a quality certification, for specific usages (profiles) of data available in the platform.

More formally, a quality report QR *satisfies* a profile P iff each quality metric goal of P is satisfied by the values of QR. Given a set (of collections or scores) $\mathcal{S}$ and a profile P, the *filtering* of $\mathcal{S}$ according to a profile P is the set $\{e \in \mathcal{S} \,|\, QR_e \text{ satisfies } P\}$.

4. APPLICATION: GOALS AND USAGES FOR MEI COLLECTION

We interviewed librarians in charge of the two DSL NEUMA and THE LOST VOICES PROJECT (simply denoted by LOST VOICES in the following). Following the *Goal Question Metric* approach (driven by data use cases, see [2] for details), we exhibited a set of relevant quality questions and metrics for data quality management.

4.1 User requirements for NEUMA

Production of scores. NEUMA is an open repository of scores in MusicXML and MEI. Contributions to NEUMA come either from musicologists for on-line publication purposes, with highly ranked quality standards, or from legacy sources (KernScores for instance, converted to XML formats). For on-line editions, a clean and consistent rendering is required, as well as an homogeneous presentation of the scores of a same collection. The level of details

of meta-data should not strongly vary from one score to the other (in a same collection). Legacy collections are incorporated in NEUMA for teaching or research purposes.

Usage of scores. All the scores submitted to the library are processed with a uniform workflow, which includes production of Lilypond scripts for rendering purposes, conversion from/to Music/MEI, extraction of textual and musical features for indexing, etc. Applying this workflow exacerbates the heterogeneity of the input and reveals a lot of discrepancies and variations in the tolerance levels of the various tools that need to access the score representation. Lilypond for instance hardly accepts incomplete measures, whereas this is tolerated and corrected as much as possible by most MusicXML-based editors. A same meta-data field (e.g., authors, title) can be found in many different places in MusicXML (things are better with MEI), resulting in an erratic rendering with any tool other than the initial editor.

Quality concerns. A common concern met by all users is the need to obtain a decent visualization of a score. Here, "decent" means that any user accessing a score in the library should be able to display it with a desktop tool without a strong readability impact. Unfortunately, this turns out to be difficult to achieve. A score created with a specific engraver $E1$ may contain issues, which are tolerated by $E1$ but result in an awful rendering with $E2$. In general, having a valid MusicXML or MEI document does not guarantee that it can correctly be visualized as a score.

If we leave apart the readability problems, specific usages dictate the quality demands and at which level (score or collections) these demands take place. The Bach chorales for instance are supplied by an external source with accurate music representation, but missing lyrics. This obviously keeps them from any use in a performance perspective, but preserves their interest for music analysis purpose.

4.2 User requirements for LOST VOICES

Production of scores. LOST VOICES is a library of Early European (mostly Renaissance) music scores. It is maintained by a research institution with two main goals: (i) publish (on regular score sheets and on-line) rare collections of Renaissance works, (ii) design and promote advanced editorial practices regarding scholar editions of early music sources, often incomplete or fragmented. All the produced scores are encoded in MEI and must comply to very detailed editorial rules. We cannot of course list them all: they cover usage of ancient and modern clefs, presence of incipits, bars and alterations, signs used for mensural notation, text/music association, etc.

Usage of scores. Scores intended for on-line edition must be submitted to an additional manual process to be compatible with MEI-based rendering tools (VexFlow [3] or Verovio [4]). This includes in particular a specific encoding of variants and missing fragments.

Quality concerns. Most of the editorial rules cannot be automatically checked, and this gives rise to two major issues related to quality management. First, all scores have to be double-checked (i.e., by the person that initially encodes the scores and by a supervisor), a very time-consuming process. Second, the library cannot as such be opened to external contributions, due to the complexity of rules and of the lack of automatic control that would reject inputs falling to match the required encoding requirements.

[3] http://www.vexflow.com
[4] http://www.verovio.org

Syntactic accuracy at the score level
Question – Are measures filled and complete? **Metric** – Proportion of syntactically accurate measures over the total number of measures in the score; 1 for non-measured music.
Question – Do parts have the same length? **Metric** – Proportion of non outliers length parts over (all) the parts of the score.
Question – Is the voice nomenclature correct? **Metric** – Boolean (yes/no).

Table 1. **Syntactic accuracy questions and metrics**

4.3 Quality Metrics

Based on the previous studies, we defined an initial taxonomy of two DSL quality schemas for, resp., NEUMA and LOST VOICES. It is worth noting that the two schemas are significantly distinct, which supports our design choice of a DSL-level modeling of quality requirements.

Due to space restrictions, we illustrate the schemas with a tiny sample of the collected metrics requirements, focusing on consensual quality dimensions of literature: the *consistency*, the *accuracy* and the *completeness*. Other dimensions could be considered if needed. For instance, considering a *provenance* dimension of data (e.g. author, currency, timeliness, volatility) could be relevant.

4.3.1 Score Level

Accuracy is defined as the closeness between data value and their considered correct representation. Classically, two kinds of accuracy are considered: the syntactic accuracy and the semantic one. *Syntactic accuracy* in turn takes two forms. One might first check if the data respect an adequate format (validity). External constraints may also be introduced as goals representing specific editorial rules. For instance LOST VOICES requires a specific voice nomenclature (Superius; Cantus; Altus; Contratenor). In all cases, all the metrics in this dimension can automatically be computed from the score encoding. Table 1 contains a few examples.

Semantic accuracy measures the closeness of a value to a considered true real-world value. Its measurement supposes that there is somewhere a reference for the score content, and cannot thus be evaluated by merely looking at an individual document. For the time being, our schemas do not include semantic accuracy metrics. We defer alternative approaches to future work (see Section 5).

Completeness measures in what extent the score contains all the required information. Table 2 contains some examples. It is worth recalling that defining a metric, and measuring its value, does *not* constitute an *absolute* indicator

Completeness at the score level
Question – Is there a figured bass? **Metric** – Boolean.
Question – Are lyrics present (for vocal music only)? **Metric** – Boolean.
Question – Are meta-data fields present? **Metric** – Proportion of available and syntactically required meta-data fields.

Table 2. **Completeness questions and metrics**

Consistency at the score level
Question – Are rendering options consistently used in the score encoding? **Metric** – Proportion of rendering options detected among a set of given ones (e.g., note heads, beaming, positioning, spacing, clef changes).
Question – Are performance indications uniformly present? **Metric** – Uniform encoding of slurs, articulation symbols, etc.

Table 3. **Consistency questions and metrics**

of the DSL quality. Measuring the presence of a figured bass for instance is only important in some usages, and for specific corpora, and its absence does not mean that the corpora are not fit for other usages.

Consistency, at the score level, mostly denotes a uniform encoding of notational features. Positioning information for score elements (notes, chords), fingering, uniform and constant representation of the figured bass are relevant examples for our use cases (see Table 3).

4.3.2 Collection Level

Most of the collection-level metrics are obtained by an aggregation process, which summarizes one or several score-level measurements spread over the collection. In the simplest form, each metric at the score-level allows to define a corresponding metric at the collection-level, which is computed as an average or standard deviation of the score-level metric. Another part of the collection-level metrics are simply not inferred from the collection level. We give a few examples of representative situations.

Accuracy measurements are typically obtained by simple statistical calculation. The (syntactic) accuracy metrics related to measures for instance compute the ratio of scores that contains incomplete measures. Another, less directly computable aspect, is related to the collection structure and presentation. NEUMA for instance requires a fixed ordering of the scores in a collection (Table 4). Note that the later metrics (as many of the same kinds) requires an external information which might be, if available, a public reference of the collection content.

Completeness of a collection (Table 5) measures to what extent the collection is complete enough w.r.t. an expected population size. Here, again, this either requires an external information of reference, or an evaluation by an expert or a group of experts.

Consistency measures in what extend scores of a collection respect a uniform representation (in Table 6).

Collection accuracy
Question – Are measures correctly encoded? **Metric** – Ratio of correct measures.
Question – Are scores ordered as required? **Metric** – Deviation from the required order.

Table 4. **Collection accuracy questions and metrics**

Collection completeness
Question – Is the collection population complete enough? **Metric** – Proportion of available scores over the expected reference population.

Table 5. **Collection completeness questions and metrics**

Collection consistency
Question – Are (meta-)data supplied uniformly supplied? **Metrics** – Standard deviations of the collection population for metrics of Tables 1, 2 and 3.

Table 6. **Collection consistency questions and metrics**

5. CONCLUSION AND PERSPECTIVES

In this paper, we proposed a methodology for assessing data quality in a DSL, based on user preferences. Our approach defines a generic data model that supports the specification of quality schemas, lets users define their goals with respect to the schema of their DSL, and matches usages against quality evaluation. We used this approach for two real digital libraries, and formalized with our model the users' requirements. The implementation is currently in progress for all the metrics that can be evaluated without any external information. This covers syntactic accuracy at the score level, and collection-level aggregated metrics.

Our proposal is a first step that must be completed in several directions. First, several of the metrics identified during our preliminary study cannot be evaluated from the notation itself, but require an external reference. A first solution is a **collaborative** evaluation (some methodologies were proposed e.g. in [6, 1]), for instance based on crowdsourcing. This approach is particularly relevant for the quality dimensions that require external skills like, e.g., semantic accuracy mentioned in 4.3. Another one is to exploit open **semantic web data** by interlinking the DSL collections with other data sources [16].

A second important perspective is to address another aspect of quality management, namely **quality improvement** techniques [4]. Such an improvement can be fully automatic in some specific cases (e.g., filling incomplete measures with rests) but in general, the goal is to help users to identify the insert/update process deficiencies, and to suggest effective improvement strategies.

To our knowledge, no previous work in the literature has proposed metrics for the quality evaluation of music notation. We believe that the topic is important given the lack of constraints of current formats, and the growing production of XML encoded scores.

Acknowledgement: This work has been funded by the French CNRS under the Mastodons project GioQoso.

6. REFERENCES

[1] Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Sören Auer, and Jens Lehmann. Crowdsourcing linked data quality assessment. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 260–276, 2013.

[2] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. *Encyclopedia of Software Engineering*, chapter The Goal Question Metric Approach. Wiley, 1994.

[3] Carlo Batini, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. Methodologies for data quality assessment and improvement. *ACM Comput. Surv.*, 41(3):16:1–16:52, July 2009.

[4] Carlo Batini and Monica Scannapieco. *Data Quality: Concepts, Methodologies and Techniques*. Data-Centric Systems and Applications. Springer, 2006.

[5] Martin J. Eppler and Markus Helfert. A framework for the classification of data quality costs and an analysis of their progression. In *Proceedings of the International Conference on Information Quality*, pages 311–325, 2004.

[6] Yolanda Gil and Varun Ratnakar. Trusting information sources one citizen at a time. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 162–176, 2002.

[7] Michael Good. *MusicXML for Notation and Analysis*, pages 113–124. W. B. Hewlett and E. Selfridge-Field, MIT Press, 2001.

[8] Andrew Hankinson, Perry Roland, and Ichiro Fujinaga. The Music Encoding Initiative as a Document-Encoding Framework. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 293–298, 2011.

[9] Music Encoding Initiative. `http://music-encoding.org`, 2015. Accessed Oct. 2015.

[10] MuseScore. Web site. https://musescore.org/.

[11] Thomas C. Redman. *Data Quality for the Information Age*. Artech House Inc., 1996.

[12] Jenn Riley and Constance A. Mayer. Ask a Librarian: The Role of Librarians in the Music Information Retrieval . In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2006.

[13] Perry Rolland. The Music Encoding Initiative (MEI). In *Proc. Intl. Conf. on Musical Applications Using XML*, pages 55–59, 2002.

[14] Eleanor Selfridge-Field, editor. *Beyond MIDI : The Handbook of Musical Codes*. Cambridge: The MIT Press, 1997.

[15] W3C Music Notation Community Group. https://www.w3.org/community/music-notation/, 2015. Last accessed Jan. 2016.

[16] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer. Quality assessment for linked data: A survey. *Semantic Web*, 7(1):63–93, 2016.

ALIGNED HIERARCHIES: A MULTI-SCALE STRUCTURE-BASED REPRESENTATION FOR MUSIC-BASED DATA STREAMS

Katherine M. Kinnaird
Department of Mathematics, Statistics, and Computer Science
Macalester College, Saint Paul, Minnesota, USA
kkinnair@macalester.edu

ABSTRACT

We introduce *aligned hierarchies*, a low-dimensional representation for music-based data streams, such as recordings of songs or digitized representations of scores. The aligned hierarchies encode all hierarchical decompositions of repeated elements from a high-dimensional and noisy music-based data stream into one object. These aligned hierarchies can be embedded into a classification space with a natural notion of distance. We construct the aligned hierarchies by finding, encoding, and synthesizing all repeated structure present in a music-based data stream. For a data set of digitized scores, we conducted experiments addressing the fingerprint task that achieved perfect precision-recall values. These experiments provide an initial proof of concept for the aligned hierarchies addressing MIR tasks.

1. INTRODUCTION

From Foote's field-shifting introduction of the self-similarity matrix visualization for music-based data streams in [9] to the enhanced matrix representations in [11, 17] and hierarchical segmentations in [14, 18, 21], music information retrieval (MIR) researchers have been creating and using representations for music-based data streams in pursuit of addressing a variety of MIR tasks, including structure tasks [10, 14, 17, 18], comparison tasks [2–4, 11], and the beat tracking task [1, 5, 8, 13]. These representations are often tailored to a particular task, limited to a single layer of information, or committed to a single decomposition of structure. As a result most of the representations for music-based data streams provide narrow insight into the content of the data stream they represent.

In this work, we introduce *aligned hierarchies*, a novel representation that encodes multi-scale pattern information and overlays all *hierarchical* decompositions of those patterns onto one object by *aligning* [1] these hierarchical decompositions along a common time axis. This representation uncovers repeated structures observed in matrix representations widely used in MIR (such as self-similarity matrices, self-dissimilarity matrices, and recurrence plots) and can be used to visualize all decompositions of the repeated structures present in a particular music-based data stream as well as the relationships between the repeats in that data stream. By including and aligning all repeated structures found in a music-based data stream, the aligned hierarchies exist in the middle ground of representations between the density of information in Foote's self-similarity matrix visualization [9] and the sparsity of information in representations like those found in [1, 11, 14, 20].

Beyond the visualization benefits, aligned hierarchies have several compelling properties. Unlike many representations in the literature, the aligned hierarchies can be embedded into a classification space with a natural distance function. This distance function serves as the basis for comparing two music-based data streams by measuring the total dissimilarity of the patterns present. Additionally, the aligned hierarchies can be post-processed to narrow our exploration of a music-based data stream to certain lengths of structure, or to address numerous MIR tasks, including the cover song task, the segmentation task, and the chorus detection task. Such post-processing techniques are not the focus of this paper and will be explored further in future work. In this paper, as a proof of concept for our approach to MIR comparison tasks, we use aligned hierarchies to perform experiments addressing the fingerprint task on a data set of digitized scores.

There are previous structure-based approaches to the cover song task, such as [1, 11, 20], that do not use the formal segmentation of pieces of music and instead, use enhanced matrix representations of songs as the basis of their comparisons. Like those in [9], these representations compare the entire song to itself, but fail to intuitively show detailed structural decompositions of each song. In [2–4], a variety of music comparison tasks are addressed by developing a method of comparison based on audio shingles, which encode local information. In this work, we use audio shingles as the feature vectors to form the self-dissimilarity matrices representing the scores in the data set.

In Section 2, we introduce the aligned hierarchies and the algorithm that builds them. In Section 3, we define the classification space that aligned hierarchies embed into and the associated distance function. In Section 4, we report on experiments using aligned hierarchies to address

[1] We note that 'alignment' in this case refers to placing found structure along a common axis, not to matching a score to the recording of a piece.

© Katherine M. Kinnaird. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Katherine M. Kinnaird. "Aligned Hierarchies: A Multi-Scale Structure-Based Representation for Music-Based Data Streams", 17th International Society for Music Information Retrieval Conference, 2016.

the fingerprint task for a data set of digitized scores, and we summarize our contributions in Section 5.

2. ALIGNED HIERARCHIES

In this section, we define the aligned hierarchies for a music-based data stream and present a motivating example. We introduce the three phases for constructing the aligned hierarchies with discussions about the purpose and motivation for each phase. For simplicity, we will use 'song' to refer to any kind of music-based data stream.

The algorithm finds meaningful repetitive structure in a song from the self-dissimilarity matrix representing that song. The algorithm aligns all possible hierarchies of that structure into one object, called the ***aligned hierarchies*** of the song. The aligned hierarchies H has three components: the onset matrix B_H with the length vector w_H and annotation vector α_H that together act as a key for B_H.

The ***onset matrix*** B_H is an $(n \times s)$-binary matrix, where s is the number of time steps in the song, and where n is the number of distinct kinds of repeated structure found in the song. We define B_H as follows

$$(B_H)_{i,j} = \begin{cases} 1 & \text{if an instance of } i^{\text{th}} \text{ repeated} \\ & \text{structure begins at time step } j, \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The ***length vector*** w_H records the lengths of the repeated structure captured in the rows of B_H in terms of number of time steps, and the ***annotation vector*** α_H records the labels for the groups of repeated structure encoded in these rows. These labels restart at 1 for each distinct length of repeated structure and serve to distinguish groups of repeated structure with the same length from each other. We note that we can exchange any two rows in B_H representing repeats with the same length without losing any information stored in the aligned hierarchies and without changing either w_H or α_H.

2.1 Motivating Example

Suppose we have a song that has two kinds of non-overlapping repetitive structure, such as a verse and a chorus, denoted V and C, respectively, appearing in the order $VCVCV$. We say that the song has the *segmentation* $VCVCV$, where V and C are the repeated sections. We can segment the song in several ways: $\{V, C, V, C, V\}$, $\{(VC), (VC), V\}$, or $\{V, (CV), (CV)\}$, with (VC) representing the piece of structure composed of the V structure followed by the C structure and similarly for (CV). Noting that both (VC) and (CV) can be decomposed into smaller pieces, we would like to find an object that captures and synthesizes all possible decompositions. Figure 1 is a visualization of one such object where the V structure is 3 beats long and the C structure is 5 beats long.

The object that produces the visualization shown in Figure 1 is known as the aligned hierarchies, and it encodes the occurrences and lengths of all the repeated structure found in a song. In Figure 1, we see that repeats of (VC) and the repeats of (CV) overlap in time, but are not contained in each other. We also note that all decompositions of the repeats of (VC) and (CV) are encoded in this object.

In this example, we have four kinds of repeated structures: V, C, (VC), and (CV). Therefore B_H associated to the aligned hierarchies will have four rows, one corresponding to each kind of repeated structure, and 19 columns, one for each beat. Listing the rows in order of the lengths of the repeated structures and the initial occurrences of those repeats, we have that B_H is a sparse matrix with 1's for the V structure at $\{$(1,1), (1,9), (1,17)$\}$, with 1's for the C structure at $\{$(2,4), (2,12)$\}$, with 1's for the (VC) structure at $\{$(3,1), (3,9)$\}$, and with 1's for the (CV) structure at $\{$(4,4), (4,12)$\}$. Then w_H is the column vector $[3, 5, 8, 8]^t$ and α_H is $[1, 1, 1, 2]^t$.

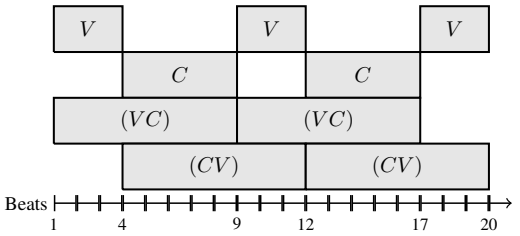


Figure 1: Visualization of aligned hierarchies for a song with segmentation $VCVCV$ incorporating all possible decompositions of the song with V structure 3 beats long and C structure 5 beats long.

2.2 Building the Aligned Hierarchies

The construction of the aligned hierarchies begins with either a self-similarity matrix or a self-dissimilarity matrix. By beginning with a matrix representation for a song, we assume that we do not have access to the original presentation of the song, such as the audio recording, score, or midi file. In a world with proprietary data, extremely high-dimensional data, and limited or restricted access to data, we believe that it is important to develop robust techniques for representing and comparing songs beginning from a data representation that cannot be reverse engineered back to the original presentation of the data. For this work, we will use a self-dissimilarity matrix to represent each song; an example of one for the score of Chopin Mazurka Op. 6, No. 1 is shown in Figure 3a.

Our construction of the aligned hierarchies for a song is motivated by the fact that repeated structures in a song are represented as diagonals of small-valued entries in $\mathcal{D}$, the self-dissimilarity matrix representing the song [6, 16, 17]. If such a diagonal of length k exists in $\mathcal{D}$ beginning at entry (i, j), then the section of the song beginning at time step i that is k time steps long is a repeat of the k time step long section beginning at time step j, and vice versa. We call these sections a pair of repeats of size k.

We construct the aligned hierarchies from simple and meaningful repetitive structure present in a song. For example, suppose a sequence of five chords is played repeatedly in a song. We do not regard repetitions of just the

first three chords as meaningful repeats, unless there is at least one instance in the song of those three chords without the last two or at least one instance of the last two chords without the first three.

Building the aligned hierarchies has three phases:

1. Extract repeated structure of all possible lengths from $\mathcal{D}$, the self-dissimilarity matrix of the song
2. Distill extracted repeated structure into their essential structure components
3. Build aligned hierarchies for the song using the essential structure components

2.2.1 Phase 1 - Extract Repeated Structure from Self-Dissimilarity Matrix $\mathcal{D}$

There are four steps to extracting repeated structure from $\mathcal{D}$. First, we define what repeats are, in context of the data and task at hand. Second, we extract the coarsest repeated structure from $\mathcal{D}$. Third, we use this found structure to uncover further repeated structure hidden by the presentation of the song as $\mathcal{D}$. Lastly, we create groups of repeated structure from the extracted pairs of repeats. In this last step, we enforce a mimicking how humans notice and interpret patterns by removing any group of repeated structure that contains overlapping repeats.

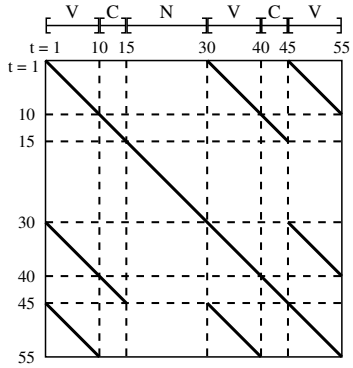
Step 1: Based on the data and the task of interest, we set a threshold T that defines how similar two sections must be in order to be considered repeats of each other and then threshold $\mathcal{D}$ accordingly. We note that many ways exist in the literature to set this threshold, such as [2, 10, 11, 16]. The resulting thresholded matrix $\mathcal{T}$ is a binary matrix of the same dimensions as $\mathcal{D}$ and is given by

$$\mathcal{T}_{i,j} = \begin{cases} 1 & \text{if } \mathcal{D}_{i,j} < T \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Step 2: We next find and extract pairs of coarse repeats in the song, by finding all non-zero diagonals in $\mathcal{T}$, recording relevant information about the pair of repeats, and finally removing the associated diagonal from $\mathcal{T}$. We loop over all possible repeat lengths, beginning with the largest possible structure (the number of columns in $\mathcal{T}$) and ending with 1 (the smallest possible structure).

To find simple and meaningful structure of exactly length k represented by diagonals of exactly length k, we must remove all diagonals of length greater than k. Suppose that we did not remove diagonals of length $(k+1)$ before searching for diagonals of length k, and let $\widehat{d}_{i,j}$ be one such diagonal of 1's in $\mathcal{T}$. Then along with the other diagonals of length k, our algorithm would find two diagonals of length k: one starting at (i, j) and another starting at $(i+1, j+1)$. Our algorithm would not be able to tell that these diagonals of length k are contained in the diagonal $\widehat{d}_{i,j}$ or that together these diagonals make the diagonal $\widehat{d}_{i,j}$. Thus our algorithm would not be finding simple and meaningful repeated structure in the song as required.

Step 3: Once we have extracted all diagonals from $\mathcal{T}$, we use the smaller extracted repeated structure to find additional repeated structures hidden in the coarse repeats. Suppose we examine a piece of text where a certain word is repeated both by itself and in a repeated phrase. In the previous step, our algorithm would find the repeated word on its own and the repeated phrase, but would not detect the repeated word as part of that repeated phrase. In this step, our algorithm realizes that our repeated word is part of the repeated phrase and that the repeated phrase breaks up into at most three pieces, those being: 1) the part of the phrase before our repeated word, 2) the repeated word itself, and 3) the part of the phrase after the repeated word.



(a) $\mathcal{T}$ for a toy song with sections marked

	Start Time Step	Start Time Step	Repeat Length
VC	1	31	15
V	1	46	10
V	31	46	10

(b) Pairs of repeats after Step 2 of Phase 1, the initial extraction from $\mathcal{T}$ with sections marked

	Start Time Step	Start Time Step	Repeat Length
VC	1	31	15
V	1	46	10
V	31	46	10
V	1	31	10
C	11	41	5

(c) Pairs of repeats after Step 3 of Phase 1, the second part of extraction with sections marked

Figure 2: Thresholded matrix $\mathcal{T}$ and the pairs of repeats uncovered after each step of repeat extraction for toy song with segmentation $VCNVCV$

Consider the song with segmentation $VCNVCV$ and with the thresholded distance matrix $\mathcal{T}$ shown in Figure 2a. In the initial extraction, the algorithm finds three pairs of repeats, two pairs encoding repeats of the V structure by itself and one pair encoding two repeats of (VC), as shown in Table 2b. But the algorithm has not detected that the pair of (VC) repeats contain the smaller found V structure as well as the yet to be isolated C structure. In this step, as shown in Table 2c, by using either of the pairs of V repeats, we find that the pair of (VC) repeats does contain a pair of V repeats as well as a pair of smaller repeats that is not the same as the V structure, known as the C structure.

Step 4: In the last step of this phase, we form groups of repeats from the pairs of repeats such that each kind of repeated structure has exactly one group of repeats associ-

ated to it. For the example shown in Figure 2a, we have three groups: one associated to (VC), another associated to V, and a third associated to C.

To mimic human segmentation of music, we check that each group does not contain repeats that overlap in time. For the example in Section 2.1, we are more likely to describe the structure of a popular song by saying "the verse and chorus are repeated twice together followed by the verse again," than by saying "the verse, chorus, and verse are repeated together twice such that those two repeats overlap at one verse." Thus we do not encode the repeated structure (VCV) in the aligned hierarchies shown in Figure 1, even though it occurs twice in $VCVCV$.

2.2.2 Phase 2 - Distill Essential Structure Components

Just like words that are composed of syllables, musical elements, such as motifs and chord progressions, are composed of smaller components. In this step, we distill repeats of the song into their *essential structure components*, the building blocks that form every repeat in the song.

By definition, we allow each time step to be contained in at most one of the song's essential structure components. In this phase, we pairwise compare groups of repeats, checking if the repeats in a given pair of groups overlap in time. If they do, we divide the repeats in a similar fashion as used in Step 3 in Phase 1, forming new groups of repeats that do not overlap in time. We iterate this process, dividing repeats as necessary, until each time step is contained in at most one repeated structure. The repeats remaining at the end of this phase are our essential structure components. For the example in Section 2.1 shown in Figure 1, the essential structure components are the instances of the V and C structures. Figure 3b is a visualization of the essential structure components for the score of Chopin's Mazurka Op. 6, No. 1.

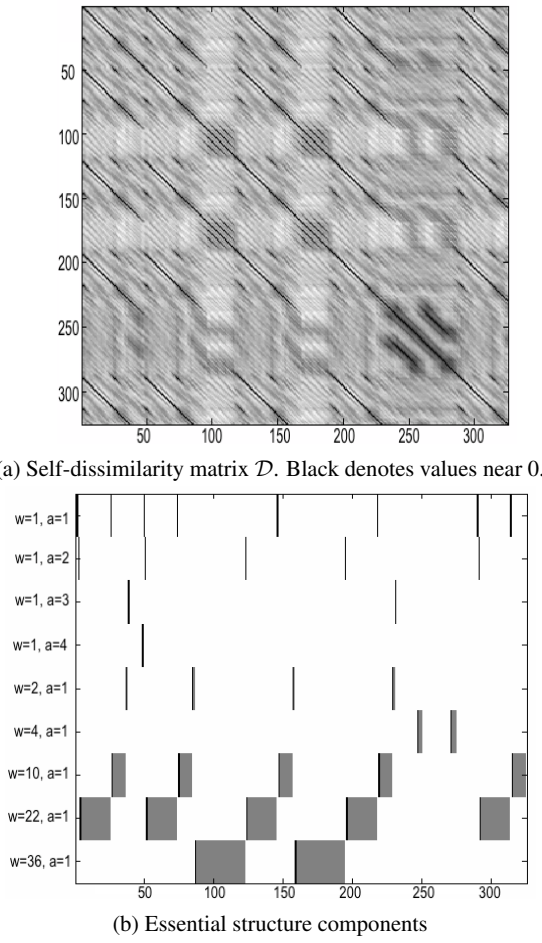
2.2.3 Phase 3 - Construct Aligned Hierarchies from Essential Structure Components

In this final phase, we build the aligned hierarchies from the essential structure components. We employ a process that is akin to taking right and left unions of the essential structure components to find all possible non-overlapping repeats in the song. We encode these repeats in the onset matrix and form the length and annotation vectors that together are the key for the onset matrix. Figure 4 is a visualization of the aligned hierarchies for a score of Chopin's Mazurka Op. 6, No. 1.

3. COMPARING ALIGNED HIERARCHIES

To compare aligned hierarchies, we embed them into a classification space with a distance function measuring the total dissimilarity between pairs of songs. In Section 3.1, we explain how aligned hierarchies embed into this classification space, and we present the distance function used for comparing aligned hierarchies of songs in Section 3.2. [2]

[2] The proofs for the material in this section can be found in the author's doctoral thesis [12].



(a) Self-dissimilarity matrix $\mathcal{D}$. Black denotes values near 0.

(b) Essential structure components

Figure 3: Visualizations for a score of Chopin's Mazurka Op. 6, No. 1 with repeat markers observed.

3.1 Classification Space for Aligned Hierarchies

To define $(\mathcal{S}^*)^n$, the space that we embed aligned hierarchies into, while simultaneously demonstrating how this embedding occurs, we begin by representing aligned hierarchies as a sequence of matrices.

Definition 3.1. Given a particular song with s time steps and its aligned hierarchies H, we define a sequence of s binary matrices $\left\{B^k\right\}_{k=1}^{s}$ where the k^{th} binary matrix B^k is the rows of B_H such that $w_H = k$, which are the rows corresponding to repeats of exactly k time steps. If there are no repeats of exactly k time steps, then B^k is a row of s zeros. For brevity, we will use $\left\{B^k\right\}$ for $\left\{B^k\right\}_{k=1}^{s}$.

We note that each binary matrix in $\left\{B^k\right\}$ does not have a pair of vectors acting as a key for it, as we have in H. Our definition of $\left\{B^k\right\}$ naturally encodes the information from w_H in $\left\{B^k\right\}$. Similarly, we construct α_H so that the labels for the groups of repeats restart at 1 for each distinct repeat length l. Thus, for each l, the label corresponding to a row in $B^l \in \left\{B^k\right\}$ is simply that row's index in B^l.

We recall that we can exchange any two rows of B_H with $w_H = l$ without changing the annotation labels. So we say that two matrices encoding repeats of exactly

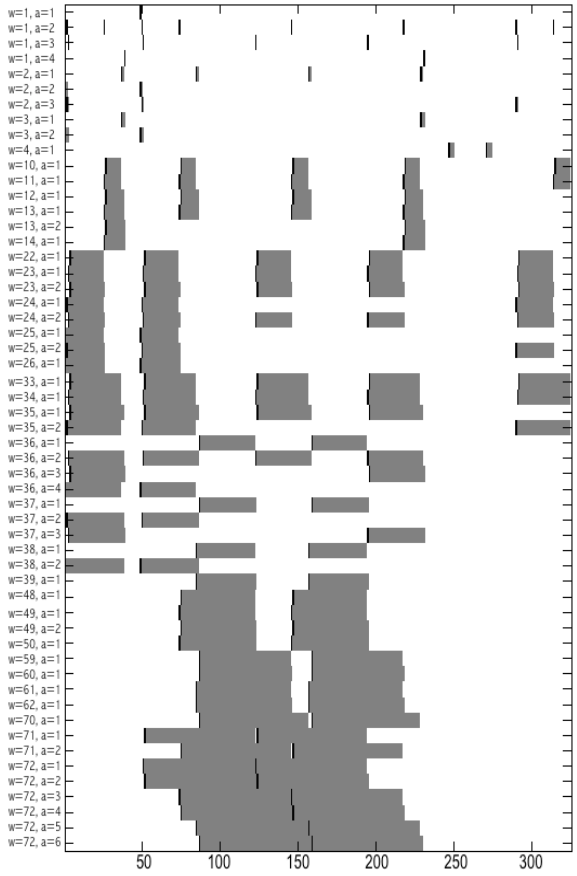


Figure 4: Visualization of aligned hierarchies for a score of Chopin's Mazurka Op. 6, No. 1 with repeat markers observed.

length l are the same if one is a row permutation of the other. Therefore the space that we embed the aligned hierarchies into must inherit this notion of matrix equality.

Definition 3.2. Let $\mathcal{S}$ be the space of $(m \times t)$-binary matrices with $m, t \in \mathbb{Z}_{\geq 1}$. Consider the symmetric group S_m, the group of all permutations for the rows of the matrices. The matrix denoted $M_{\sigma(r)} \in \mathcal{S}$ is the matrix with the rows of $M \in \mathcal{S}$ in the order prescribed by $\sigma(r) \in S_m$.

Proposition 3.1. *Let $\sim$ be the relation on $\mathcal{S}$ such that for $M, Q \in \mathcal{S}$, we say that $M \sim Q$ if $M = Q_{\sigma(r)}$, for some $\sigma(r) \in S_m$. Then $\sim$ is an equivalence relationship on $\mathcal{S}$.*

Definition 3.3. Let $\mathcal{S}^*$ be the quotient space $\mathcal{S}/\sim$. Then the product space $(\mathcal{S}^*)^n$ is composed of n-copies of $\mathcal{S}^*$.

We embed H into $(\mathcal{S}^*)^n$ by setting $t = s$, the number of time steps in the song, and choosing $m = \kappa_{\max}$, where $\kappa_{\max} = \max_{l \in \{1,\ldots,s\}} \left\{ r | r \text{ is the number of rows in } B^l \right\}$. Then we place each $B^l \in \left\{B^k\right\}$ into the l^{th} space of $(\mathcal{S}^*)^n$. So the l^{th} quotient space in $(\mathcal{S}^*)^n$ corresponds to the classification space of patterns of length l present in songs.

Notation: The elements of product space $(\mathcal{S}^*)^n$ are sequences of elements in $\mathcal{S}^*$, pedantically denoted as $(q^g)_{g=1}^n$ with $q^g \in \mathcal{S}^*$, for each $g \in \{1, \ldots, n\}$. For brevity, we will use (q^g) for $(q^g)_{g=1}^n$.

3.2 Metric for Comparing Aligned Hierarchies

To define a metric on the space $(\mathcal{S}^*)^n$ that will measure the total dissimilarity between two songs represented by their aligned hierarchies, we begin by defining a function that measures the dissimilarity between patterns of a *fixed* size present in those two aligned hierarchies.

Definition 3.4. Let $||\cdot||_1$ be the entry-wise 1-norm. Given any $s_1, s_2 \in \mathcal{S}^*$, let $f : \mathcal{S}^* \times \mathcal{S}^* \to \mathbb{R}$ be the function given by

$$f(s_1, s_2) = \min_{\substack{\delta \in s_1 \\ \beta \in s_2}} ||\delta - \beta||_1 \tag{3}$$

Proposition 3.2. *The function $f : \mathcal{S}^* \times \mathcal{S}^* \to \mathbb{R}$ is a distance function.*

To define the metric that measures the *total* dissimilarity between two songs, we use the above function f to compute the dissimilarity between the repeated patterns at each size and total the measured dissimilarities. This gives us the *total* dissimilarity between the repeated patterns of *all* sizes present in two aligned hierarchies.

Corollary 1. *Let $(q^g), (r^g) \in (\mathcal{S}^*)^n$. The function $d_H : (\mathcal{S}^*)^n \times (\mathcal{S}^*)^n \to \mathbb{R}$ is a distance function, where d_H is given by*

$$d_H\left((q^g), (r^g)\right) = \sum_{g=1}^{n} f(q^g, r^g). \tag{4}$$

4. PROOF OF CONCEPT RESULTS

In this section, we consider the fingerprint task for a data set of digitized musical scores. These experiments serve as a proof of concept for our method of comparing songs via their aligned hierarchies. With the exception of the feature extraction, the code implementing the creation and comparison of aligned hierarchies is written in MATLAB. [3]

4.1 Data Set and Features

Our data set is based on 52 Mazurka scores by Chopin. For each score, we download two human-coded, digitized versions, called ***kern files*, posted on the KernScore online database [19]. [4] The first version has the repeated sections repeated as many times as marked in the score and the second has the repeated sections presented only once per time written. For scores that have no sections that are repeated in their entirety, we download the single **kern file twice, marking one copy as having the repetitions repeated and the second copy as having the repetitions not repeated. Each version of a score is referred to as a song and there are 104 songs in our data set.

In this data set, the notion of time is in terms of beats with one time step per beat. For each beat, we extract the chroma feature vector, encoding the amount of each of the 12 Western pitch classes present in that beat [15]. To do

[3] The URL to the code used for the experiments can be found at https://github.com/kmkinnaird/ThesisCode/releases/tag/vT.final2

[4] The **kern files can be accessed at: http://kern.humdrum.org/search?s=t&keyword=Chopin

this, we used the `music21` Python library [7]. [5] We form audio shingles, that encode local contextual information, by concatenating γ consecutive chroma feature vectors, for a fixed integer γ.

We create a symmetric self-dissimilarity matrix $\mathcal{D}$ using a cosine dissimilarity measure between all pairs of audio shingles. Let a_i, a_j be the audio shingles for time steps i and j, respectively. Then we define

$$\mathcal{D}_{i,j} = \left(1 - \frac{< a_i, a_j >}{||a_i||_2||a_j||_2}\right) \quad (5)$$

By setting γ, we set the smallest size of repeated structure that can be detected. For this work, we set $\gamma = 6$ or $\gamma = 12$. Assuming that the average tempo of a Mazurka is approximately 120 beats per minute, if $\gamma = 6$, our shingles encode about 3 seconds of information similar to the audio shingles in [2,3]. Similarly, if $\gamma = 12$, our shingles encode four bars of three beats each or about 6 seconds.

4.2 Evaluation Procedure

For all of our experiments, given a particular threshold, we construct the aligned hierarchies for each score. We compute the pairwise distances between the songs' representations as described in Section 3.2. Using these pairwise distances, we create a network for the data set with the songs in the data set as the nodes. In the fingerprint task, we only match songs that are exact copies of each other, and so we define an edge between two nodes if the distance between the two aligned hierarchies associated to the songs is 0.

We evaluate the results of our experiments by computing the precision-recall values for the resulting network compared against a network representing the ground truth, which is formed by placing edges between the two identical copies of the score present in the data set. This ground truth was informed by a data-key based on human-coded, meta-information about the scores.

For each experiment, we set γ, the width of the audio shingles, and T, the threshold value for defining when two audio shingles are repeats of each other. The choice of γ and T affects the amount of structure classified as repeats, which impacts whether or not a song has aligned hierarchies to represent it. If a song does not have aligned hierarchies, due to the choice of γ and T, we remove the node representing that song from consideration in both our experiment network and in our ground truth network, as there would be nothing for our method to use for comparison.

4.3 Results

We conducted 10 experiments with the threshold $T \in \{0.01, 0.02, 0.03, 0.04, 0.05\}$ and with $\gamma \in \{6, 12\}$. Each experiment yielded a perfect precision-recall value. For the experiment with $T = 0.01$ and $\gamma = 12$, we had 5 songs without aligned hierarchies including 2 pairs of songs based on scores without repeated sections. For the experiment with $T = 0.02$ and $\gamma = 12$, we had 1 song without aligned hierarchies, but this song was based on a score with repeated sections and thus, under the fingerprint task, would not be matched to another song in our data set.

We note that our method did discover an error in the data key for the Mazurka scores. According to the human-coded, meta-information, Mazurka Op. 17, No. 1 was classified as having sections marked in the score as being repeated. However, the score of Mazurka Op. 17, No. 1 in fact does not have any sections marked to be repeated. Our algorithm correctly detected this error, and we corrected our version of the data key for these Mazurka scores. The corrected data key is our ground truth, which is what our precision-recall values are based on. To our knowledge, there is no published work using this data set, therefore we cannot provide numerical comparisons between our method and other ones.

For all 10 experiments, we have correctly identified all scores, with aligned hierarchies, that do not have sections marked in the score to be repeated. Based on the construction of the score data set, a perfect recall rate was expected. More interestingly, the perfect precision rate means that we do not falsely match scores using the aligned hierarchies.

5. CONCLUSION

In this paper, we have introduced the aligned hierarchies, an innovative, multi-scale structure-based representation for music-based data streams. The aligned hierarchies provide a novel visualization for repeated structure in music-based data streams. Differing from the literature of enhanced matrix representations, instead of showing a comparison of a data stream to itself, the visualization of the aligned hierarchies synthesizes all possible hierarchical decompositions of that data stream onto one time axis, allowing for a straightforward understanding of the temporal relationships between the repeated structures found in a particular data stream.

The aligned hierarchies also provide a mathematically rigorous method for comparing music-based data streams using this low-dimensional representation. We performed experiments addressing the fingerprint task for data based on digitized scores. These experiments had perfect precision-recall rates and provided a proof of concept for the aligned hierarchies.

In future work, we will develop post-processing techniques for the aligned hierarchies. These techniques will allow us to address additional MIR tasks, such as the cover song task and the chorus detection task. We also will continue to develop the theory and metrics associated with aligned hierarchies and their derivatives.

Acknowledgements

This work is a portion of the author's doctoral thesis [12], which was partially funded by the GK-12 Program at Dartmouth College (NSF award # 0947790). Part of this work was performed while the author was visiting the Institute for Pure and Applied Mathematics (IPAM), which is supported by the National Science Foundation. The author thanks Scott Pauls, Michael Casey, and Dan Ellis for their feedback on early versions of this work.

[5] See http://web.mit.edu/music21/ for information about `music21`.

6. REFERENCES

[1] J. Bello. Measuring structural similarity in music. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2013–2025, 2011.

[2] M. Casey, C. Rhodes, and M. Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(5):1015 – 1028, 2008.

[3] M. Casey and M. Slaney. Song intersection by approximate nearest neighbor search. *Proceedings of the International Society for Music Information Retrieval*, pages 144–149, 2006.

[4] M. Casey and M. Slaney. Fast recognition of remixed audio. *2007 IEEE International Conference on Audio, Speech and Signal Processing*, pages IV – 1425 – IV–1428, 2007.

[5] M. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668– 696, 2008.

[6] M. Cooper and J. Foote. Summarizing popular music via structural similarity analysis. *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 127 –130, 2003.

[7] M. S. Cuthbert and C. Ariza. `music21`: A toolkit for computer-aided musicology and symbolic music data. *11th International Society for Music Information Retrieval Conference*, 2010.

[8] D.P.W. Ellis and G.E. Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages IV – 1429–1432, 2007.

[9] J. Foote. Visualizing music and audio using self-similarity. *Proc. ACM Multimedia 99*, pages 77–80, 1999.

[10] M. Goto. A chorus-section detection method for musical audio signals and its application to a music listening station. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1783–1794, 2006.

[11] P. Grosche, J. Serrà, M. Müller, and J.Ll. Arcos. Structure-based audio fingerprinting for music retrieval. *13th International Society for Music Information Retrieval Conference*, 2012.

[12] K. M. Kinnaird. *Aligned Hierarchies for Sequential Data*. PhD thesis, Dartmouth College, 2014.

[13] B. McFee and D. P. W. Ellis. Better beat tracking through robust onset aggregation. In *International conference on acoustics, speech and signal processing*, ICASSP, 2014.

[14] B. McFee and D. P. W. Ellis. Learning to segment songs with ordinal linear discriminant analysis. In *International conference on acoustics, speech and signal processing*, ICASSP, 2014.

[15] M. Müller and S. Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. *12th International Society for Music Information Retrieval Conference*, 2011.

[16] M. Müller, P. Grosche, and N. Jiang. A segment-based fitness measure for capturing repetitive structures of music recordings. *12th International Society for Music Information Retrieval Conference*, pages 615–620, 2011.

[17] J. Paulus, M. Müller, and A. Klapuri. Audio-based music structure analysis. *11th International Society for Music Information Retrieval Conference*, pages 625–636, 2010.

[18] C. Rhodes and M. Casey. Algorithms for determining and labelling approximate hierarchical self-similarity. *8th International Society for Music Information Retrieval Conference*, 2007.

[19] C.S. Sapp. Online database of scores in the humdrum file format. *Proceedings of the International Society for Music Information Retrieval*, pages 664–665, 2005.

[20] D.F. Silva, H. Papadopoulos, G.E.A.P.A. Batista, and D.P.W. Ellis. A video compression-based approach to measure music structure similarity. *Proceedings of the International Society for Music Information Retrieval*, pages 95–100, 2013.

[21] K. Yoshii and M. Goto. Unsupervised music understanding based on nonparametric bayesian models. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5353–5356, 2012.

ANALYSING SCATTERING-BASED MUSIC CONTENT ANALYSIS SYSTEMS: WHERE'S THE MUSIC?

Francisco Rodríguez-Algarra Centre for Digital Music
Bob L. Sturm Centre for Digital Music
Hugo Maruri-Aguilar School of Mathematical Sciences
Queen Mary University of London, U.K.
{f.rodriguezalgarra, b.sturm, h.maruri-aguilar}@qmul.ac.uk

ABSTRACT

Music content analysis (MCA) systems built using scattering transform features are reported quite successful in the *GTZAN* benchmark music dataset. In this paper, we seek to answer why. We first analyse the feature extraction and classification components of scattering-based MCA systems. This guides us to perform intervention experiments on three factors: train/test partition, classifier and recording spectrum. The partition intervention shows a decrease in the amount of reproduced ground truth by the resulting systems. We then replace the learning algorithm with a binary decision tree, and identify the impact of specific feature dimensions. We finally alter the spectral content related to such dimensions, which reveals that these scattering-based systems exploit acoustic information below 20 Hz to reproduce *GTZAN* ground truth. The source code to reproduce our experiments is available online. [1]

1. INTRODUCTION

Music content analysis (MCA) systems trained and tested in [2] reproduce a large amount of the ground truth of the benchmark music dataset *GTZAN* [18], and are among the "best" reported in the literature [14]. They use support vector machines (SVM) classifiers trained on features extracted from audio by the *scattering transform*, a non-linear spectrotemporal modulation representation using a cascade of wavelet transforms [8]. The mathematical derivation of the scattering transform enforces invariances to local time and frequency shifts, which is a desirable property for music classification tasks. Scattering features are considered to have perceptual relevance [2], and can be related to modulation features [5]. Such features are potentially useful for timbre-related music classification tasks, such as instrument recognition [11], or genre recognition [7].

Reproducing the ground truth of a dataset does not necessarily reflect the ability of a system to address a particular task [12–15]. In this paper, we analyse scattering-based MCA systems to determine why they reproduce so much *GTZAN* ground truth. Our approach involves system analysis and experimental interventions. System analysis involves decomposing an MCA system into its components to understand how each contributes to its overall behaviour. Our system analysis of scattering-based MCA systems in Sec. 2 shows that they use some information from inaudible frequencies, i.e., below 20 Hz [4]. Experimental interventions, on the other hand, involve testing hypotheses about what a system is actually doing by altering some factor to see how system behaviour changes. In Sec. 3, we perform intervention experiments to confirm that scattering-based MCA systems exploit information below 20 Hz to reproduce *GTZAN* ground truth. When we attenuate that information, ground truth reproduction decreases.

We conceive our work here as a case study within the development of an improved systematic methodology for evaluating MCA systems. This is one challenge posed in the Music Information Retrieval (MIR) Roadmap [10], and exemplifies the pipeline in [15]. In Sec. 4 we discuss the implications of our results, and suggest how they might be integrated in a general evaluation framework.

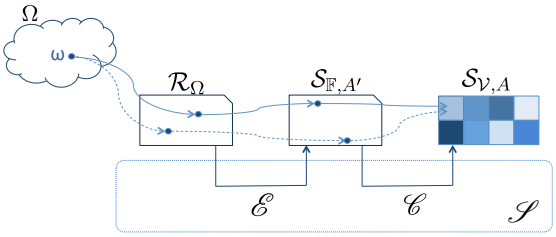


Figure 1. Schematic representation of a music content analysis (MCA) system [16].

2. SYSTEM ANALYSIS

Using the formalism of [16], an MCA system $\mathscr{S}$ maps a recording universe $\mathcal{R}_\Omega$ — a particular realisation of an intangible music universe Ω — to a description universe $\mathcal{S}_{\mathcal{V},A}$. As shown in Fig. 1, this mapping is decomposed into two stages. First, a feature extractor $\mathscr{E}$ maps $\mathcal{R}_\Omega$ to a feature universe $\mathcal{S}_{\mathbb{F},A'}$; then, a classifier $\mathscr{C}$ maps $\mathcal{S}_{\mathbb{F},A'}$ to $\mathcal{S}_{\mathcal{V},A}$.

The environment and definition of the MCA systems in [2] are as follows. $\mathcal{R}_\Omega$ consists of time-domain sig-

[1] https://code.soundsoftware.ac.uk/projects/scatter-analysis

© Francisco Rodríguez-Algarra, Bob L. Sturm, Hugo Maruri-Aguilar. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Francisco Rodríguez-Algarra, Bob L. Sturm, Hugo Maruri-Aguilar. "Analysing Scattering-based Music Content Analysis Systems: Where's the Music?", 17th International Society for Music Information Retrieval Conference, 2016.

ID	$\mathbb{F}$	Feature Description
a	$\mathbb{R}^{252}$	Mel-frequency spectrogram (84 coefficients, 740-ms frames, 50% overlap), concatenated with first- and second-order time derivatives over the sequence of feature vectors [2]
b	$\mathbb{R}^{85}$	First-order ($l = 1$) time-scattering features (effective sampling rate 2.7 Hz)
c	$\mathbb{R}^{747}$	Second-order ($l = 2$) time scattering features (effective sampling rate 2.7 Hz)
d	$\mathbb{R}^{1574}$	First-order time-frequency scattering features
e	$\mathbb{R}^{1907}$	First-order time-frequency-adaptive scattering features
f	$\mathbb{R}^{2769}$	Third-order ($l = 3$) time scattering features (effective sampling rate 2.7 Hz)

Table 1. Description of $\mathcal{S}_{\mathbb{F},A'}$ (feature universes) used in [2]. A' permits only vector sequences of length 80.

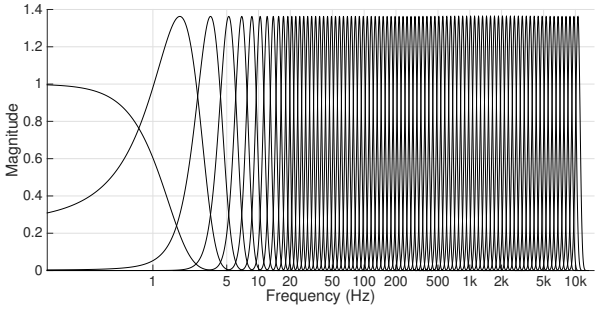
nals of duration about 30 seconds uniformly sampled at $F_s = 22050$ Hz (the sampling rate of *GTZAN*). $\mathcal{S}_{\mathcal{V},A}$ is the set of the 10 *GTZAN* labels. $\mathcal{S}_{\mathbb{F},A'}$ is a space consisting of sequences of 80 elements of a vector vocabulary $\mathbb{F}$. All MCA systems trained in [2] use the same $\mathcal{S}_{\mathcal{V},A}$, the same learning method to build the classifiers, but different $\mathcal{S}_{\mathbb{F},A'}$. More specifically, the semantic rules A' are the same for all systems (sequences of length 80), with only a difference in the feature vocabulary, $\mathbb{F}$. Table 1 describes the six different $\mathcal{S}_{\mathbb{F},A'}$ appearing in [2].

We now analyse the two components of the systems built using first- ("b vectors") and second-layer ("c vectors") time scattering features (see Table 1). Systems built using f vectors can be understood as a further iteration of the process described here. In addition to that, the inclusion of frequency-scattering features (d and e vectors), does not affect our conclusions.

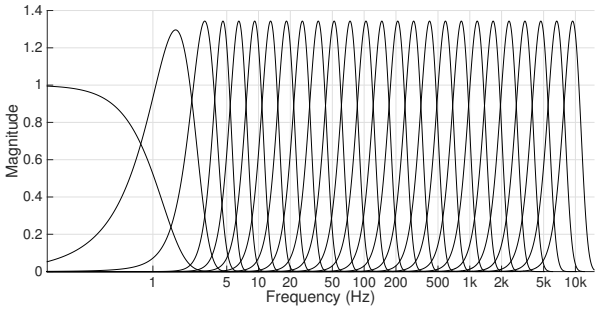
2.1 Feature extractors of b and c ($\mathscr{E}_{\mathtt{b}}$ and $\mathscr{E}_{\mathtt{c}}$)

The feature extraction procedure begins by first extending a recording to be of length $2^{21} = 2097152$ samples using what is referred to in the code as "padding" by "symmetric boundary condition with half-sample symmetry": the $N \approx 5 \cdot 2^{17}$ samples of $r \in \mathcal{R}_\Omega$ are concatenated with the same but time-reversed, then concatenated with its first ~ 50000 samples, and its last ~ 50000 samples, and finally the time-reversed samples again. This "padded" signal is then transformed into the frequency domain by the FFT. The complex spectrum is then multiplied by the magnitude response of each of 85 filters of a filterbank designed using a scaling function and dilations of a one-dimensional Gabor mother wavelet with 8 wavelets per octave up to a maximum dilation of $2^{73/8}$. (The bandwidth of the lowest 11 bands are made constant.) Figure 2(a) shows the magnitude responses of the bands of this filterbank (FB1). Each spectrum product is then reshaped — equivalent to decimation in the time-domain —, transformed to the time domain by the inverse FFT, and then windowed to the portion corresponding to r in the padded sequences.

Next, the time-series output of each band of FB1 is rectified, padded using the same padding method as above,



(a) Filterbank 1 (FB1)



(b) Filterbank 2 (FB2)

Figure 2. Magnitude responses of the bands in the filterbanks for scattering feature IDs b and c.

transformed into the frequency domain by the FFT, and then multiplied by the magnitude response of each of 25 filters of a filterbank designed with a scaling function and dilations of a one-dimensional Morlet mother wavelet, with 2 wavelets per octave up to a maximum dilation of $2^{23/2}$. Figure 2(b) shows the magnitude responses of the bands of this filterbank (FB2). Each FB2 spectrum product is then reshaped — again, equivalent to decimation in the time-domain —, transformed to the time domain by the inverse FFT, and then windowed corresponding to the original forward-going sequence in the padded sequences (length 80). Finally, $\mathscr{E}_{\mathtt{b}}$ retains only those values related to the DC filter of FB2, and computes the natural log of all values (added with a small positive value). This results in 80 b vectors. For creating 80 c vectors, $\mathscr{E}_{\mathtt{c}}$ takes those FB2 time-series outputs with non-negligible energy, [3] "renormalises" each non-zero frequency band (to account for energy captured in the first layer of scattering coefficients), and takes the natural log of all values (added with a small positive value).

Figure 3 shows the relationship between the dimensions of b and c vectors and the centre frequencies of FB1 and FB2 bands. For display purposes, the bottom-most row is from the scaling function of FB2. The 85 dimensions of a b vector are at bottom, with dimensions [1, 75:85] coming from FB1 bands with centre frequencies below 20 Hz. Dimensions [1, 75:85, 737:747] of a c vector come from such bands. Dimensions [2:12] of a b vector, and [2:12, 86:268] of a c vector, are from FB1 bands with centre frequencies above 4186 Hz (pitch C8).

[2] [2] does not actually compute Δ- and Δ-Δ-MFCCs, but instead cyclically time-shifts the sequence of MFCCs ahead and behind by one frame so that the classifier has flexibility in learning a transformation.

[3] In fact, not every rectified FB1 band output is filtered by all FB2 bands because filtering by FB1 will remove all frequencies outside its band.

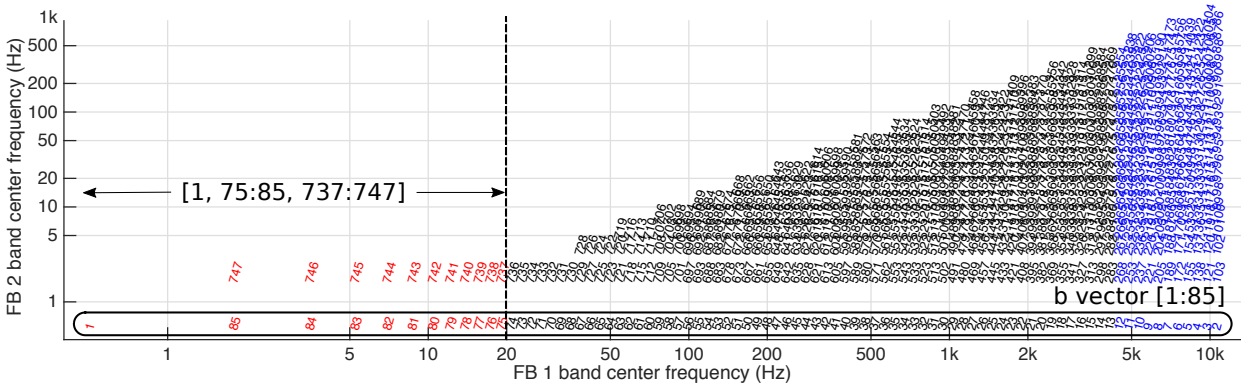


Figure 3. Relationship of b and c vector dimensions to FB1 and FB2 band centre frequencies. Dimensions [1, 75:85] of b vectors, and [1, 75:85, 737:747] of c vectors, are from bands with centre frequencies below 20 Hz.

2.2 Classifier $\mathscr{C}$

Define the number of support vectors of a trained SVM as $|SV|$. Classifiers $\mathscr{C}$ of the MCA systems in [2] are characterised by a set of support vectors $\mathbf{V} \in \mathbb{F}^{|SV|}$, a Gaussian kernel parameter γ, a weight matrix $\mathbf{W} \in \mathbb{R}^{|SV| \times 45}$, and a bias vector $\boldsymbol{\rho} \in \mathbb{R}^{45}$. (45 is the number of pair-wise combinations of the 10 elements in $\mathcal{S}_{\mathcal{V},A}$, i.e., label 1 vs. label 2, label 1 vs. label 3, etc.) $\mathscr{C}$ maps $\mathcal{S}_{\mathbb{F},A'}$ to $\mathcal{S}_{\mathcal{V},A}$ by majority vote from the individual mappings of all elements $f_j \in \mathbb{F}$ of a sequence from $r \in \mathcal{R}_\Omega$ by an SVM classifier $\mathscr{C}'$. $\mathscr{C}'$, thus, maps $\mathbb{F}$ to $\mathcal{S}_{\mathcal{V},A}$ by computing 45 pair-wise decisions by means of $sign(\mathbf{W}^T \mathrm{e}^{-\gamma \mathbf{K}(f)} - \boldsymbol{\rho})$, where $\mathbf{K}(f)$ is a vector of squared Euclidean norm of differences between f and all $v_j \in \mathbf{V}$. $\mathscr{C}'$ then maps f to $\mathcal{S}_{\mathcal{V},A}$ by majority vote from the 45 pair-wise decisions.

The authors of [2] use LibSVM [4] to build $\mathscr{C}'$ using a Gaussian kernel with a subset of the feature vectors (downsampled by 2). They optimise the SVM parameters by grid search and 5-fold cross-validation on a training set. LibSVM uses a 1 vs. 1 strategy to deal with multiclass classification problems, so each support vector receives a weight for each of the nine possible pair-wise decisions involving the class associated with the support vector. The matrix $\mathbf{W}$ contains weights associated with all possible 45 pair-wise decisions. The training of the SVM also generates the vector $\boldsymbol{\rho}$ containing a bias term for each pair-wise decision.

3. SEARCHING FOR THE MUSIC

We now report three intervention experiments we design to answer our question: how are scattering-based MCA systems reproducing so much *GTZAN* ground truth? We adapt the code used for the experiments in [2] (available online [5]). The experiments performed in [2] do not consider the known faults of *GTZAN* [14], so in Sec. 3.1 we reproduce them using two different train-test partitioning conditions. We observe a decrease in performance, but not as dramatic as seen in past re-evaluations [14]. In Sec. 3.2, we replace the classifier $\mathscr{C}$ with a binary decision tree (BDT) trained with different subsets of scattering features. This leads us to identify the impact of specific feature dimensions. The analysis of the feature extractor in Sec. 2.1 allows us to relate such dimensions with spectral bands of the audio signal. In Sec. 3.3, we alter the spectral content of the test recordings and observe how *GTZAN* ground truth reproduction changes. This reveals that these scattering-based MCA systems exploit acoustic information below 20 Hz.

3.1 Partitioning intervention

The benchmark music dataset *GTZAN* contains faults (e.g., repetitions) that can affect the amount of ground truth that an MCA system reproduces [14]. This amount often decreases when we train and test it using a "fault-filtered" partition of *GTZAN*, as done in [6, 14]. This suggests that the faults in the dataset are related to the amount of ground truth reproduced by a system.

While [2] evaluates the performance of the scattering-based MCA systems using 10-fold stratified cross-validation, we employ two different hold-out train-test partitioning conditions. The first is *RANDOM*, which mimics the train-test procedure in [2]: we randomly select 75% of the recordings of each label for the training set, leaving the remaining 25% for the testing set. The second is *FAULT*, which is the "fault-filtered" partitioning procedure in [6], with the training and validation sets merged. This partitioning condition considers various problems of the dataset: we remove 70 replicated or distorted recordings [14]; we then assign by hand 640 recordings to the training set and the remaining 290 to the testing set, avoiding repetition of artists across partitions [9]. Due to memory constraints, we decrease by a factor of 4 the number of scattering features in the pre-computation of the Gaussian kernel of the SVM. This reduces the computational cost without sacrificing much performance. [6]

Table 2 shows the normalised accuracies (mean recalls) of our systems along with those reported in [2] for the six features described in Table 1. We see the differences between the results in [2] and ours in *RANDOM* are small, and most of them within reason considering the standard

[4] https://www.csie.ntu.edu.tw/~cjlin/libsvm/
[5] http://www.di.ens.fr/data/software

[6] We acknowledge Joakim Andén and Vincent Lostanlen for their valuable advice.

	Original *GTZAN* recordings			Attenuated [0, 20] Hz	
ID	Reported in [2]	*RANDOM*	*FAULT*	*RANDOM*	*FAULT*
a	82.0 ± 4.2	78.00	53.29	39.20	30.09
b	80.9 ± 4.5	79.20	54.96	31.60	22.42
c	89.3 ± 3.1	88.00	66.46	50.80	44.47
d	90.7 ± 2.4	87.20	68.49	62.40	55.11
e	91.4 ± 2.2	85.60	68.61	64.80	44.52
f	89.4 ± 2.5	83.60	68.32	64.80	53.16

Table 2. Normalised accuracies (mean recall) in *GTZAN* dataset obtained by scattering-based MCA systems in [2] and our systems using *RANDOM* and *FAULT* partitioning conditions, trained and tested with the original *GTZAN* recordings (left) and versions ones with information below 20 Hz (right) attenuated (see Sec. 3.3).

N	*RANDOM*	*FAULT*
1	52.99	51.82
2	65.05	65.27
3	71.42	71.62
4	75.53	75.80
5	79.48	79.74

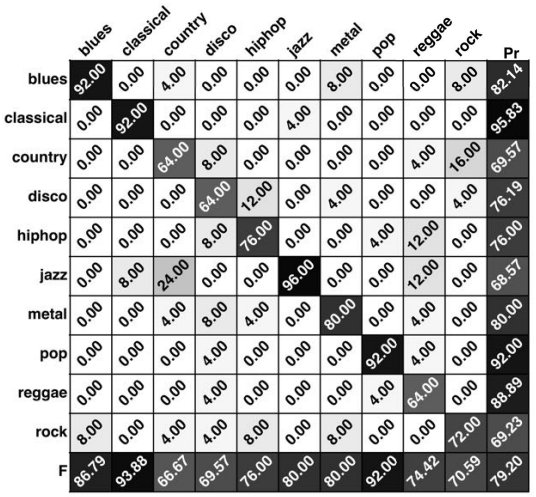
Table 3. Cumulative percentage of variance captured by the N highest principal components of b vectors in the training sets of *RANDOM* and *FAULT* partitioning conditions.

deviations reported in [2]. In *RANDOM*, we see an increase of accuracy when we include second-order scattering features, i.e., b to c. We find that adding depth to the features, however, does not increase further the amount of ground truth reproduced, and even decreases it when we include third-order features (c to f), contrary to what is reported in [2]. Most importantly, we observe a considerable decrease in the amount of ground truth reproduced by all systems between *RANDOM* and *FAULT*. Figures 4(a) and 4(b) show the figure-of-merit (FoM) of the systems trained and tested in *RANDOM* and *FAULT* with b vectors, respectively. We see recalls and F-measures of every label decrease except for "classical", which increase.

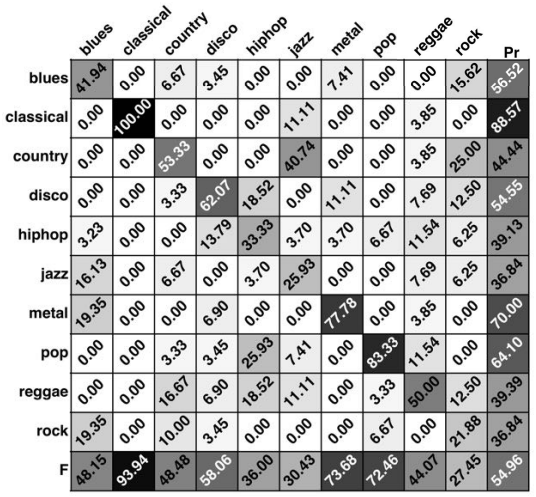
Figure 5 shows the eigenvectors of the first five principal components of first-layer scattering features (b vectors) in the training sets of *RANDOM* and *FAULT*. (Table 3 shows the percentage of variance captured by the first N principal components.) We see large changes in the lowest and highest dimensions of the fourth component. This suggests that these dimensions of the scattering features capture information that differs in both training sets, which may play a role in the performance differences we observe. The characteristics of $\mathscr{C}$ and $\mathscr{C}'$, however, make it difficult to determine the influence that each individual feature dimension (or subset of dimensions) has in the overall performance of a system. For this reason, in Section 3.2 we replace the SVM by a binary decision tree (BDT) classifier, which allows an easier interpretation of $\mathcal{S}_{\mathbb{F},A'}$ and its relationship with $\mathcal{S}_{\mathcal{V},A}$.

3.2 Classifier intervention

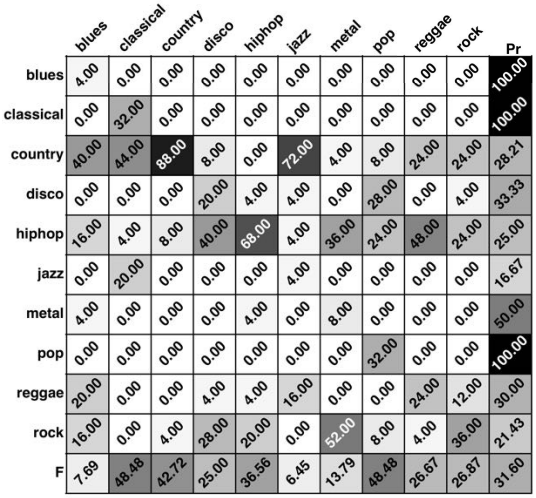
SVM classifiers generate decision boundaries in multi-dimensional spaces. While this can benefit prediction, it hampers their interpretability. In our case, this implies that



(a) *RANDOM*

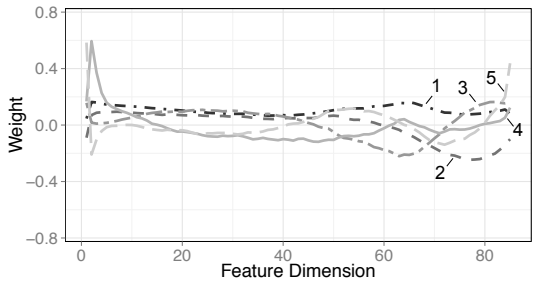


(b) *FAULT*

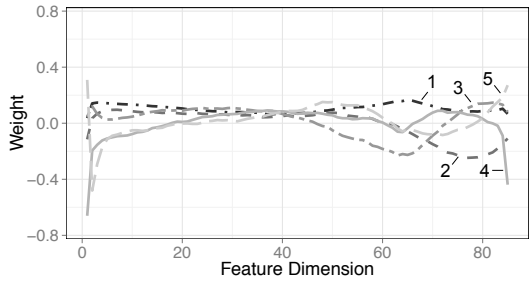


(c) *RANDOM*, information below 20 Hz attenuated

Figure 4. Figure-of-merit (FoM) obtained with b vectors by SVM systems trained and tested in (a) *RANDOM* and (b) *FAULT* (Sec. 3.1), as well as (c) SVM trained in *RANDOM* and tested in recordings with content below 20 Hz attenuated (Sec. 3.3). Column is ground truth, row is prediction. Far-right column is precision, diagonal is recall, bottom row is F-score, lower right-hand corner is normalised accuracy. Off-diagonals are confusions.



(a) *RANDOM*



(b) *FAULT*

Figure 5. Eigenvectors of the first five principal components (labelled) of b vectors in the training sets of (a) *RANDOM* (79.74% of variance captured) and (b) *FAULT* (79.48% of variance captured) partitioning conditions.

ID	*RANDOM*	*FAULT*
a	72.80	45.70
b	71.60	42.35
c	80.00	49.91
d	79.20	46.81
e	79.60	44.77
f	79.20	46.48

Table 4. Normalised accuracies (mean recall) in *GTZAN* for MCA systems built using binary decision tree classifiers using features described in Table 1, trained and tested with *RANDOM* and *FAULT* partitioning conditions.

the relevance of each individual dimension of the scattering feature vectors gets blurred. We now replace the SVM classifiers used in [2] by BDT, consisting of a set of rules defined by linear splits of the feature space one dimension at a time. BDT are considered to be among the easiest learning methods to construct and understand [1], at the cost of potentially less accuracy.

Table 4 shows the normalised accuracies we obtain with MATLAB's BDT classifier, [7] for the two partitioning conditions defined in Sec. 3.1, using the different feature vectors described in Table 1. Clearly, there exists a major difference between the two training conditions, similar to what Table 2 shows for SVM. We see a decrease of around 8 percentage points in the amount of ground truth reproduced by each of the BDT systems in *RANDOM* compared to the SVM systems in Table 2. On the other hand, when training the BDT systems in *FAULT*, we observe falls in performance with respect to *RANDOM* at least as large as those reported in Table 2. This suggests that the amount of

[7] http://uk.mathworks.com/help/stats/classificationtree-class.html

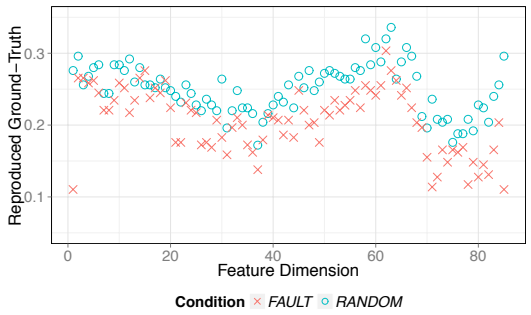


Figure 6. Proportion of ground truth reproduced by BDT classifiers trained with single dimensions of b vectors in *RANDOM* and *FAULT* partitioning conditions.

ground truth reproduced by the systems in both conditions differ due to distinct information being captured by the feature extractors, and not necessarily as an effect of the classification algorithm. The training of the SVM classifier, if anything, appears to mitigate the potential performance decrease in c-f vectors.

We now explore differences in reproduced *GTZAN* ground truth between partitioning conditions by each dimension of the scattering features individually. We thus train and test BDTs with each of the 85 dimensions of b vectors in both *RANDOM* and *FAULT*. Figure 6 shows the classification accuracies we obtain. We see clear differences between conditions, especially in dimensions identified in Sec. 2.1 as belonging to bands close to or outside the limits of normal human hearing (namely [1, 70:85]).

We also explore how much ground truth BDT systems can reproduce using solely information below 20 Hz. BDT systems trained with dimensions 1 and 75:85 of b vectors achieve a 60.40% of normalised accuracy in *RANDOM*, which is close to the performance originally reported in [18] for the *GTZAN* dataset. In *FAULT*, however, the normalised accuracy drops to 22.47%. Adding dimensions 737:747 from c vectors (modulations from FB2 of information below 20 Hz) only marginally increases the performance in both conditions. These results suggest that our scattering-based MCA systems could be exploiting acoustic information from below 20 Hz. We next perform interventions to test this hypothesis.

3.3 Filtering intervention

We now see how the amount of ground truth reproduced by a scattering-based an MCA system changes when we attenuate acoustic information below 20 Hz. We thus apply a fifth-order Butterworth high-pass filter to attenuate all frequencies below 20 Hz by at least 30 dB to the test recordings in both *RANDOM* and *FAULT*. We check that we do not perceive differences between filtered and non-filtered versions. We then use the SVM systems in Sec. 3.1 to predict labels in the filtered test recordings. The two right-most columns of Table 2 show the normalised accuracies we obtain. We clearly see that the figures drop from those reported in Sec. 3.1. In particular, the decrease of accuracy using b vectors in *RANDOM* is close to 50 percentage points, while that using features generated from

deeper scattering layers is smaller but still notable. Systems trained in *FAULT* also suffer in the performance measured.

Figure 4(c) shows the FoM obtained by an SVM trained in *FAULT* with $\mathtt{b}$ vectors and tested in high-pass filtered recordings. We note that the changes in FoM between Figs. 4(a) and 4(c) do not always match those reported in Sec. 3.1 between Figs. 4(a) and 4(b). More precisely, recall and F-measure decrease instead of increase in "classical", and increase instead of decrease in "country". This suggests that partitioning and information below 20 Hz are distinct factors affecting the amount of ground truth systems reproduce, notwithstanding an interaction between them as suggested by Figs. 5 and 6. Our results allow us to conclude that the scattering-based MCA systems trained and tested in [2] benefit from partitioning and exploit acoustic information below 20 Hz to reproduce a large amount of *GTZAN* ground truth.

4. DISCUSSION

Our analysis in Sec. 2.1 shows how first- and second-layer time-scattering features relate to acoustic information. We see that several dimensions of such features capture information at frequencies below 20 Hz, which is inaudible to humans [4].

We find in the intervention experiments in Secs. 3.1 and 3.2 that partitioning affects the amount of *GTZAN* ground truth scattering-based systems reproduce. Removing the known faults of the dataset and avoiding artist replication across folds leads to a decrease in the FoM we obtain, but to a lesser extent than previous re-evaluations of other MCA systems [6, 14]. We also note that differences between the first principal components of first-layer time-scattering features lay mainly within the dimensions corresponding to frequency bands below 20 Hz.

When we replace the SVM classifier with a BDT (Sec. 3.2), we see differences in the amount of reproduced ground truth similar to those we find for SVM systems between partitioning conditions. This suggests that the distinct acoustic information the scattering features capture causes differences in performance, regardless of the particular learning algorithm employed. Furthermore, we find that BDT systems trained with individual dimensions of first-layer time-scattering features reproduce an amount of *GTZAN* ground truth larger than that expected when selecting randomly. Again, we see differences between partitioning conditions, especially in the dimensions capturing information below 20 Hz. Moreover, we reproduce almost as much *GTZAN* ground truth as the one originally reported in [18] by using a BDT trained in *RANDOM* with only information below 20 Hz. This result suggests that acoustic information below 20 Hz present in *GTZAN* recordings may inflate the performance of MCA systems trained and tested in the benchmark music dataset.

Our system analysis in Sec. 2 and intervention experiments in Sec. 3.1 and 3.2 point toward information present in frequencies below 20 Hz playing an important role in the apparent success of the scattering-based MCA systems we examine. The results of our experiments in Sec. 3.3 clearly reveal that the amount of *GTZAN* ground truth SVM scattering-based systems reproduce decreases when we attenuate that information in test recordings. This implies these systems are using inaudible information. We conclude that the scattering-based MCA systems in [2] exploit acoustic information not controlled by partitioning and below 20 Hz to reproduce a large amount of *GTZAN* ground truth. Machine music listening is an exciting prospect as it complements and even extends human abilities, but we dispute the relevance of acoustic information below 20 Hz to address the problem intended by *GTZAN* [18].

The results of our three intervention experiments suggest a complex relationship between the accuracy measured of a system, the contribution of its feature extraction and machine learning components, and the conditions of the training and testing dataset. We already know that the faults and partitioning of *GTZAN* can have significant effects on an outcome, and that there is an interaction with the components of a system [14, 17]. Our experiments here show for the systems we examine that acoustic information below 20 Hz can greatly affect an outcome, and that this interacts with the components of a system and the dataset partitioning. This thus calls into question the interpretation of the results reported in [2] (column 2 of Table 2) as unbiased estimates of system success. In future work, we will specify more complex measurement models, e.g., [17].

Understanding how and why a system works is essential to determine its suitability for a specific task, not to mention its improvement. Our work here demonstrates the use of system analysis and the intervention experiment to address this problem. For instance, our conclusions suggest modifying the FB1 filterbank in the scattering features extractor to avoid capturing information below 20 Hz. They also suggest removing information below 20 Hz from any element of $\mathcal{R}_\Omega$ as a pre-processing step before training an MCA system, if relevant.

5. CONCLUSION

In this paper, we report several steps we followed to determine what the scattering-based MCA systems reported in [2] have actually learned to do in order to reproduce the ground truth of *GTZAN*. We show how performing system analysis guides our design of appropriate intervention experiments. The results lead us to conclude that these MCA systems benefit not only from the partitioning of the dataset, but also from acoustic information below 20 Hz.

Our work here constitutes steps toward a holistic analysis of MCA systems — an action point for MIR evaluation identified in [10]. Our ultimate goal is to help develop a general MIR research pipeline that integrates system analysis and interventions, and is grounded in formal principles of statistical design of experiments, e.g., [3]. Such a pipeline will provide a solid empirical foundation upon which to build machine music listening systems and technologies [15].

6. REFERENCES

[1] E. Alpaydin. *Introduction to Machine Learning*. The MIT Press, Cambridge, MA, USA, 3rd edition, 2014.

[2] J. Andén and S. Mallat. Deep Scattering Spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, 2014.

[3] R. A. Bailey. *Design of Comparative Experiments*. Cambridge University Press, 2008.

[4] A. Chaudhuri. *Fundamentals of Sensory Perception*. Oxford University Press, 2011.

[5] T. Chi, P. Ru, and S. A. Shamma. Multiresolution Spectrotemporal Analysis of Complex Sounds. *Journal of the Acoustical Society of America*, 118(2):887–906, 2005.

[6] C. Kereliuk, B. L. Sturm, and J. Larsen. Deep Learning and Music Adversaries. *IEEE Transactions on Multimedia*, 17(11):2059–2071, 2015.

[7] C. Lee, J. Shih, K. Yu, and H. Lin. Automatic Music Genre Classification Based on Modulation Spectral Analysis of Spectral and Cepstral Features. *IEEE Transactions on Multimedia Multimedia*, 11(4):670–682, June 2009.

[8] S. Mallat. Group Invariant Scattering. *Communications on Pure and Applied Mathematics*, LXV:1331–1398, 2012.

[9] E. Pampalk, A. Flexer, and G. Widmer. Improvements of Audio-Based Similarity and Genre Classification. In *Proc. 6th International Society for Music Information Retrieval Conference (ISMIR'05)*, pages 628–633, London, UK, September 2005.

[10] X. Serra, M. Magas, E. Benetos, M. Chudy, S. Dixon, A. Flexer, E. Gómez, F. Gouyon, P. Herrera, S. Jordá, O. Paytuvi, G. Peeters, H. V. Schluter, and G. Widmer. *Roadmap for Music Information Research*. The MIReS Consortium, 2013.

[11] K. Siedenburg, I. Fujinaga, and S. McAdams. A Comparison of Approaches to Timbre Descriptors in Music Information Retrieval and Music Psychology. *Journal of New Music Research*, 45(1):27–41, January 2016.

[12] B. L. Sturm. Classification Accuracy Is Not Enough. *Journal of Intelligent Information Systems*, 41(3):371–406, 2013.

[13] B. L. Sturm. A Simple Method to Determine if a Music Information Retrieval System Is a "Horse". *IEEE Transactions on Multimedia*, 16(6):1636–1644, 2014.

[14] B. L. Sturm. The State of the Art Ten Years After a State of the Art: Future Research in Music Information Retrieval. *Journal of New Music Research*, 43(2):147–172, 2014.

[15] B. L. Sturm. Revisiting Priorities: Improving MIR Evaluation Practices. In *Proc. 17th International Society for Music Information Retrieval Conference (ISMIR'16)*, New York, NY, USA, August 2016.

[16] B. L. Sturm, R. Bardeli, T. Langlois, and V. Emiya. Formalizing the Problem of Music Description. In *Proc. 15th International Society for Music Information Retrieval Conference (ISMIR'14)*, pages 89–94, Taipei, Taiwan, October 2014.

[17] B. L. Sturm, H. Maruri-Aguilar, B. Parker, and H. Grossman. The Scientific Evaluation of Music Content Analysis Systems: Valid Empirical Foundations for Future Real-World Impact. In *Proc. ICML Machine Learning for Music Discovery Workshop*, Lille, France, July 2015.

[18] G. Tzanetakis and P. Cook. Musical Genre Classification of Audio Signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–301, 2002.

BEAT TRACKING WITH A CEPSTROID INVARIANT NEURAL NETWORK

Anders Elowsson
KTH Royal Institute of Technology
elov@kth.se

ABSTRACT

We present a novel rhythm tracking architecture that learns how to track tempo and beats through layered learning. A basic assumption of the system is that humans understand rhythm by letting salient periodicities in the music act as a framework, upon which the rhythmical structure is interpreted. Therefore, the system estimates the cepstroid (the most salient periodicity of the music), and uses a neural network that is invariant with regards to the cepstroid length. The input of the network consists mainly of features that capture onset characteristics along time, such as spectral differences. The invariant properties of the network are achieved by subsampling the input vectors with a hop size derived from a musically relevant subdivision of the computed cepstroid of each song. The output is filtered to detect relevant periodicities and then used in conjunction with two additional networks, which estimates the speed and tempo of the music, to predict the final beat positions. We show that the architecture has a high performance on music with public annotations.

1. INTRODUCTION

The beats of a musical piece are salient positions in the rhythmic structure, and generally the pulse scale that a human listener would tap their foot or hand to in conjunction with the music. As such, beat positions are an emergent perceptual property of the musical sound, but in various cases also dictated by conventional methods of notating different musical styles. Beat tracking is a popular subject of research within the Music Information Retrieval (MIR) community. At the heart of human perception of beat are the onsets of the music. Therefore, onset detection functions are commonly used as a front end for beat tracking. The most basic property that characterize these onsets is an increase in energy in some frequency bands. Extracted onsets can either be used in a discretized manner as in [9, 18, 19], or continuous features of the onset detection functions can be utilized [8, 23, 28]. As information in the pitch domain of music is important, chord changes can also be used to guide the beat tracking [26].

After relevant onset functions have been extracted, the periodicities of the music are usually determined by e.g. comb filters [28], the autocorrelation function [10, 19], or by calculating the cepstroid vector [11]. Other ways to understand rhythm are to explicitly model the rhythmic patterns [24], or to combine several different models to get better generalization capabilities [4]. To estimate the beat positions, hidden Markov models [23] or dynamic Bayesian networks (DBNs) have been used [25, 30].

Although onset detection functions often are computed by the spectral flux (SF) of the audio, it has become more common to learn onset detection functions with a neural network (NN) [3, 29]. Given the success of these networks it is not surprising that the same framework has been successfully used also for detecting beat positions [2]. When these network try to predict beat positions, they must understand how different rhythmical elements are connected; this is a very complex task.

1.1 Invariant properties of rhythm

When trying to understand a new piece of music, the listener must form a framework onto which the elements of the music can be deciphered. For example, we use scales and harmony to understand pitch in western music. The tones of a musical piece are not classified by their fundamental frequency, but by their fundamental frequency in relation to the other tones in the piece. In the same way, for the time dimension of music, the listener builds a framework, or *grid*, across time to understand how the different sounds or onsets relate to each other. This framework need not initially be at the beat level. In fact, in various music pieces, beat positions are not the first perceptually emergent timing property of the music. In some pieces, we may first get a strong sense of repetition at downbeat positions, or at subdivisions of the beat. In either of these cases, we identify beat positions after an initial framework of rhythm has been established. If we could establish such a correct framework for a learning algorithm, it would be able to build better representations of the rhythmical structure, as the input features would be deciphered within an underlying metrical structure. In this study we try to use this idea to improve beat tracking.

2. METHOD

In the proposed system we use multiple neural networks that each try to model different aspects related to rhythm,

© Anders Elowsson. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Anders Elowsson. "Beat Tracking with a Cepstroid Invariant Neural Network", 17th International Society for Music Information Retrieval Conference, 2016.

as shown in Figure 1. First we process the audio with harmonic/percussive source separation (HP-separation) and multiple fundamental frequency (MF_0) estimation. From the processed audio, features are calculated that capture onset characteristics along time, such as the SF and the pitch flux (PF). Then we try to find the most salient periodicity of the music (which we call the *cepstroid*), by analyzing histograms of the previously calculated onset characteristics in a NN (Cep Network). We use the cepstroid to subsample the flux vectors with a hop size derived from a subdivision of the computed cepstroid. The subsampled vectors are used as input features in our cepstroid invariant neural network (CINN). The CINN can track beat positions in complex rhythmic patterns, because the previous processing has made the input vectors invariant with regards to the cepstroid of the music. This means that the same neural activation patterns can be used for MEs of different tempi. In addition, the speed of the music is estimated with an ensemble of neural networks, using global features for onset characteristics as input. As the last learning step, the tempo is estimated. This is done by letting an ensemble of neural networks evaluate different plausible tempo candidates. Finally, the phase of the beat is determined by filtering the output of the CINN in conjunction with the tempo estimate; and beat positions are estimated.

An overview of the system is given in Figure 1. In Sections 2.1-2.4 we describe the steps to calculate the input features of our NNs and in Section 2.5 we give an overview of the NNs. In Section 2.6-2.9 we describe the different NNs, and in Section 2.10, we describe how the phase of the beat is calculated.

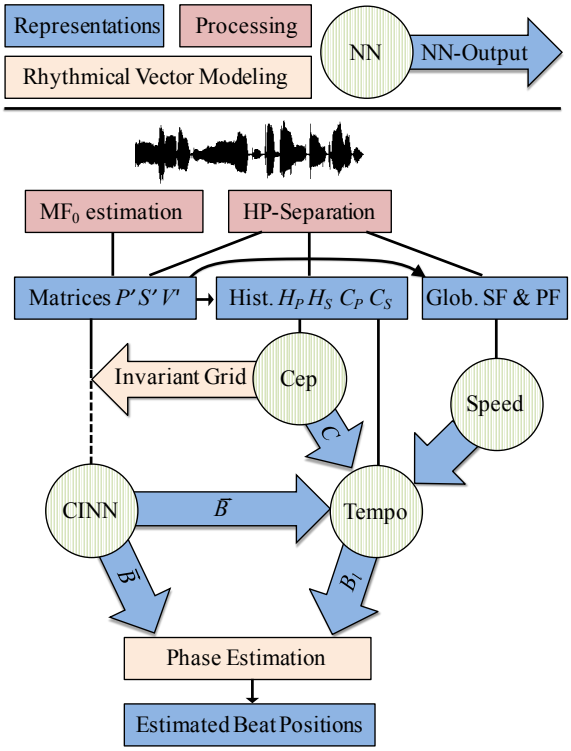


Figure 1. Overview of the proposed system. The audio is first processed with MF_0 estimation and HP-separation. Raw input features for the neural networks are computed and the outputs of the neural networks are combined to build a model of tempo and beats in each song.

2.1 Audio Processing

The audio waveform was converted to a sampling frequency of 44.1 kHz. Then, as a first step, HP-separation was applied. This is a common strategy (e.g. [16]), used to isolate the percussive instruments, so that subsequent learning algorithms can accurately analyze their rhythmic patterns. The source separation of our implementation is based on the method described in [15]. With a median filter across each frame in the frequency direction of a spectrogram, harmonic sounds are detected as outliers, and with a median filter across each frequency bin in the time direction, percussive sounds are detected as outliers. We use these filters to extract a percussive waveform P_1 and a harmonic waveform H_1, from the original waveform O. We further suppress harmonic sounds in P_1 (such as traces of the vocals or the bass guitar) by applying a median filter in the frequency direction of the Constant-Q transform (CQT), as described in [11, 13]. This additional filtering produces a clean percussive waveform P_2, and a harmonic waveform H_2 consisting of the traces of pitched sounds filtered out from P_1.

The task of tracking MF_0s of the audio is usually performed by polyphonic transcription algorithms (e.g. [1]). From several of these algorithms, the frame-wise MF_0s can be extracted at the semi-tone level. We used a frame-wise estimate from [14], extracted at a hop size of 5.8 ms (256 samples).

2.2 Calculating Flux Matrices P', S' and V'

Three types of flux matrices (P', S' and V') were calculated, all extracted at a hop size of 5.8 ms.

2.2.1 Calculating P'

Two spectral flux matrices (P_1' and P_2') were calculated from the percussive waveforms P_1 and P_2. The short time Fourier transform (STFT) was applied to P_1 and P_2 with a window size of 2048 samples and the spectral flux of the resulting spectrograms was computed. Let $X_{i,j}$ represent the magnitude at the ith frequency bin of the jth frame of the spectrograms. The SF for each bin is then given by

$$P'_{i,j} = X_{i,j} - X_{i,j-s} \quad (1)$$

In this implementation we used a step size s of 7 (40 ms).

2.2.2 Calculating V'

The vibrato suppressed SF was computed for waveforms containing instruments with harmonics (H_1, H_2 and O), giving the flux matrices (V'_{H_1}, V'_{H_2} and V'_O). We used the algorithm for vibrato suppression first described in [12] (p. 4), but changed the resolution of the CQT to 36 bins per octave (down from 60) to get a better time resolution.

First, the spectrogram is computed with the CQT. Then, shifts of a peak by one bin, without an increase in sound level, are suppressed by subtracting the sound level of each bin of the new frame, with the maximum sound level of the adjacent bins in the old frame. This means that for the vibrato suppressed SF (V'), Eqn (1) is changed by including adjacent bins and calculating the maximum value before applying the subtraction.

$$V'_{i,j} = X_{i,j} - \max(X_{i-1,j-s}, X_{i,j-s}, X_{i+1,j-s}) \quad (2)$$

2.2.3 Calculating S'

When listening to a melody, we use pitch in conjunction with onset positions to infer the rhythmical structure. Therefore, it seems beneficial to utilize the pitch dimension of music in the beat tracking as well. We calculated the PF by applying the same function as described for the SF in Eqn (1) to the "*semigram*" – the estimated MF_0s in a *pitchogram*, interpolated to a resolution of one semitone per bin. The output is the rate of change in the *semigram*, covering pitches between midi pitch 26 and 104, and we will denote this feature matrix as S'.

2.3 Calculating Histograms H_P, H_S, C_P, and C_S

Next we compute two periodicity histograms H_P and H_S from the flux matrices P'_1 and S', and then transform them into the cepstroid vectors C_P and C_S.

The processing is based on a method recently introduced in [11]. In this method, a periodicity histogram of inter-*onset* intervals (IOIs) is computed, with the contribution of each onset-pair determined by their spectral similarity and their perceptual impact. The basic idea is that the IOI of two strong onsets with similar spectra (such as two snare hits) should constitute a relevant level of periodicity in the music. In our implementation we instead apply the processing frame-wise on P'_1 and S', using the spectral similarity and perceptual impact at each inter-*frame* interval. We use the same notion of spectral similarity and perceptual impact as in [11] when computing H_P from P'_1, but when we compute H_S from S', the notion of *spectral distance* is replaced with *tonal distance*. First we smooth S' in the pitch direction with a Hann window of size 13 (approximately an octave). We then build a histogram of tonal distances for each frame, letting n represent the nth semitone of S' and k the kth frame, giving us the tonal distance at all histogram positions a

$$\forall a \in \{1, \cdots, 1900\} \sum_{i=-50,-45,\cdots,50} \sum_{n=26}^{104} |S'^{n}_{k+i} - S'^{n}_{k+i+a}| \quad (3)$$

By using the grid defined by i in Eqn (3), we try to capture similarities in a few consecutive tones. The grid stretches over 100 frames, which corresponds to roughly 0.5 seconds. The idea is that repetitions of small motives occurs at musically relevant periods.

To get the cepstroid vector from a histogram, the discrete cosine transform (DCT) is first applied. The resulting spectrum unveils periodically recurring peaks of the histogram. In this spectral representation, frequency represents the period length and magnitude corresponds to salience in the metrical structure. We then interpolate back to the time domain by inserting spectral magnitudes at the position corresponding to their wavelength. Finally, the Hadamard product of the original histogram and the transformed version is computed to reduce noise. The result is a cepstroid vector (C_P, C_S). The name *cepstroid* (derived from *period*) was chosen based on similarities to how the *cepstrum* is computed from the *spectrum*.

2.4 Calculating Global SF and PF

Global features for the SF and PF were calculated for our speed estimation. We extracted features from the feature matrices of Section 2.2. The matrices were divided into log-spaced frequency bands over the entire spectrum by applying triangular filters as specified in Table 1.

Feature Matrices	P'_1	P'_2	S'	V'_O	V'_{H_1}	V'_{H_2}
Number of bands	3	3	1,2,4	3	3	3

Table 1. The feature matrices are divided into bands.

After the filtering stage we have 22 feature vectors, and each feature vector X is converted into 12 global features. We compute the means $\bar{X}$, $\overline{X^{0.2}}$ and $\overline{X^{0.5}}$, where 0.2 and 0.5 represents the element-wise power (3 features). Also, X is sorted based on magnitude into percentiles, and Hann windows of widths {41, 61}, centered at percentiles {31, 41} are applied (4 features). We finally extract the percentiles at values {20, 30, 35, 40, 50} (5 features).

2.5 Neural Network Settings

Here we define the settings for all neural networks. In the subsequent Sections 2.6-2.9, further details are provided for each individual NN. All networks were standard feed-forward neural networks with one to three hidden layers.

2.5.1 Ensemble Learning

We employed ensemble learning by creating multiple instances of a network and averaging their predictions. The central idea behind ensemble learning is to use different models that are better than random and more or less uncorrelated. The average of these models can then be expected to provide a better prediction than randomly choosing one of them [27]. For the Tempo and Speed networks, we created an ensemble by randomly selecting a subset of the features for the training of 20 networks (Tempo) or 60 networks (Speed). For the CINN, only 3 networks were used in the ensemble due to time constraints, and all features were used in each network.

2.5.2 Target values

The target values in the networks are defined as:

- **Cep** - Classifying if a frame represents a correct (1) or an incorrect cepstroid (0). The beat interval, downbeat interval, and duple octaves above the downbeat or below the beat were defined as correct.

- **CINN** - Classifying if the current frame is at a beat position (1), or if it is not at a beat position (0).
- **Speed** - Fitting to the log of the global beat length.
- **Tempo** - Classifying which of two tempo candidates that is correct (1) and which is incorrect (0).

2.5.3 Settings of the Neural Networks

We use scaled conjugate descent to train the networks. In Table 2, settings of the neural networks are defined.

	Hidden	***Epoch***	***EaSt***	***EnLe***	***OL***
Cep	{20, 20, 20}	600	100	-	*LoSi*
CINN	{25}	1000		3.	*LoSi*
Speed	{6, 6, 6}	20	4	60_{40}	*Li*
Tempo	{20, 20}	100		20_{60}	*LoSi*

Table 2. The settings for the neural networks of the system. *Hidden*, denotes the size of the hidden layers and *Epoch* is the maximum number of epochs we ran the network. *EaSt* defines how many epochs without an increase in performance that were allowed for the internal validation set of the neural networks. *EnLe* is specified as NE_{NF}, where NE is the number of ensembles and NF is the number of randomly drawn features for each ensemble. *OL* specifies if a logistic activation function (*LoSi*) or a linear summation (*Li*) was used for the output layer.

The activation function of the first hidden layer was always a hyperbolic tangent (tanh) unit, and for subsequent hidden layers it was always a rectified linear unit (ReLU). The use of a mixture of tanh units and ReLUs may seem unconventional but can be motivated. The success of ReLUs is often attributed to their propensity to alleviate the problem of vanishing gradients [17]. Vanishing gradients are often introduced by sigmoid and tanh units when those units are placed in the later layers, because gradients flow backwards through the network during training. With tanh units in the first layer, only gradients for one layer of weight and bias values will be affected. At the same time, the network will be allowed to make use of the smoother non-linearities of the tanh units.

2.6 Cepstroid Neural Network (Cep)

In the first NN we compute the most salient periodicity of the music. To do this we use the cepstroid vectors (C_P and C_S) previously computed in Section 2.3. First, two additional vectors are created from both cepstroid vectors by filtering the vectors with a Gaussian $\sigma = 7.5$, and a Laplacian of a Gaussian $\sigma = 7.5$. Then we include octave versions, by interpolating to a time resolution given by

$$\left(\frac{1}{2}\right)^n, \qquad \left(\frac{1}{2}\right)^n \times \left(\frac{1}{3}\right), \qquad \forall n \in \{-2, -1, 0, 1, 2\} \tag{4}$$

Finally, much like one layer and one receptive field of a convolutional neural network, we go frame by frame through the vectors, trying to classify each histogram frame as correct or incorrect, depending on if that particular time position corresponds to a correct cepstroid. The input features are the magnitude values of the vectors at each frame. As true targets, the beat interval *and* the downbeat interval, as well as duple octaves above the downbeat and duple octaves below the beat are used. The output of the network is our final cepstroid vector (C) and the highest peak is used as our cepstroid ($\hat{C}$).

2.7 Cepstroid Invariant Neural Network (CINN)

After the cepstroid has been computed, we use it to derive the hop size h for our grid in each ME, at which we will subsample the input vectors of the network. By setting h to an appropriate multiple of the cepstroid, the input vectors of songs with different tempo (but potentially a similar rhythmical structure) will be synchronized; and the network can therefore make use of the same neural activation patterns for MEs of different tempi. This enables the CINN to easily identify distinct rhythmical patterns (similar to the ability of a human listener). We want a hop size between approximately 50-100 ms, and therefore compute which duple ratio of 70 ms that is closest to the current cepstroid

$$\min_{n=\cdots,-2,-1,0,1,2,\cdots} \left| \log_2 \frac{70}{\hat{C}/2^n} \right| \tag{5}$$

The value of n, which minimizes the function above, is then used to calculate the hop size h of the ME by

$$h = \frac{\hat{C}}{2^n} \tag{6}$$

The rather coarse hop size (50-100 ms) is used as we wish to include features from several seconds of audio, without the input layer becoming too large. However, to make the network aware of peaks that slips through the coarse grid, we perform a peak picking on the vector $\overrightarrow{P_1'}$, which we have first computed by summing P_1' across frequency. For each grid position, we write the magnitude of the closest peak, the absolute distance to the closest peak, as well as the sign of the computed distance to three feature vectors that we will denote by $\hat{P}$.

Just as for the speed features described in Section 2.4, we filter the feature matrices P_1', S' and V_O' with triangular filters to extract feature vectors. In summary, for each grid position, we extract features by interpolating over the 16 feature vectors defined in Table 3.

Feature	P'_1	$\hat{P}$	S'	V_O
Number of bands/features	6	3	6	1

Table 3. Feature vectors that are interpolated to the grid defined by the cepstroid.

For each frame we try to estimate if it corresponds to a beat (1) or not (0). We include 38 grid-points in each direction from the current frame position, resulting in a time window of $2 \cdot h \cdot 38$ seconds. At $h = 70$ ms, the time window is approximately 5.3 seconds. The computed beat activation from the CINN will be denoted as the beat vector $\vec{B}$ in the subsequent processing.

2.8 Speed Neural Network

Octave errors are a prevalent problem in tempo estimation and beat tracking, and different methods for choosing the correct tempo octave have previously been proposed [13]. It was recently shown that a *continuous* measure of the speed of the music can be very effective at alleviating octave errors [11]. We therefore compute a continuous speed estimate, which guides our tempo estimation, using the input features described in Section 2.4. The ground truth annotation of speed A_s, is derived from the logarithm of the annotated beat length AB_l

$$A_s = \log_2 AB_l \tag{7}$$

Eqn (7) is motivated by our logarithmic perception of tempo [6]. As we have very few annotations (1 per ME), we increase the generalization capabilities with ensemble learning. We also use an inner cross validation (5-fold) for the training set. If this is not done, the subsequent tempo network will overestimate the relevance of the computed speed, rendering a decrease in test performance.

2.9 Tempo Neural Network

The tempo is estimated by finding tempo candidates, and letting the neural network perform a classification between extracted candidates to pick the most likely tempo. First, the candidates are extracted by creating a histogram $H_{\vec{B}}$ of the beat vector $\vec{B}$ (that we previously extracted with the CINN). The energy at each histogram bin is computed as the sum of the product of the magnitudes of the frames of $\vec{B}$ at the frame offset given by a

$$\forall a \in \{1, \cdots, 1900\} \qquad \sum_i \vec{B}_i \cdot \vec{B}_{i+a} \tag{8}$$

We process the histogram to extract a cepstroid vector $C_{\vec{B}}$, by using the same processing scheme as described for C_P in Section 2.3. Peaks are then extracted in both $H_{\vec{B}}$ and $C_{\vec{B}}$, and the corresponding beat length of the histogram peaks are used as tempo candidates.

The neural network is not directly trained to classify if a tempo candidate is correct or incorrect. Instead, to create training data, each possible pair of tempo candidates are examined, and the network is trained to classify which of the two candidates in the pair that correspond to the correct tempo (using only pairs with one correct candidate for the training data).

For testing, the tempo candidate that receives the highest probability in its match-ups against the other candidates is picked as the tempo estimate. This idea was first described in [11] (in that case without using any preceding beat tracking and using a logistic regression without ensemble learning). Input features are defined for both tempo candidates in the pair by their corresponding beat length B_l. We compute:

- The magnitude at B_l in $H_{\vec{B}}$, $C_{\vec{B}}$ and in the feature vectors used for the Cep NN (see Section 2.6). We include octave ratios as defined in Eqn (4).
- We compute $x = \log_2 B_l - Speed$. Then $\mathrm{sgn}(x)$ and $|x|$ are used as features.
- A Boolean vector for all musically relevant ratios defined in Eqn (4), where the corresponding index is 1 if the pair of tempo candidates have that ratio.

We constrain possible tempo candidates to the range 23-270 BPM. This range is a bit excessive for the given datasets, but will allow the system to generalize better to other types of music with more extreme tempi.

2.10 Phase Estimation

At the final stage, we detect the phase of the beat vector and estimate the beat positions. The tempo often drifts slightly in music, for example during performances by live musicians. To model this in a robust way, we compute the CQT of the beat vector. The result is a spectrogram where each frequency corresponds to a particular tempo, the magnitude corresponds to beat strength, and where the phase corresponds to the phase of the beat at specific time positions. The beat vector is upsampled (100 times higher resolution) prior to applying the CQT, and we use 60 bins per octave. We filter the spectrogram with a Hann window of width one tempo octave (60 bins), centered at the frequency that corresponds to the previously computed tempo. As a result, any magnitudes outside of the correct tempo octave are set to 0 in the spectrogram. When the inverse CQT (ICQT) is finally applied to the filtered spectrogram, the result is a beat vector which resembles a sinusoid, where the peaks correspond to tentative beat positions. With this processing technique we have jointly estimated the phase and drift, using a fast transform which seems to be suitable for beat tracking.

The beat estimations are finally refined slightly by comparing the peaks of the computed sinusoidal beat vector with the peaks of the original beat vector from the CINN. Let us define a grid i, consisting of 100 points, onto which we interpolate phase deviations that are within $\pm$ 40 % of the estimated beat length. We then create a "driftogram" M by evaluating each estimated beat position j, adding 1 to each drift position $M_{i,j}$ where a peak was found in the original beat vector. The driftogram is smoothed with a Hann window of size 17 across the beat direction and size 27 across the drift direction. To adjust the beat position, we use the the maximum value for each beat frame of M.

3. EVALUATION

3.1 Datasets

We used the three datasets defined in Table 4 to evaluate our system. The Ballroom datasets consist of ballroom dance music and was annotated by [20, 24]. The Hainsworth dataset [21] is comprised of varying genres, and

the SMC dataset [22] consists of MEs that were chosen based on their difficulty and ambiguity. Tempo annotations were computed by picking the highest peak of a smoothed histogram of the annotated inter-beat intervals.

Dataset	Number of MEs	Length
Ballroom	698	6h 4m
Hainsworth	222	3h 20m
SMC	217	2h 25m

Table 4. Datasets used for evaluation, and their size.

3.2 Evaluation Metrics

There are several different metrics for beat tracking, all trying to capture different relevant aspects of the performance. For an extensive review of different evaluation metrics, we refer the reader to [7].

F-measure is calculated from *Recall* and *Accuracy*, using a limit of ± 70 ms for the beat positions. **P-Score** measures the correlation between annotations and detections. **CMLc** is derived by finding the longest *Correct Metrical Level* with *continuity* required and **CMLt** is similar to CMLc but does not require continuity. **AMLc** is derived by finding the longest *Allowed Metrical Level* with continuity required. This measure allows for several different metrical levels and off-beats. **AMLt** is Similar as AMLc but does not require continuity. The standard tempo estimation metric $\mathbf{Acc_1}$ was computed from the output of the Tempo Network. It corresponds the fraction of MEs that are within 8 % of the annotated tempo.

3.3 Evaluation procedure

We used a 5-fold cross validation to evaluate the system on the Ballroom dataset. More specifically, the training fold was used to train all the different neural networks of the system. After all networks were trained, the test fold was evaluated on the complete system and the results returned. Then the procedure was repeated with the next train/test-split. The Hainsworth and SMC datasets were evaluated by running the MEs on a system previously trained on the complete Ballroom dataset.

As a benchmark for our cross-fold validation results on the Ballroom dataset, we use the cross-fold validation results of the state-of-the-art systems for tempo estimation [5], and beat tracking [25]. The systems were evaluated on a song-by-song basis with data provided by the authors. To make statistical tests we use bootstrapping for paired samples, with a significance level of $p < 0.01$. For the Hainsworth and SMC dataset, benchmarking is most appropriate with systems that were trained on separate training sets. We use [16] as a benchmark for tempo estimation, and [8] as a benchmark for beat tracking.

4. RESULTS

4.1 Tempo

The tempo estimation results (Acc_1), are shown in Table 5, together with the results of the benchmarks.

(Acc_1)	Ballroom	Hainsworth	SMC
Proposed	**0.973***	**0.802**	**0.332**
Böck [5]	**0.947***	0.865*	0.576*
Gkiokas [16]	0.625	**0.667**	**0.346**

Table 5. The results for our tempo estimation system in comparison with the benchmarks. Results marked with (*) were obtained from cross-fold validation. Results in bold are most relevant to compare. Statistical significance for systems with song-by-song data in comparison with the proposed system is underlined.

4.2 Beat tracking

Table 6 shows the performance of the system, evaluated as described in Section 3.2.

Ballroom	F-Me	P-Sc	CMLc	CMLt	AMLc	AMLt
Proposed	**92.5***	**92.2***	**86.8***	**90.3***	**89.4***	**93.2***
Krebs [25]	**91.6***	**88.8***	**83.6***	**85.1***	**90.4***	**92.2***
Hainsworth						
Proposed	**74.2**	**77.7**	**57.6**	**67.6**	**65.0**	**79.2**
Davies [8]	**-**	**-**	**54.8**	**61.2**	**68.1**	**78.9**
SMC						
Proposed	37.5	49.4	14.9	22.5	20.9	33.2

Table 6. The results for our proposed system in comparison with the benchmarks. Results marked with (*) were obtained from a cross-fold validation. Statistical significance for systems with song-by-song data in comparison with the proposed system is underlined.

5. SUMMARY & CONCLUSIONS

We have presented a novel beat tracking and tempo estimation system that uses a cepstroid invariant neural network. The many connected networks make it possible to explicitly capture different aspects of rhythm. With a Cep network we compute a salient level of repetition of the music. The invariant representations that were computed by subsampling the feature vectors allowed us to obtain an accurate beat vector in a CINN. By applying the CQT to the beat vector, and then filtering the spectrogram to keep only magnitudes that corresponds to the estimated tempo before applying the ICQT, we computed the phase of the beat. Alternative post processing strategies, such as applying a DBN on the beat vector, could potentially improve the performance. The results are comparable to the benchmarks both for tempo estimation and beat tracking. This indicates that the ideas put forward in this paper are important, and we hope that they can inspire new network architectures for MIR. Tests on hidden datasets for the relevant MIREX tasks would be useful to draw further conclusion regarding the performance.

6. ACKNOWLEDGEMENTS

Thanks to Anders Friberg for helpful discussions as well as proofreading. This work was supported by the Swedish Research Council, Grant Nr. 2012-4685.

7. REFERENCES

[1] E. Benetos: “Automatic transcription of polyphonic music exploiting temporal evolution,” *Dissertation*. Queen Mary, University of London, 2012.

[2] S. Böck and M. Schedl: “Enhanced beat tracking with context aware neural networks,” In *Proc. of DAFx,* 2011.

[3] S. Böck, A. Arzt, F. Krebs, and M. Sched: “Online real-time onset detection with recurrent neural networks,” In *Proc. of DAFx,* 2012.

[4] S. Böck, F. Krebs, and G. Widmer: "A Multi-model Approach to Beat Tracking Considering Heterogeneous Music Styles," In *Proc. of ISMIR*, 2014.

[5] S. Böck, F. Krebs, and G. Widmer: “Accurate tempo estimation based on recurrent neural networks and resonating comb filters,” In *Proc. of ISMI*R, pp. 625-631, 2015.

[6] A. T. Cemgil, B. Kappen, P. Desain, and H. Honing: ”On tempo tracking: Tempogram Representation and Kalman filtering,” *J. New Music Research*, Vol. 29, No. 4, pp. 259-273, 2000.

[7] M. E. P. Davies, N. Degara, and M. D. Plumbley: “Evaluation methods for musical audio beat tracking algorithms,” Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06, 2009.

[8] M. Davies and M. Plumbley: “Context-dependent beat tracking of musical audio,” *IEEE Trans on Audio, Speech and Language Processing*, Vol. 15, No. 3, pp. 1009–1020, 2007.

[9] S. Dixon: “Evaluation of audio beat tracking system beatroot,” *J. of New Music Research,* Vol. 36, No. 1, pp. 39–51, 2007.

[10] D. Eck. “Beat tracking using an autocorrelation phase matrix,” In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007),* Vol. 4, pp. 1313–1316, 2007.

[11] A. Elowsson and A. Friberg: "Modeling the perception of tempo," *J. of Acoustical Society of America,* Vol. 137, No. 6, pp. 3163-3177, 2015.

[12] A. Elowsson and A. Friberg: “Modelling perception of speed in music audio,” *Proc. of SMC*, pp. 735-741, 2013.

[13] A. Elowsson, A. Friberg, G. Madison, and J. Paulin: “Modelling the speed of music using features from harmonic/percussive separated audio,” *Proc. of ISMIR,* pp. 481-486, 2013.

[14] A. Elowsson and A. Friberg: “Polyphonic Transcription with Deep Layered Learning,” MIREX Multiple Fundamental Frequency Estimation & Tracking, 2 pages, 2014.

[15] D. FitzGerald: “Harmonic/percussive separation using median filtering,” *Proc. of DAFx-10,* 4 pages, 2010.

[16] A. Gkiokas, V. Katsouros, G. Carayannis, and T. Stafylakis: “Music tempo estimation and beat tracking by applying source separation and metrical relations,” In Proc. of ICASSP, pp. 421–424, 2012.

[17] X. Glorot, Xavier, A. Bordes, and Y. Bengio: "Deep sparse rectifier neural networks," *International Conference on Artificial Intelligence and Statistics,* 2011.

[18] M. Goto and Y. Muraoka: “Music understanding at the beat level real-time beat tracking for audio signals,” in *Proc. of IJCAI (Int. Joint Conf. on AI) / Workshop on CASA*, pp. 68–75, 1995.

[19] M. Goto and Y. Muraoka: “Beat tracking based on multiple agent architecture a real-time beat tracking system for audio signals,” In *Proc. of the International Conference on Multiagent Systems*, pp. 103–110, 1996.

[20] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano: “An experimental comparison of audio tempo induction algorithms,” *IEEE Trans. on Audio, Speech and Language Processing,* Vol. 14, No. 5, pp. 1832-1844, 2006.

[21] S. Hainsworth and M. Macleod: “Particle filtering applied to musical tempo tracking,” *EURASIP J. on Applied Signal Processing*, Vol. 15, pp 2385–2395, 2004.

[22] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon: “Selective sampling for beat tracking evaluation,” *IEEE Trans. on Audio, Speech, and Language Processing*, Vol. 20, No. 9, pp. 2539–2548, 2012.

[23] A. Klapuri, A. Eronen, and J. Astola: “Analysis of the meter of acoustic musical signals,” *IEEE Trans. on Audio, Speech and Language Processing,* Vol. 14, No. 1, pp. 342–355, 2006.

[24] F. Krebs, S. Böck, and G. Widmer: “Rhythmic pattern modeling for beat and downbeat tracking in musical audio,” In *Proc. of ISMIR*, pp. 227–232, Curitiba, Brazil, November 2013.

[25] F. Krebs, S. Böck, and G. Widmer: “An Efficient State-Space Model for Joint Tempo and Meter Tracking,” In *Proc. of ISMI*R, pp. 72-78, 2015.

[26] G. Peeters and H. Papadopoulos: “Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation,” *IEEE Trans. on Audio, Speech, and Language Processing,* Vol. 19, No. 6, pp. 1754–1769, 2011.

[27] R. Polikar: “Ensemble based systems in decision making,” *Circuits and Systems Magazine, IEEE,* Vol. 6, No. 3, pp. 21-45, 2006.

[28] E. Scheirer: “Tempo and beat analysis of acoustic musical signals,” *J. Acoust. Soc. Am.,* Vol. 103, No. 1, pp. 588–601, 1998.

[29] J. Schlüter, and S. Böck: "Musical onset detection with convolutional neural networks," In *6th International Workshop on Machine Learning and Music (MML)*, Prague, Czech Republic. 2013.

[30] N. Whiteley, A. Cemgil, and S. Godsill: “Bayesian modelling of temporal structure in musical audio,” In *Proc. of ISMIR,* pp. 29–34, 2006.

BOOTSTRAPPING A SYSTEM FOR PHONEME RECOGNITION AND KEYWORD SPOTTING IN UNACCOMPANIED SINGING

Anna M. Kruspe
Fraunhofer IDMT, Ilmenau, Germany
kpe@idmt.fraunhofer.de

ABSTRACT

Speech recognition in singing is still a largely unsolved problem. Acoustic models trained on speech usually produce unsatisfactory results when used for phoneme recognition in singing. On the flipside, there is no phonetically annotated singing data set that could be used to train more accurate acoustic models for this task.

In this paper, we attempt to solve this problem using the *DAMP* data set which contains a large number of recordings of amateur singing in good quality. We first align them to the matching textual lyrics using an acoustic model trained on speech.

We then use the resulting phoneme alignment to train new acoustic models using only subsets of the *DAMP* singing data. These models are then tested for phoneme recognition and, on top of that, keyword spotting. Evaluation is performed for different subsets of *DAMP* and for an unrelated set of the vocal tracks of commercial pop songs. Results are compared to those obtained with acoustic models trained on the *TIMIT* speech data set and on a version of *TIMIT* augmented for singing. Our new approach shows significant improvements over both.

1. INTRODUCTION

Automatic speech recognition encompasses a large variety of research topics, but the developed algorithms have so far rarely been adapted to singing. Most of these tasks become harder when used on singing because singing data has different characteristics, which are also often more varied than in pure speech [12] [2]. For example, the typical fundamental frequency for women in speech is between 165 and $200Hz$, while in singing it can reach more than $1000Hz$. Other differences include harmonics, durations, pronunciation, and vibrato.

Speech recognition in singing can be used in many interesting practical applications, such as automatic lyrics-to-music alignment, keyword spotting in songs, language identification of musical pieces or lyrics transcription.

A first step in many of these tasks is the recognition of phonemes in the audio recording. We showed in [9] that phoneme recognition is a bottleneck in tasks such as language identification and keyword spotting in singing. Other publications also demonstrate that phoneme recognition on singing is more difficult than on speech [15] [5] [12]. This is further compounded by the fact that models are usually trained on pure speech data.

As shown on a small scale in [5] and [9], recognition gets better when singing is used as part of the training data. This has so far not been done comprehensively due to the lack of singing data sets annotated with phonemes or words.

In this paper, we present a new approach to training acoustic models on actual singing data. This is done by first assembling the data from a set of recordings of unaccompanied singing and the matching textual lyrics. These lyrics are then automatically aligned to the audio data using models trained solely on speech. Next, the resulting annotated data sets are used to train new acoustic models for phoneme recognition in singing. We then evaluate the phoneme recognition results on different subsets of the singing corpus and on an unrelated data set of vocal tracks. Finally, we also use the recognized phonemes to perform keyword spotting.

This paper is structured as follows: We first present the state of the art in section 2 and the data sets in section 3. Then, we describe our proposed approach in more detail in section 4. The experiments and their results are presented in sections 5 and 6. Finally, we give a conclusion in section 7 and make suggestions for future experiments in section 8.

2. STATE OF THE ART

2.1 Phoneme recognition in singing

As described in [12], [2], and [9], there are significant differences between speech and singing audio, such as pitch and harmonics, vibrato, phoneme durations and pronunciation. These factors make phoneme recognition on singing more difficult than on speech. It has only been a topic of research for the past few years.

Fujihara et al. first presented an approach using Probabilistic Spectral Templates to model phonemes in [3]. The phoneme models are gender-specific and only model five vowels, but also work for singing with instrumental accompaniment. The best result is 65% correctly classified frames.

In [4], Gruhne et al. describe a classical approach that employs feature extraction and various machine learning

© Anna M. Kruspe. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Anna M. Kruspe. "Bootstrapping a system for phoneme recognition and keyword spotting in unaccompanied singing", 17th International Society for Music Information Retrieval Conference, 2016.

algorithms to classify singing into 15 phoneme classes. It also includes a step that removes non-harmonic components from the signal. The best result of 58% correctly classified frames is achieved with Support Vector Machine (SVM) classifiers. The approach is expanded upon in [17].

Mesaros presented a complex approach that is based on Hidden Markov Models which are trained on Mel-Frequency Cepstral Coefficients (MFCCs) and then adapted to singing using three phoneme classes separately [15] [14]. The approach also employs language modeling and has options for vocal separation and gender and voice adaptation. The achieved phoneme error rate on unaccompanied singing is 1.06 without adaptation and 0.8 with singing adaptation using 40 phonemes (the error rate greater than one means that there were more insertion, deletion, or substitution errors than phoneme instances). The results also improve when using gender-specific adaptation (to an average of 0.81%) and even more when language modeling is included (to 0.67%).

Hansen presents a system in [5] which combines the results of two Multilayer Perceptrons (MLPs), one using MFCC features and one using TRAP (Temporal Pattern) features. Training is done with a small amount of singing data. Viterbi decoding is then performed on the resulting posterior probabilities. On a set of 27 phonemes, this approach achieves a recall of up to 48%.

Finally, we trained new models for phoneme recognition in singing by modifying speech data to make it more "song-like" [11]. We employed time-stretching, pitch-shifting, and vibrato generation algorithms. Using a model trained on speech data with all three modifications, we obtained 18% correctly classified frames (6% improvement) and a weighted phoneme error rate of 0.71 (0.06 improvement).

Generally, comparing the existing approaches is not trivial since different datasets, different phoneme sets, and different evaluation measures are used.

2.2 Keyword spotting in singing

A first approach to keyword spotting in singing was presented in [9]. This approach employs keyword-filler HMMs which detect the keyword. The recognition is performed on phoneme posteriorgrams, which were generated with acoustic models trained on speech. We obtained F_1 measures of 33% for spoken lyrics and 24% for a-capella singing. Using post-processing techniques on the posteriorgrams, the a-capella result was improved up to 27%.

In [10], we improved upon this result by employing phoneme duration modeling algorithms. The best result on a-capella singing was an F_1 measure of 39%.

In [1], HMM models and position-HMM-DBNs were employed to search for certain phrases of lyrics in traditional Turkish music. The approach obtained an F_1 measure of 13% for the 1-best result.

3. DATA SETS

3.1 Speech data sets

For training our baseline phoneme recognition models, we used the well-known *Timit* speech data set [7]. Its training section consists of 4620 phoneme-annotated English utterances spoken by native speakers. Each utterance is a few seconds long.

Additionally, we also trained phoneme models on a modification of *Timit* where pitch-shifting, time-stretching, and vibrato were applied to the audio data. This process was described in [11]. The data set will be referred to as *TimitM*.

3.2 Singing data sets

3.2.1 Damp

For training models specific to singing, we used the *DAMP* data set, which is freely available from Stanford University [1] [16]. This data set contains more than 34,000 recordings of amateur singing of full songs with no background music, which were obtained from the *Smule Sing!* karaoke app. Each performance is labeled with metadata such as the gender of the singer, the region of origin, the song title, etc. The singers performed 301 English language pop songs. The recordings have good sound quality with little background noise, but come from a lot of different recording conditions.

No lyrics annotations are available for this data set, but we obtained the textual lyrics from the *Smule Sing!* website [2]. These were, however, not aligned in any way. We performed such an alignment on the word and phoneme levels automatically (see section 4.1).

Out of all those recordings, we created several different sub-data sets:

DampB Contains 20 full recordings per song (6000 in sum), both male and female.

DampBB Same as before, but phoneme instances were discarded until they were balanced and a maximum of 250,000 frames per phoneme where left, where possible. This data set is about 4% the size of *DampB*.

DampBB_small Same as before, but phoneme instances were discarded until they were balanced and 60,000 frames per phoneme were left (a bit fewer than the amount contained in *Timit*). This data set is about half the size of *DampBB*.

DampFB and DampMB Using 20 full recordings per song and gender (6000 each), these data sets were then reduced in the same way as *DampBB*. *DampFB* is roughly the same size, *DampMB* is a bit smaller because there are fewer male recordings.

DampTestF and DampTestM Contains one full recording per song and gender (300 each). These data sets were used for testing. There is no overlap with any of the training data sets.

[1] https://ccrma.stanford.edu/damp/

[2] http://www.smule.com/songs

#	Keywords
2	eyes
3	love, away, time, life, night
4	never, baby, world, think, heart, only, every
5	always, little

Table 1: All 15 tested keywords, ordered by number of phonemes.

Order-13 MFCCs plus deltas and double-deltas were extracted from all data sets and used in all experiments.

3.2.2 Acap

We also ran some tests on a small data set of the vocal tracks of 15 pop songs, which were hand-annotated with phonemes and words. This data set was first presented in [5]. Despite the small size, we provide results on this data set for comparison with our previous approaches, and because the ground truth annotations can be assumed to be correct (in contrast with the automatically generated annotations of the *Damp*-based data sets).

3.3 Keywords

From the 301 different song lyrics of the *Damp* data sets, 15 keywords were chosen by semantic content and frequency to test our keyword spotting algorithm. Each keyword occurs in at least 50 of the 301 songs. The keywords are shown in table 1.

4. PROPOSED APPROACH

4.1 Lyrics alignment

Since the textual lyrics were not aligned to the singing audio data, we first performed an automatic alignment step. A monophone HMM acoustic model trained on *Timit* using HTK was used. Alignment was performed on the word and phoneme levels. This is the same principle of so-called "Forced Alignment" that is commonly used in Automatic Speech Recognition [8] (although it is commonly done on shorter utterances). We hand-checked some examples and found the alignment to already be very good over-all. Of course, errors cannot be avoided when doing automatic forced alignment. We considered repeating this process with the newly trained models, but preliminary tests suggested that this would not improve the alignments very much.

The resulting annotations were used in the following experiments. This approach provided us with a large amount of annotated singing data, which could not feasibly have been done manually.

4.2 New acoustic models

Using these automatically generated annotations, we then trained new acoustic models on *DampB*, *DampBB*, *DampBB_small*, *DampFB*, and *DampMB*. Models were also trained on *Timit* and *TimitM*.

All models are DNNs with three hidden layers of 1024, 850, and again 1024 dimensions with a sigmoid activation function. The output layer corresponds to 37 monophones. Inputs are either frame-wise MFCCs (39 dimensions) or MFCCs with 4 context frames on either side (351 dimensions).

4.3 Phoneme recognition and evaluation

Using these models, phoneme posteriorgrams were then generated on the test data sets (*DampTestF*, *DampTestM*, and *Acap*). For all non-gender dependent acoustic models, results over both of the *DampTest* sets were averaged.

The recognized phonemes were then evaluated using the percentage of correct frames, the phoneme error rate, and the weighted phoneme error rate as evaluation measures (see [11]). In the case of the *DampTest* data sets, the results were compared to the automatic alignment results, which is a potential source of error.

4.4 Keyword spotting and evaluation

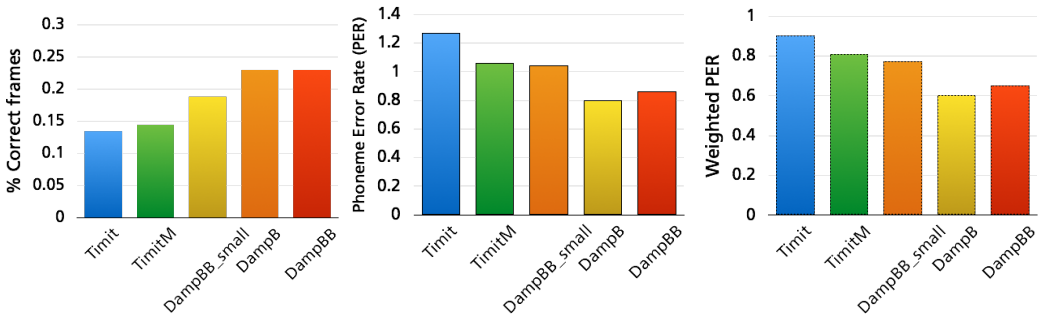
Finally, the phoneme posteriorgrams were evaluated for keyword spotting. A keyword-filler HMM algorithm was employed. Keyword-filler HMMs consist of two sub-HMMs: One to model the keyword and one to model everything else (=filler). The keyword HMM has a simple left-to-right topology with one state per keyword phoneme. The filler HMM is a fully connected loop of all phonemes. When the Viterbi path with the highest likelihood passes through the keyword HMM rather than the filler loop, the keyword is detected. We previously employed this approach in [9]. However, the evaluation data set based on *Damp* is a different, much bigger set of recordings. The keyword set was changed to better reflect frequently occurring words in these songs. Additionally, the keyword detection was performed on whole songs, which may be more realistic for practical applications. For comparison, results on our old data set (*Acap*) for whole songs are also provided below. Song-wise F_1 measures were calculated for evaluation.

5. PHONEME RECOGNITION EXPERIMENTS AND RESULTS

5.1 Experiment A: Comparison of models trained on *Timit* and *Damp* data sets

In our first experiments, we generated phoneme posteriorgrams on the data sets *DampTestF* and *DampTestM* using the models trained on the two variants of *Timit* and on the three differently-sized *Damp* training sets that were not split by gender. The results are averaged over both sets. For comparison, we also generated these posteriorgrams on *Acap*. The results for the *DampTest* sets in terms of percentage of correct frames, phoneme error rate, and weighted phoneme error rate are shown in figure 1.

Models trained on the modified version of *Timit* already show some improvement over plain *Timit* [11], but even the small *Damp* training set improves the result significantly more. As mentioned before, this data set is actually smaller than *Timit*. The percentage of correct frames rises



(a) Percentage of correct frames (b) Phoneme error rate (c) Weighted phoneme error rate

Figure 1: Evaluation measures for the results obtained on the *DampTest* data sets using models trained on *Timit* and on various *Damp*-based data sets.

from 13% to 19%, the phoneme error rate sinks from 1.27 to 1.04, and the weighted phoneme error rate from 0.9 to 0.77.

When the whole set of 6000 recordings is used for training (*DampB*), the percentage of correct frames even rises to 23%, while the phoneme error rate falls to 0.8 and the weighted phoneme error rate to 0.6. When using the smaller, more balanced version (*DampBB*), these results are somewhat worse, but not much, with 23% correct frames, a phoneme error rate of .86, and a weighted phoneme error rate of 0.65. This is particularly interesting because this data set is only 4% the size of the bigger one and training is therefore much faster.

The results on the *Acap* data set show a similar improvement, but are better in general. The percentage of correctly classified frames jumps from 12% to 22%, 25%, and 27% for *DampBB_small*, *DampB*, and *DampBB* respectively. The weighted phoneme error rate sinks from 0.8 to 0.69, 0.61, and 0.56. Since this data set is has been annotated by hand and is completely different material from the training data sets, we are confident that our approach is able to model the properties of each phoneme, rather than reproducing the model that was used for aligning the singing training sets. The somewhat better values might be caused by these more accurate annotations, too, or by the fact that these are recordings of professional singers who enunciate more clearly.

5.2 Experiment B: Influence of context frames

We then ran the same experiment again, but this time used models that were trained with 4 context frames on either side of each input frame. This provides more long-term information. The results are shown in figure 2. (In each figure, the "No context" part is the result from the previous experiment).

Surprisingly, using context frames did not improve the result in any case except for the *DampBB_small* models. Since this is the smallest data set, this improvement might happen just because the context frames virtually provide more training data for each phoneme. In the other cases, there already seems to be a sufficient amount of training data and the context frames may blur the training data instead of providing more information about the context of each phoneme. Additionally, it is possible that this approach compounds error that were made in the automatic alignment in the case of the bigger *Damp* training data sets.

The same effect can be observed when calculating these values on the hand-annotated *Acap* test data set. We therefore decided to not employ context frames in the following experiments. This also speeds up the training process.

5.3 Experiment C: Comparison of gender-dependent models

Finally, we generated phoneme posteriorgrams using models that were only trained on recordings of the same gender. I.e., for phoneme recognition on the *DampTestF* set, we used a model trained only on female singing recordings (*DampFB*). The results are shown in figure 3. (Note that the results for *DampB* and *DampBB* are different from the previous experiments because the test data sets were split by gender).

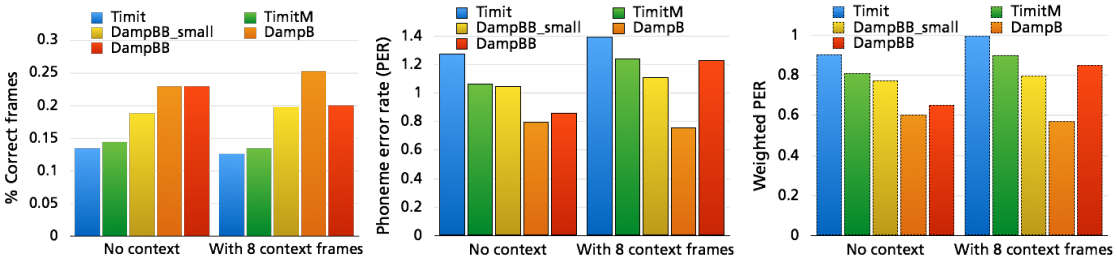
Surprisingly, the results do not improve when using gender-specific acoustic models. The percentage of correct frames, when compared to the results using the models trained on the *DampBB* drops sligthly from 23% to 21% for the female test set, and stays at 23% for the male one. The weighted phoneme error rate rises from 0.65 to 0.68 and from 0.65 to 0.69 for the female and male test sets respectively.

This might happen because the training data sets are slightly smaller, but, more likely, because some variation in the singing voices might be lost when using training data of only one gender. In singing, pitches cover a broader range than in speech. This effect might take away some of the improvement usually seen in speech recognition when using gender-specific models.

6. KEYWORD SPOTTING EXPERIMENTS AND RESULTS

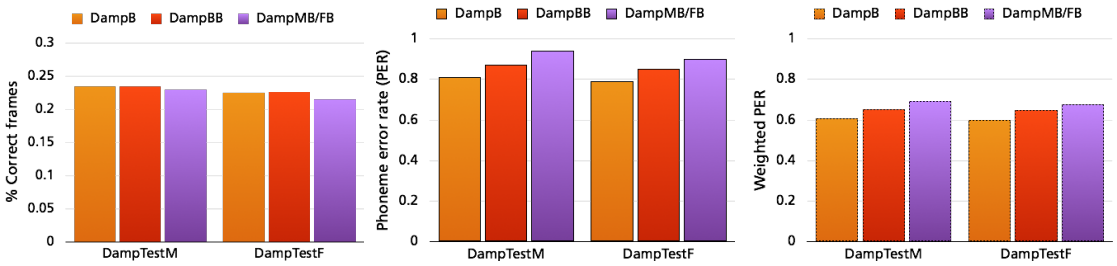
6.1 Experiment D: Comparison of models trained on *Timit* and *Damp* data sets

We then performed keyword spotting on the phoneme posteriorgrams from Experiment A. The results in terms of F_1



(a) Percentage of correct frames (b) Phoneme error rate (c) Weighted phoneme error rate

Figure 2: Evaluation measures for the results obtained on the *DampTest* data sets using models trained on *Timit* and on various *Damp*-based data sets with no context and with 8 context frames.



(a) Percentage of correct frames (b) Phoneme error rate (c) Weighted phoneme error rate

Figure 3: Evaluation measures for the results obtained on the *DampTestM* and *DampTestF* data sets using models trained on *Damp*-based data sets, mixed and split by gender.

measure across the whole *DampTest* sets are shown in figure 4a. Figure 4b show the results of the same experiment on the small *Acap* data set.

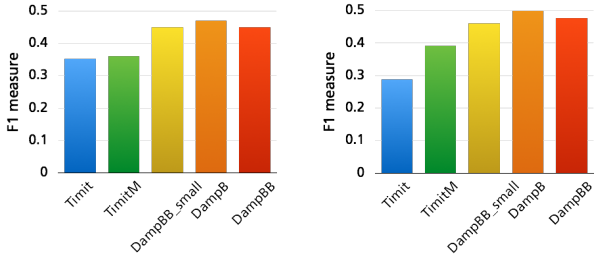
Across all keywords, we obtain a document-wise F_1 measure of 0.35 using the posteriorgrams generated with the *Timit* model on the *DampTest* data sets. This result is slightly higher for the *TimitM* models and rises to 0.45 using the model trained on the small *DampBB_small* singing data set. Surprisingly, the model trained on *DampBB* is only slightly better than the much smaller one. Using the very big *DampB* training data set, the F_1 measure reaches 0.47.

On the hand-annotated *Acap* test set, the difference is even more pronounced, rising from 0.29 for the *Timit* model to 0.5 for *DampB*. This might, again, be caused by the more accurate annotations or by the higher-quality singing. Additionally, the data set is much smaller with fewer occurrences of each keyword, which could emphasize both positive and negative tendencies in the detection.

6.2 Experiment E: Comparison of gender-dependent models

We also performed keyword spotting on the posteriorgrams generated with the gender-dependent models from Experiment C. The results are shown in figure 5.

In contrast to the phoneme recognition resultes from Experiment C, the gender-dependent models perform slightly better for keyword spotting than the mixed one of the same size, and almost as good as the one trained on much more data (*DampB*). The F_1 measures for the female test set are 0.48 for the *DampB* model, 0.45 for the *DampBB* model, and 0.46 for the *DampFB* model. For the male test set, they are 0.46 and 0.45 for the first two, and 0.46 for the *DampMB* model.



(a) F_1 measures for keyword spotting results on the *DampTest* data sets.

(b) Keyword spotting results on the *Acap* data set.

Figure 4: F_1 measures for keyword spotting results using posteriorgrams generated with various acoustic models.

6.3 Experiment F: Individual analysis of keyword results

Figure 6 shows the individual F_1 measures for each keyword using the best model (*DampB*), ordered by their occurrence in the *DampTest* sets from high to low (i.e. number of songs which include the song). There appears to be a tendency for more frequent keywords to be detected more accurately. This happens because a high recall is often achievable, while the precision depends very much on the accuracy of the input posteriorgrams. The more frequent a keyword, the easier it also becomes to achieve a higher precision for it.

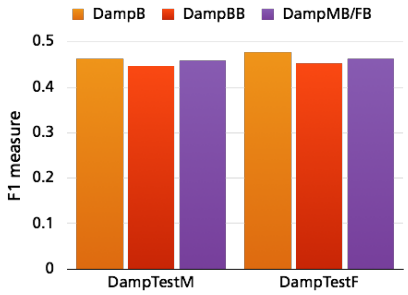


Figure 5: F_1 measures for keyword spotting results on the *DampTestM* and *DampTestF* data sets using mixed and gender-dependent models.

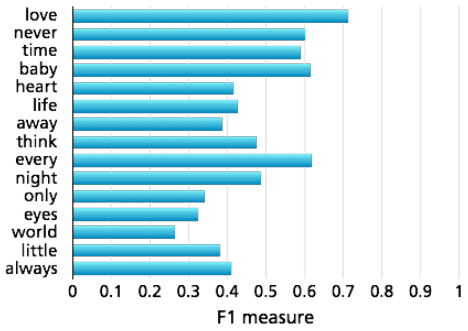


Figure 6: Individual F_1 measures for the results for each keyword, using the acoustic model trained on *DampB*.

As shown in literature [18], the detection accuracy also depends on the length of the keyword: Keywords with more phonemes are usually easier to detect. This might explain the relative peak for "every", "little", and "always", in contrast to "eyes" or "world". Since keyword detection systems tend to perform better for longer words and most of our keywords only have 3 or 4 phonemes, this result is especially interesting.

One potential source of error are sequences of phonemes that overlap with our keywords, but are not included in the calculation of the precision. Equally spelled words were included, but split phrases or other spellings were not (e.g. "away" as part of "castaway" would be counted, but "a way" would not be counted as "away"). This might artificially lower our results and we will look into possibilities for improvement in the future. Additionally, only one pronunciation for each keyword was provided, but there may be several possible.

7. CONCLUSION

In this paper, we trained new acoustic models on a large corpus of unaccompanied singing recordings. Since no annotations for these existed, we first had to automatically align lyrics to them. The new models could then directly be trained on these automatic annotations. To our knowledge, this has not been done before for singing.

We trained three different models with mixed gender recordings: One on 6000 full recordings of 301 songs, one on just 4% of this data, and one which was balanced by phonemes and is roughly half the size of the medium-sized one. We then tested their performance on two other subsets of the same corpus which did not overlap with the training data, and on a small unrelated data set of commercial vocal tracks which were hand-annotated.

In all cases, the three new models showed a strong improvement over those trained only on speech. Even the model trained on the smallest set produced a jump in correctly classified frames from 13% to 19%, and in weighted phoneme error rate from 0.9 to 0.77 on the large test set. With the model trained on the medium-sized data set, we obtained 23% correct frames and a weighted phoneme error rate of 0.65. With the biggest one, the weighted phoneme error rate falls to 0.6. The results are similar on the small hand-annotated test set.

We also tested acoustic models with 8 context frames, and models trained on gender-specific data. Neither of them showed improvement over the first ones.

We then performed keyword spotting for 15 keywords on phoneme posteriorgrams generated with these new models using a keyword-filler approach. The resulting F_1 measure rises from 0.35 for the models trained on speech to 0.47 for our new models. This result is especially interesting because most of the keywords have few phonemes. For keyword spotting, gender-dependent models perform slightly better than mixed-gender models of the same size.

8. FUTURE WORK

So far, we have only tested this approach using MFCC features. As shown in our previous experiments [9], other features like TRAP or PLP may work better on singing. So-called log-mel filterbank features have also been used successfully with DNNs [6]. Another interesting factor is the size and configuration of the classifiers, of which we have only tested one so far. Since the alignment appears to provide valid training data, we believe the features and model configuration could be the greatest sources of improvement.

We showed that there is only a slight amount of improvement between the model trained on all 6000 songs and the one trained only on 4% of this data. It would be interesting to find the exact point at which additional training data does not further improve the models. On the evaluation side, a keyword spotting approach that allows for pronunciation variants or sub-words may produce better results. Language modeling might also help to alleviate some of the errors made during phoneme recognition.

These models have not yet been applied to singing with background music, which would be interesting for practical applications. Since this would probably decrease the result when used on big, unlimited data sets, more specified systems would be more manageable, e.g. for specific music styles, sets of songs, keywords, or specialized applications. Searching for whole phrases instead of short keywords could also make the results better usable in practice.

As shown in [13] and [2], alignment of textual lyrics and singing already works well. A combined approach that also employs textual information could be very practical.

9. REFERENCES

[1] G. Dzhambazov, S. Sentürk, and X. Serra. Searching lyrical phrases in a-capella turkish makam recordings. In *Proceedings of the 16th International Conference on Music Information Retrieval (ISMIR)*, Malaga, Spain, 2015.

[2] H. Fujihara and M. Goto. *Multimodal Music Processing*, chapter Lyrics-to-audio alignment and its applications. Dagstuhl Follow-Ups, 2012.

[3] H. Fujihara, M. Goto, and H. G. Okuno. A novel framework for recognizing phonemes of singing voice in polyphonic music. In *WASPAA*, pages 17–20. IEEE, 2009.

[4] M. Gruhne, K. Schmidt, and C. Dittmar. Phoneme recognition on popular music. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria, September 2007.

[5] J. K. Hansen. Recognition of phonemes in a-cappella recordings using temporal patterns and mel frequency cepstral coefficients. In *9th Sound and Music Computing Conference (SMC)*, pages 494–499, Copenhagen, Denmark, 2012.

[6] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition. *Signal Processing Magazine*, 2012.

[7] J. S. Garofolo et al. TIMIT Acoustic-Phonetic Continuous Speech Corpus. Technical report, Linguistic Data Consortium, Philadelphia, 1993.

[8] D. Jurafsky and J. H. Martin. *Speech and language processing: An introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, 2009.

[9] A. M. Kruspe. Keyword spotting in a-capella singing. In *15th International Conference on Music Information Retrieval (ISMIR)*, Taipei, Taiwan, 2014.

[10] A. M. Kruspe. Keyword spotting in a-capella singing with duration-modeled HMMs. In *EUSIPCO*, Nice, France, 2015.

[11] A. M. Kruspe. Training phoneme models for singing with "songified" speech data. In *15th International Conference on Music Information Retrieval (ISMIR)*, Malaga, Spain, 2015.

[12] A. Loscos, P. Cano, and J. Bonada. Low-delay singing voice alignment to text. In *Proceedings of the ICMC*, 1999.

[13] A. Mesaros and T. Virtanen. Automatic alignment of music audio and lyrics. In *DaFX-08*, Espoo, Finland, 2008.

[14] A. Mesaros and T. Virtanen. Automatic recognition of lyrics in singing. *EURASIP J. Audio, Speech and Music Processing*, 2010, 2010.

[15] A. Mesaros and T. Virtanen. Recognition of phonemes and words in singing. In *ICASSP*, pages 2146–2149. IEEE, 2010.

[16] J. C. Smith. *Correlation analyses of encoded music performance*. PhD thesis, Stanford University, 2013.

[17] G. Szepannek, M. Gruhne, B. Bischl, S. Krey, T. Harczos, F. Klefenz, C. Dittmar, and C. Weihs. *Classification as a tool for research*, chapter Perceptually Based Phoneme Recognition in Popular Music. Springer, Heidelberg, 2010.

[18] A. J. K. Thambiratnam. *Acoustic keyword spotting in speech with applications to data mining*. PhD thesis, Queensland University of Technology, 2005.

CAN MICROBLOGS PREDICT MUSIC CHARTS? AN ANALYSIS OF THE RELATIONSHIP BETWEEN #NOWPLAYING TWEETS AND MUSIC CHARTS

Eva Zangerle, Martin Pichl, Benedikt Hupfauf, Günther Specht
Department of Computer Science
University of Innsbruck, Austria
{eva.zangerle, martin.pichl, benedikt.hupfauf, guenther.specht}@uibk.ac.at

ABSTRACT

Twitter is one of the leading social media platforms, where hundreds of millions of tweets cover a wide range of topics, including the music a user is listening to. Such #nowplaying tweets may serve as an indicator for future charts, however, this has not been thoroughly studied yet. Therefore, we investigate to which extent such tweets correlate with the Billboard Hot 100 charts and whether they allow for music charts prediction. The analysis is based on #nowplaying tweets and the Billboard charts of the years 2014 and 2015. We analyze three different aspects in regards to the time series representing #nowplaying tweets and the Billboard charts: (i) the correlation of Twitter and the Billboard charts, (ii) the temporal relation between those two and (iii) the prediction performance in regards to charts positions of tracks. We find that while there is a mild correlation between tweets and the charts, there is a temporal lag between these two time series for 90% of all tracks. As for the predictive power of Twitter, we find that incorporating Twitter information in a multivariate model results in a significant decrease of both the mean RMSE as well as the variance of rank predictions.

1. INTRODUCTION

The microblogging platform Twitter has long since become one of the leading social media platforms, serving 271 million active users, who publish approximately 500 million tweets every day [3]. On Twitter, users for instance share their opinions about various topics, post interesting articles, let the world know what they are currently doing, and interact with other users. Furthermore, users share what music they are currently listening to in so-called #nowplaying tweets. These tweets can be identified by one of the hashtags #nowplaying, #listento or #listeningto, and typically feature title and artist of the track the user is listening to (e.g. "#NowPlaying Everlong – Foo Fighters http://spoti.fi/J1Gqhs"). The majority of such tweets are automatically generated by music players or music streaming platforms (as in the previous example tweet, Spotify).

#nowplaying tweets have already been analyzed in regards to the listening behavior of users from around the world [20], or to perform user-specific music recommendation tasks [24]. Also, Schedl and Tkalčič looked into the genre distribution within #nowplaying tweets, in particular, the use of social media of classical music enthusiasts [21]. Kim et al. present a prediction analysis of Billboard charts based on Twitter data [13]. The authors crawled #nowplaying tweets and the Billboard Hot 100 over the course of 10 weeks and extracted tweets that contain a song or artist already contained in the charts. The actual prediction was then performed by a classification task based on the number of tweets about the given track and artist, and the number of weeks the track has already been in the Billboard Hot 100. The results indicate that the future success of a track, which is already in the charts, can be predicted accurately. However, we argue that only using those tweets which feature artists or songs currently in the Billboard Hot 100, and using information about how long a certain song has already been in the charts limits this approach and its general applicability. Furthermore, the study by Kim et al. was limited to an analysis period of 10 weeks.

Following up on this research, we are interested in generalizing the hypothesis of chart prediction based on Twitter data, and look into how suitable #nowplaying tweets are when it comes to predicting the Billboard Hot 100. Therefore, we shed light on the following research questions (RQ) in this study:

- **RQ1:** To which extent do #nowplaying-tweets resemble the Billboard Hot 100?
- **RQ2:** How are #nowplaying tweets and the Billboard Hot 100 temporally related?
- **RQ3:** How can Twitter data be exploited for predicting music charts?

In this work, we model the Billboard charts data as well as the Twitter data as time series [10]. Based on this data, we look into three different aspects to answer the research questions: (i) the correlation of Twitter-activity regarding

© Eva Zangerle, Martin Pichl, Benedikt Hupfauf, Günther Specht. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Eva Zangerle, Martin Pichl, Benedikt Hupfauf, Günther Specht. "Can Microblogs Predict Music Charts? An Analysis of the Relationship between #Nowplaying tweets and Music Charts ", 17th International Society for Music Information Retrieval Conference, 2016.

musical tracks and the Billboard charts, (ii) the temporal relation of tweets and charts to investigate whether there is a timely offset between Twitter and the Billboard Hot 100 charts and (iii) the predictive power of #nowplaying-tweets with respect to the Billboard Hot 100.

The contribution of this work can be summarized as follows: To the best of our knowledge, this is the first paper providing a deep analysis on the relationship between #nowplaying tweets and charts. Furthermore, our study is based on data collected over two years, covering a time frame substantially longer than previous research and particularly, utilizing time series to perform the analyses. We find that the basic correlation of time series representing the Billboard Hot 100 and Twitter performance of all individual tracks is moderate. When performing a cross-correlation analysis to compute a time-agnostic correlation analysis, the correlation coefficient is 0.57. Analyzing the lag between the two time series shows that the temporal offset between musical tweets and charts would allow for a prediction from a temporal perspective for 41% of all tracks. Our prediction experiments show that Twitter data enhances the quality of charts rank predictions as the multivariate model incorporating both Billboard and Twitter data reduces the prediction error significantly and further also reduces the error variance significantly.

The remainder of this paper is structured as follows. Section 2 presents background information and approaches related to our analyses. Section 3 introduces the dataset underlying our analyses, and Section 4 describes the analyses methods we facilitate. Section 5 presents the results of our analysis, and Section 6 discusses our findings in the light of the posed research question. Section 7 concludes the paper.

2. BACKGROUND AND RELATED WORK

Research related to the analyses presented in this paper can be categorized into the following categories: (i) charts and hit prediction based on Twitter data, (ii) analyses of musical tweets and (iii) time series analyses in social media.

Kim et al. [13] present the first approach to predict the success of songs in terms of their Billboard Hot 100 rankings based on Twitter data. Therefore, the authors facilitate a dataset comprising 10 weeks of #nowplaying tweets, and the Billboard Hot 100 of the same time period. The authors use three different features for their further computations: a song's popularity on Twitter, an artist's popularity on Twitter, and the number of weeks a song was in the Billboard Hot 100. Based on these features, the authors compute the Pearson correlation coefficient between the chart-ranking of a particular song in the Billboard Hot 100, and each of the aforementioned features. They find that the song popularity obtains the highest correlation with the ranking of the given songs. In a second step, the authors build different regression models (linear, quadratic linear, and support vector regression models) to predict the ranking of a certain song. The authors find that using all three features with a support vector regression model produces the best prediction performance (r^2=0.75). As for detecting whether a certain song will be a hit, the authors divide their dataset into hits and non-hits and try to predict whether a random non-hit song will be a hit by using random forest classification. The results show that hits (chart rank 1-10) can be predicted with a precision value of 0.92 and a recall value of 0.88. However, these prediction tasks are only performed for songs which are already in the Billboard Hot 100. Incorporating the number of weeks a song was in the charts requires knowledge of the charts and as a consequence, such a method cannot be applied to new songs which are possibly about to enter the charts.

Generally, #nowplaying tweets have been in the focus of researchers aiming to detect musical preference patterns from around the world. However, except for Kim et al., none of these approaches aim to predict charts based on this data. Hauger and Schedl extract genre patterns for regions of the world based on geolocation information of #nowplaying tweets [20]. Schedl et al. also analyse the listening behavior of Twitter users with a particular focus on geospatial aspects [19]. Following up on this research, Schedl and Tkalčič looked into the general genre distribution among tweets, with an emphasis on classical music [21]. Furthermore, Pichl et al. facilitate #nowplaying tweets from the streaming platform Spotify to recommend artists to users [16].

Social media data has been modeled as time series for a variety of analyses, e.g., for predicting the stock market based on emotion within social media [7]. Similarly, time series have been used for modeling mood information extracted from Twitter [6] or the detection of influenza epidemics via Twitter [8]. Further, Sakaki et al. facilitate time series for real-time event detection [18]. Huang et al. model the user of tags on Twitter as a time series to understand how hashtags are used on Twitter [12]. Also, Twitter models data as time series to monitor the health and success of their services as well as to detect anomalies in regards to spikes in user attention [4, 5].

3. DATA

In the following, we describe the data used for the performed analyses. In principle, we require data from two different sources, gathered over the same time frame: #nowplaying tweets, and information about the charts at the given time. As for the set of musical tweets, we choose to base our analyses on a dataset the #nowplaying dataset provided by Zangerle et al. [25] as it is the most extensive dataset of #nowplaying tweets publicly available, which is constantly updated. In total, we utilize all of the 111,260,925 #nowplaying tweets that are available for the years 2014 and 2015. These tweets form the basis of our analysis. To evaluate to what extent Twitter data can be used to predict the charts, we choose to use the Billboard Hot 100 as a reference, as they are one of the most influential indicators for the popularity of songs [2]. Analogously to the work by Schedl and Tkalčič [21], we are aware that the Billboard charts reflect the U.S. market only. Within the Billboard charts, the songs are ranked according to a score computed based on radio airplay, sales and streaming activity [1].

We crawled the top 100 of each week of the years 2014 and 2015 from the Billboard Website [2]. In total, we gathered 886 distinct songs. On average, a track stays within the Hot 100 for 11.74 weeks (SD=10.58), the minimum number of weeks for a track within the Hot 100 is one week and the maximum is 58 weeks.

4. METHODS

This section describes the methods implemented in our study. We propose three analyses to investigate whether the prediction of charts is possible (i) from a correlation perspective, (ii) from a temporal perspective with respect to the distribution and popularity of songs on Twitter and in the charts, and (iii) a prediction perspective analyzing to which extent Twitter data can contribute to predicting future charts for given songs.

To be able to compare #nowplaying tweets and the Billboard Hot 100, we quantify the overlap of musical tweets and charts by counting how many tweets refer to a song on the charts. We aim to match tweets against all tracks that have been on the charts since 2011 to ensure a maximum overlap. We iterate over all tweets and all songs that have been in the charts in nested loops and match title and artist independently. The track title is considered successfully matched, if it is a substring of the tweet (case insensitive). Matching the artist is more complex, as there are several formats, for instance "Michael Jackson" and "Jackson, Michael", in use. Moreover, there are many ways to list featured artists: "Bad Meets Evil Featuring Bruno Mars – Lighters", "Bad Meets Evil – Lighters Featuring Bruno Mars", or sometimes, featured artists are simply neglected. For this reason, the artist string is split at keywords and symbols, such as "feat.", "ft.", "&", etc. If more than half of the resulting tokens can be found in the tweet (again, case insensitive), the artist counts as successfully matched. We consider a track matched, if both track title and artist were matched successfully.

Based on the gathered and preprocessed data on both Twitter and the Billboard Hot 100, we model both of these as time series [10] to compute the analyses as described in the following.

4.1 Correlation of Rankings

In a first step we aim to analyze to which extent Twitter data and Billboard Hot 100 data correlate. We perform a detailed correlation analysis of the popularity of each track in both the Billboard and the #nowplaying dataset. Kim et al. provide a correlation analysis (using Pearson correlation) for (i) the log of the track's playcount and (ii) the log of the artist's playcount and (iii) the number of weeks the song was in the Billboard Hot 100. The authors define the playcount as the median number of mentions of the respective track or artist per day for a given week. We extend this analysis by aggregating the playcounts for a given week not only by using the median of the playcounts of the days of a week, but also computing the mean number of playcounts per track per day and the sum of playcounts for a given week. If a track was not on the Billboard charts in a given week, we consider its rank as 0. Based on this extracted information, we compute the Pearson correlation coefficient for the track and artist playcounts and the Billboard time series [15]. As for the log of the track's playcount and log of artist's playcount, we rely on Spearman's ρ [22] due to the data's ordinal scale and the fact that the logarithm is a non-monotonic transformation of the data [11].

4.2 Temporal Relationship of Tweets and Charts

As we aim to analyze to which extent Twitter data resembles trends in the Billboard Hot 100 and hence, allows for a prediction of future Billboard charts, we are naturally interested in the temporal relationship between Twitter and the Billboard charts. I.e., we analyze whether Twitter does—from a timely perspective—represent trending tracks earlier than those are reflected in the Billboard Hot 100.

To investigate this matter, we propose to perform a cross-correlation analysis of the time series based on Twitter and the time series based on Billboard data for any given music track. In principle, cross-correlation is used to compute the similarity of two signals or time series as a function of the lag between these two signals [14]. This allows for a correlation analysis independent of temporal shifts between the time series while at the same time obtaining information about the lag between the time series. Along the lines of previous research we consider a time lag optimal if it maximizes the correlation of the two signals [23]. This allows to compute the maximum correlation between two time series independent of timely shifts (i.e., Twitter data and Billboard data) and determine the lag between these time series. I.e., if we determine a negative lag for a given track, this implies that Twitter data would allow to predict future charts from a timely perspective.

For the computation of the cross-correlation we rely on the median number of mentions of each track per week on Twitter as this measure has shown to provide the highest correlation value with the Billboard charts (cf. Section 5.1).

4.3 Prediction of Billboard Charts

For the prediction of future Billboard charts, we aim to forecast the performance of a given track in regards to its rank in the future based on past and present observations. Mostly, a regression approach facilitating univariate or multivariate time series is applied for such tasks [9]. Therefore, we propose to evaluate the quality of predictions based on Billboard charts only, Twitter charts only and a combined approach. In particular, we propose to compute three prediction models to evaluate the predictive power of the individual approaches: (i) an autoregressive time series model (AR) based on solely the Billboard time series (we utilize this method as a baseline), (ii) extract the lag from the cross-correlation analysis, shift the time base and compute an AR-model based on the difference between the Twitter and Billboard time series to be able to evaluate the Twitter time series in regards to predicting

Measure	Aggregation Method for Playcounts Median	Mean	Sum
track playcount	0.50 (481)	0.49 (469)	0.49 (453)
log(track playcount)	0.49 (457)	0.48 (453)	0.47 (442)
artist playcount	0.37 (378)	0.37 (364)	0.37 (358)
log(artist playcount)	0.40 (398)	0.38 (389)	0.38 (389)

Table 1: Mean Correlation Coefficients ($p < 0.01$); Numbers in Parenthesis: Number of Significantly Correlating Tracks

the Billboard charts and (iii) utilize information from both Billboard and Twitter to perform a combined prediction approach using a vector autoregressive model for multivariate time series (VAR). Such a VAR model computes predictions based on multiple time series, in our case, Billboard data and Twitter data (particularly, the median number of tweets about each track). As for the training of the models, we rely on the Ordinary Least Squares (OLS) method [9]. We evaluate these three models in regards to their rank prediction accuracy using the standard evaluation measure root mean squared error (RMSE) [9].

5. RESULTS

In the following section we present the results of our analyses in the light of the research questions posed. We firstly present the results of the correlation analysis and subsequently provide the results of the cross-correlation analysis to look into the temporal relationship between Twitter and Billboard. Based on these findings, we present the results of the prediction analysis for Billboard charts.

5.1 Correlation of Tweets and Charts

Firstly, we analyze the correlation of the measures describing the performance of each track extracted from the #nowplaying dataset (cf. Section 4.1). Therefore, we compute the correlation between the mean, median and the sum of playcounts of the track and its rank within the Billboard Hot 100. The results of this analysis can be seen in Table 1, where we list the mean correlation coefficient across all tracks within the dataset. Please note that we only consider those tracks for which we can find a significant correlation ($p < .01$). As can be seen, we observe the highest correlation values for the track playcount with a moderate correlation of 0.50 for 481 of 886 tracks (54.29% of all tracks in the dataset). Figure 1a shows the correlation distribution for all tracks with a significant correlation at the $p < 0.01$-level. We observe both positively correlated and negatively correlated tracks. We also observe that—in line with the findings by Kim et al. [13]—using the median value of all track counts of a given week to represent the performance of any given track on Twitter shows the highest correlation values throughout all configurations. Also, the level of correlation obtained is in line with those observed by Kim et al. as those reach a correlation value of 0.56 for the log of the track playcount [13]. To answer RQ1, we find that there is a moderate correlation between #nowplaying playcount numbers and Billboard charts.

Performing an explorative study on the time series representations of given tracks in the Twitter dataset as well as the Billboard dataset shows a timely lag between these two signals, which we investigate further in the next experiment.

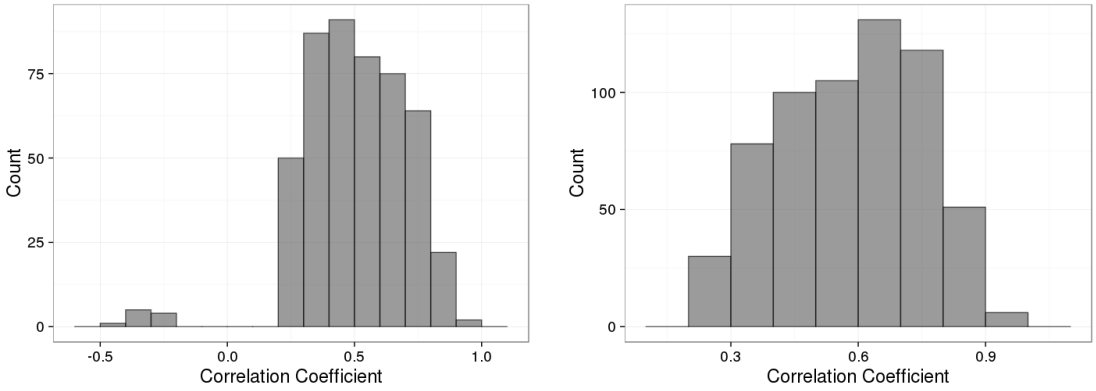
5.2 Temporal Relationship of Tweets and Charts

To gain a deeper understanding for the temporal relationship between tweets and charts, we perform a cross-correlation analysis of the respective time series. As described in Section 4.2, this analysis allows for determining the lag between two time series. Figure 2a depicts the distribution of lags detected for all tracks in the dataset. We observe that the lag histogram shows its highest peaks at -1 and 1 weeks, implying that a substantial number of tracks are mentioned and trending on Twitter either one week before or after these occur and evolve on the Billboard Hot 100 charts. At the same time, the histogram also shows that there also is a substantial number of tracks with a positive time lag. I.e., maximum correlation is reached when shifting the Twitter signal to a later point in time. In total, 286 tracks feature a negative lag (41.09%), whereas 335 tracks (48.13%) feature a positive lag and 75 tracks (10.77%) feature no lag. Table 2 shows a five-number summary of the distribution of time lags (cf. row "all"). On average, the lag between Twitter and the Billboard charts is positive (1.47), whereas the median value is 0.

	Min	Q1	Med	Mean	Q3	Max
All	-17.0	-2.0	0.0	1.47	5.0	17.0
TF	-17.0	-2.0	0.0	0.97	4.0	17.0

Table 2: Five-Number Summary: Temporal Lag (TF refers to those tracks that first occur on Twitter)

We can now utilize the computed lag to shift the base of the time series such that they are maximally correlated. This cross-correlation analysis shows that the mean correlation for all tracks is now 0.57 in contrast to 0.50. The correlation distribution of the playcount measure after the base shift is shown in Figure 1b, where the improvement is clearly visible as the distribution is now shifted towards the right, now only containing positive correlation coefficients in contrast to Figure 1a, where we still observe negative correlation coefficients. The median correlation coefficient is slightly increased from 0.50 to 0.57 when computing the cross-correlation coefficient for all tracks.



(a) Correlation

(b) Cross-Correlation

Figure 1: Histogram of Correlation Coefficients for Track Playcounts

In a second step, we repeat the experiment based on the set of tracks which first occur on Twitter before they actually appear on the Billboard Hot 100. I.e., we extract all tracks for which we observe that they first appear on Twitter and at a later point in time on the charts. This results in a total of 619 tracks. This experiment aims to look into tweets, which would actually allow for a prediction of Billboard charts as these tracks are featured and trending on Twitter *before* they actually appear in the Billboard Hot 100 charts. Figure 2b shows the lags resulting from a cross-correlation analysis of these tracks. As can be seen, the lag distribution is slightly shifted towards the left, i.e., this subset of tweets features lower lags. Table 2 shows the five-number summary of this distribution (cf. row "TF" for Twitter First). Directly comparing the distribution to the lag distribution of all tracks shows that the mean lag is lower, reaching 0.97 weeks. For 69 tracks (11.14%) we do not observe a lag between Twitter and Billboard. 286 tracks (46.20%) feature a positive lag, whereas 264 tracks (42.64%) feature a negative lag. I.e., for 264 tracks a prediction based on Twitter charts seems possible from a temporal perspective.

To answer RQ2, we observe that 89.23% of all tracks actually feature a temporal lag and that 41.09% of all tracks feature a negative lag. I.e., a prediction of charts based on Twitter data is possible from a temporal perspective. When shifting the base of the time series according to the lag, the correlation is increased to 0.57. Looking at the subset of tracks which appear on Twitter first, we observe a negative time lag for 264 tracks, which accounts for 42.64%, which would allow for a prediction.

5.3 Prediction of Charts

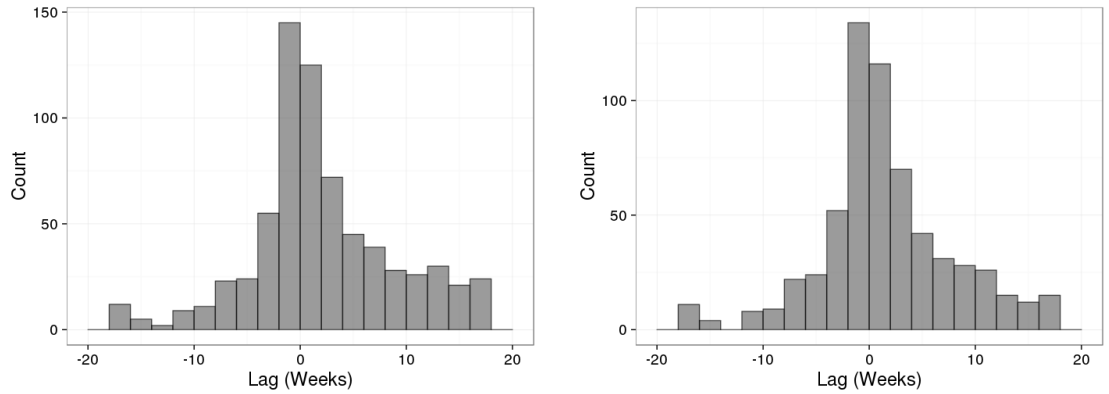
Based on the findings of the previous analyses, we aim to investigate the usefulness of Twitter data for the prediction of Billboard charts in the following. Therefore, we now present the results of a autoregression approach to predicting future Billboard charts. We applied the Augmented Dickey-Fuller test to confirm stationarity in the time series [17].

Table 3 shows the 5-point-summary of the results of the three proposed prediction models in terms of the RMSE values obtained: The Billboard-based autoregressive model (BB), the Twitter-based AR model (T) and the multivariate model combining Twitter and Billboard data (V). Please note that outliers are omitted (we consider all tracks that are more distant than 1.5 interquartile ranges from the upper or lower quartile as outliers). As can be seen, the autoregressive approach based only on Twitter data works substantially worse than the other two approaches, the median of the RMSE distribution being 116.1. In contrast, autoregressive prediction based on the Billboards model (and hence, the baseline) reaches a median RMSE of 26.8 and the VAR model reaches the lowest median RMSE with 12.6. Figure 3 shows a boxplot of the RMSE of the VAR and the Billboard autoregressive model. As can be seen, the VAR model also features a lower variance within the predicted ranks. The mean RMSE—indicated by a diamond in the boxplot—is also clearly lower for the VAR model (14.1 vs. 26.8, hence, a 48.38% lower value). Due to the non-normal distribution of the data, we apply a Mann-Whitney U test to test for significant differences in the prediction performance of the VAR and Billboard AR model and the result shows a significantly lower RMSE for the VAR-model ($p < 0.05$). A Levene's test of equality of RMSE variance shows that the VAR model reaches significantly lower error variance in terms of ranking predictions than the Billboard AR model ($p < 0.01$).

Regarding RQ3, we can therefore observe that combining Twitter and Billboard data enhances the quality of predictions. We show that the RMSE is significantly lower when incorporating Twitter information. Similarly, we show that the error variance of predictions is significantly decreased using a multivariate model.

6. DISCUSSION

In the following section, we discuss the results presented in the previous section, and put them into context in order to confirm or disprove our hypothesis that Twitter musical data can be used for predicting music charts.



(a) All Tracks

(b) Tracks First Appearing on Twitter

Figure 2: Distribution of Temporal Lags between Twitter and Billboard Hot 100

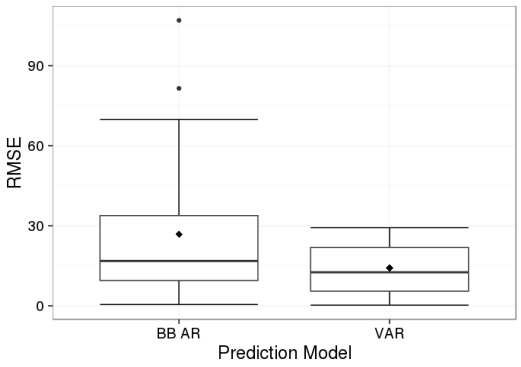


Figure 3: Boxplot of RMSE of Prediction Models (Diamond: Mean Value)

As for the correlation between the time series of the Billboard Hot 100 ranks of tracks and the time series of the tweet playcounts, we observe moderate correlation. In line with the findings by Kim et al. [13], who observe a correlation coefficient of 0.56 between the Billboard Hot 100 and the logarithm of the track playcounts, our dataset features a maximum correlation between Billboard and Twitter of 0.50. Our dataset represents a long-term view on the relationship between Twitter and the Billboard charts as we performed the analyses over the course of two years, whereas Kim et al. observed a window of 10 weeks.

Based on the performed evaluation, we find that from a temporal perspective, approximately 41% of all #nowplaying tweets can be used to predict music charts as these appear on Twitter before they actually appear on the charts.

	Min	Q1	Med	Mean	Q3	Max
T	16.3	84.5	116.1	148.6	178.2	388.4
BB	0.51	9.5	16.8	26.8	33.8	107.0
V	0.27	5.5	12.6	14.1	21.9	29.3

Table 3: Five-Number Summary: Charts Prediction RMSE (T: Twitter AR, BB: Billboard AR, V: VAR model)

However, the lag between the two time series is rather low and more importantly, the mean lag across all tracks is positive. Therefore, we argue that approaches exploiting this temporal shift for predictions does not seem promising. However, using Twitter data to enhance chart predictions based on Billboard data has shown to provide promising results. Our evaluation of autoregressive prediction approaches shows that incorporating Twitter data in the prediction process using a multivariate model significantly lowers the RMSE as well as the variance of the RMSE. I.e., we are able to predict the rank of tracks more accurately while at the same time providing a lower error margin for predictions. Hence, we come to the conclusion that #nowplaying data is able to contribute to better charts prediction and can be utilized as an additional sensor which allows for enhancing chart predictions. However, we have to note that solely relying on Twitter data for charts prediction has shown to be highly error-prone and performing substantially worse than both the Billboard AR and the VAR model presented.

Another limiting factor we are aware of is the demographic gap between the user base of Twitter and the average consumer in the U.S. music market (represented by the Billboard Hot 100 charts).

7. CONCLUSION

Based on a dataset gathered from Twitter and the Billboard charts over the course of 2014 and 2015, we analyzed the relationship between Twitter and the Billboard charts in regards to whether tweets could be utilized for predicting future Billboard charts. Therefore, we performed a three-fold analysis of the dataset. These experiments showed that in principle, Twitter and Billboard time series for tracks share a moderate correlation which is influenced by a timely shift between those two. We further find that there is a negative timely lag for 41% of all tracks. As for the predictive power of #nowplaying charts we find that a multivariate model incorporating both Billboard and Twitter data significantly reduces the prediction error while at the same time, the error variance is significantly reduced.

8. REFERENCES

[1] Billboard Charts Legend: http://www.billboard.com/biz/billboard-charts\-legend, 2016. (last visited: 2016-02-25).

[2] Billboard Hot 100: http://www.billboard.com/charts/hot-100, 2016. (last visited: 2016-02-25).

[3] Twitter: About Twitter https://about.twitter.com/company, 2016. (last visited: 2016-02-28).

[4] Twitter Blog: Introducing practical and robust anomaly detection in a time series https://blog.twitter.com/2015/introducing\-practical-and-robust-anomaly-detect\-ion-in-a-time-series, 2016. (last visited: 2016-03-01).

[5] Twitter Blog: Observability at Twitter https://blog.twitter.com/2013/observability-at-twitter, 2016. (last visited: 2016-03-01).

[6] Johan Bollen, Huina Mao, and Alberto Pepe. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, volume 11, pages 450–453. The AAAI Press, 2011.

[7] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8, 2011.

[8] David A Broniatowski, Michael J Paul, and Mark Dredze. National and local influenza surveillance through twitter: an analysis of the 2012-2013 influenza epidemic. *PloS one*, 8(12):e83672, 2013.

[9] Chris Chatfield. *Time-series forecasting*. CRC Press, 2000.

[10] James Douglas Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, 1994.

[11] Jan Hauke and Kossowski Tomasz. Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data. *Quaestiones Geographicae*, 30(2):87–93, 2011.

[12] Jeff Huang, Katherine M. Thornton, and Efthimis N. Efthimiadis. Conversational tagging in twitter. In *Proc. of the 21st ACM Conference on Hypertext and Hypermedia*, HT '10, pages 173–178, New York, NY, USA, 2010. ACM.

[13] Yekyung Kim, Bongwon Suh, and Kyogu Lee. #Nowplaying the Future Billboard: Mining Music Listening Behaviors of Twitter Users for Hit Song Prediction. In *Proc. of the First International Workshop on Social Media Retrieval and Analysis*, SoMeRA '14, pages 51–56, New York, NY, USA, 2014. ACM.

[14] Sophocles J. Orfanidis. *Introduction to Signal Processing*. Prentice-Hall, Inc., 1995.

[15] William R Pearson and David J Lipman. Improved tools for biological sequence comparison. *Proc. of the National Academy of Sciences*, 85(8):2444–2448, 1988.

[16] Martin Pichl, Eva Zangerle, and Günther Specht. Combining Spotify and Twitter Data for Generating a Recent and Public Dataset for Music Recommendation. In *Proc. of the 26nd Workshop Grundlagen von Datenbanken, Ritten, Italy*, 2014.

[17] Said E Said and David A Dickey. Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, 71(3):599–607, 1984.

[18] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proc. of the 19th International Conference on World Wide Web*, WWW '10, pages 851–860, New York, NY, USA, 2010. ACM.

[19] Markus Schedl. Leveraging Microblogs for Spatiotemporal Music Information Retrieval. In *Advances in Information Retrieval - 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, Proceedings*, pages 796–799. Springer, 2013.

[20] Markus Schedl and David Hauger. Mining Microblogs to Infer Music Artist Similarity and Cultural Listening Patterns. In *Proc. of the 21st WWW: 4th International Workshop on Advances in Music Information Research*, Lyon, France, 2012.

[21] Markus Schedl and Marko Tkalčič. Genre-based Analysis of Social Media Data on Music Listening Behavior. In *Proc. of the 1st ACM International Workshop on Internet-Scale Multimedia Management*, ISMM '14, pages 9–14, New York, NY, USA, 2014. ACM.

[22] Charles Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 15(1):72–101, 1904.

[23] Mikalai Tsytsarau, Themis Palpanas, and Malu Castellanos. Dynamics of news events and social media reaction. In *Proc. of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 901–910. ACM, 2014.

[24] Eva Zangerle, Wolfgang Gassler, and Günther Specht. Exploiting Twitter's Collective Knowledge for Music Recommendations. In *Proceedings, 2nd Workshop on Making Sense of Microposts: Big Things come in Small Packages, Lyon, France*, pages 14–17, 2012.

[25] Eva Zangerle, Martin Pichl, Wolfgang Gassler, and Günther Specht. #nowplaying Music Dataset: Extracting Listening Behavior from Twitter. In *Proc. of the 1st ACM International Workshop on Internet-Scale Multimedia Management*, ISMM '14, pages 3–8, New York, NY, USA, 2014. ACM.

CROSS TASK STUDY ON MIREX RECENT RESULTS: AN INDEX FOR EVOLUTION MEASUREMENT AND SOME STAGNATION HYPOTHESES

Ricardo Scholz **Geber Ramalho** **Giordano Cabral**
Universidade Federal de Pernambuco
Recife, PE, Brasil
reps@cin.ufpe.br glr@cin.ufpe.br grec@cin.ufpe.br

ABSTRACT

In the last 20 years, Music Information Retrieval (MIR) has been an expanding research field, and the MIREX competition has become the main evaluation venue in MIR field. Analyzing recent results for various tasks of MIREX (MIR Evaluation eXchange), we observed that the evolution of task solutions follows two different patterns: for some tasks, the results apparently hit stagnation, whereas for others, they seem getting better over time. In this paper, (a) we compile the MIREX results of the last 6 years, (b) we propose a configurable quantitative index for evolution trend measurement of MIREX tasks, and (c) we discuss possible explanations or hypotheses for the stagnation phenomena hitting some of them. This paper hopes to incite a debate in the MIR research community about the progress in the field and how to adequately measure evolution trends.

1. INTRODUCTION

In the last 20 years, mainly due to growth of audio data available in the Internet, Music Information Retrieval (MIR) has been an expanding field of research. It encompasses various problems or tasks, whose solutions have impact in music market. Since 2005, the MIR Evaluation eXchange (MIREX) [7] is the main evaluation "arena" in MIR field, proposing datasets, tasks and metrics to compare MIR solutions. A shallow analysis of its results shows they are continuously evolving for some tasks, whereas they seem stagnated for other ones.

There are several MIR and MIREX meta-analysis papers [6][7][23][24]. However, to our knowledge, a transversal study over stagnation of results on MIR tasks is lacking, as well as an index for evolution trend measurement. Also, stagnation phenomenon on many of these tasks is not yet being deeply discussed by the community.

Both the existence of common reasons and task specific reasons for stagnation on MIR tasks are very probable. Therefore, a deep study of stagnation phenomena is task-dependent, and demands the analysis of techniques, datasets and metrics used in recent years. Then, it is out of the scope of this paper to perform a deep analysis on the reasons of stagnation for each one of the MIREX tasks. This paper intends, instead, to provoke researchers involved with MIREX tasks (stagnated or not) to test some general hypotheses we suggest, and to propose their own task specific hypotheses.

Understanding of stagnation phenomena may be improved by objective evolution trends measurement. Comparing evolution trends between different datasets or metrics, for a given task, possibly help to identify how metrics and datasets bias observable results, or how each sub-problem of the task is more or less developed. In addition, evolution trends comparison between different tasks provide an overall picture of evolution in MIR research, drawing attention to what kind of methods and strategies are being used on developing tasks that could be adapted for stagnated ones.

This paper presents an accurate empirical analysis of MIREX recent results. It also proposes a configurable quantitative index for evolution trends measurement. Finally, it raises some hypotheses and questions that could possibly explain stagnation phenomena and/or hopefully help MIR research community to exchange more information about it in order to move forward.

Section 2 presents method used to analyze data. We explain and formalize a configurable index for evolution measurement on Section 3. Section 4 raises hypotheses about possible causes of stagnation. Section 5 draws some general conclusions on the performed analysis. Finally, future works are listed on Section 6.

2. METHOD

MIREX is the MIR competition that became the main evaluation venue in MIR field. It has been running since 2005. According to MIREX 2015 final results' poster, 107 researchers from 64 teams participated in the last edition and submitted algorithms for 21 active tasks, resulting in 402 runs, over 47 different datasets [11].

MIREX contributions to the MIR community are evident. Influential MIR researchers have identified four key contributions of MIREX: "training and induction into MIR", "dissemination of new research", "dissemination of data" and "benchmarking and evaluation" [7].

In order to evaluate research progress in MIR tasks, we could have tried to compare results published in


© Ricardo Scholz, Geber Ramalho, Giordano Cabral.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Ricardo Scholz, Geber Ramalho, Giordano Cabral. "Cross-task Study on MIREX Recent Results: an Index for Evolution Measurement and Some Stagnation Hypotheses", 17th International Society for Music Information Retrieval Conference, 2016.

recent years. However, we decided to focus analysis in MIREX because it can be more systematic, since: (1) MIREX tasks are well defined, and (2) submissions from different years to a given task/subtask run over the same datasets and (3) the results are evaluated using the same metrics. We do acknowledge the limitations of this methodological choice, since not all MIR algorithms developed have been evaluated in MIREX competition. But, for the sake of comparison precision and extensiveness, it seemed to be the best choice.

A timeline of tasks, subtasks and datasets used for each task or subtask was constructed with data collected from MIREX results between 2010 and 2015 [11]. In order to analyze tendencies, it is necessary to consider a relevant time frame, as well as to guarantee comparisons over time are consistent. As inclusion criterion, only tasks for which there was at least one dataset used for at least five editions since 2010 were admitted. The rationale of this choice is that observing a unique dataset ensures consistency and comparability of results, whereas considering at least five editions provides a reliable time window for trend analysis. Though, from all 28 tasks proposed between the first edition in 2005 and the last in 2015, 4 tasks were discontinued until 2008, other 6 tasks were considered very recent (started in 2013 or later), whereas the remaining 18 tasks were analyzed in this study, including 3 active tasks in 2014 which did not run in 2015.

We assumed datasets and methods for metrics computation did not change, except when explicitly documented on the task's MIREX official wiki or results' pages [11]. Among the remaining candidates, one dataset for each task or subtask was chosen to collect data for analysis. When more than one dataset was available, older datasets were preferred, to allow future researches to extend this work by comparing backwards. For Audio Genre Classification task, two datasets were equally older: Mixed Set and Latin. Mixed Set was then chosen, as a more generic set tends to provide a more realistic picture of the state of the art.

Among 18 analyzed tasks, 3 tasks presented more than one subtask. "MF0 Estimation & Tracking" is divided into "MF0 Estimation" and "Note Tracking". Actually, we believe that they could be two different tasks themselves, due to the different nature of their objectives. Then, both subtasks were analyzed. For "Query-by-tapping" (QBT), two subtasks are available: "QBT with symbolic input" (subtask 1) and "QBT with wave input" (subtask 2). We analyzed subtask 1, since onset files allow participants to concentrate on similarity matching, which is the main objective of the task, instead of onset detection. Finally, "Query-by-Singing/ Humming" (QBSH) presented two subtasks: "Classic QBSH evaluation" and "Variants QBSH evaluation". Classic evaluation (subtask 1) was chosen, since the variants evaluation adds constraints to the original problem – for instance, considering queries as variants of "ground-truth" midi.

Each task has several metrics computed. As our analysis needs to rank results, for the sake of comparison, one metric for each task or subtask was chosen. As this analysis aims to understand evolution of the state of the art on each task, more general metrics were assumed to provide a more realistic picture of each task's performance. Then, metrics often used in MIREX Overall Results Posters [11] and metrics measuring overall performance were chosen, at the expense of those measuring a given characteristic of the algorithms. For instance, F-Measure was preferred when tasks also compute Precision and Recall, as Precision and Recall compute specific performances whereas F-Measure relates to both Precision and Recall.

Considering the chosen metrics, top results for each task were analyzed. We then noticed that two groups emerged: "tasks presenting stagnated results" and "tasks presenting evolving results". The first group included tasks which presented no significant improvement on results in the last years of the competition. And the second group included tasks whose results' evolution is noticeable in the last six years. Of course, there is a high level of subjectivity on deciding when a given task is evolving (and at which pace), or stagnated. For a systematic analysis, a quantitative index for evolution measurement is necessary.

3. AN INDEX FOR EVOLUTION MEASUREMENT

To perform our study, we needed a quantitative index for measuring results' evolution trends, in order to distinguish stagnated results from evolving ones. In this section we introduce what we called "Weighted Evolution Measurement Index" (WEMI), and we discuss its semantics.

Measuring stagnation phenomena by just looking to evolution graphs has limitations, as similar graphs may

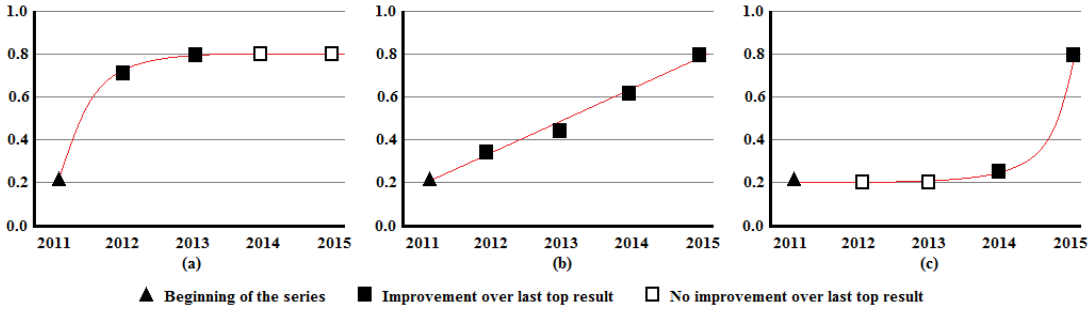


Figure 1. Examples of different evolution trends: (a) stagnation trend; (b) continuous evolution trend; and (c) recovery from recent stagnation trend.

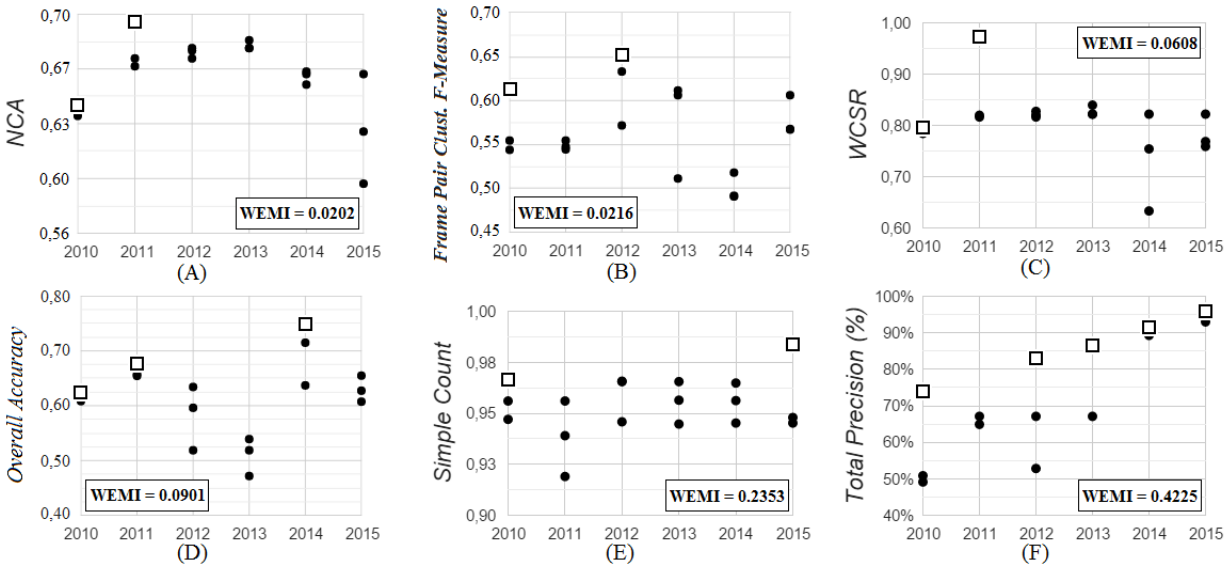


Figure 2. Some tasks results evolution plots (3 top results per year) and respective WEMI values (*w=0.6* and *c=0.0713*): (A) "Audio Music Mood Classification"; (B) "Music Structure Segmentation"; (C) "Audio Chord Estimation"; (D) "Audio Melody Extraction"; (E) Query-by-Singing/Humming"; and (F) "Score Following"; top historical results (squares) were used for trend analysis.

be hard to distinguish. For a state of the art analysis of trends, we must be able to objectively differentiate *continuous* from *intermittent* evolution, as well as measure *how* evolution occurred over time and consistently compare evolution of different tasks. For instance, consider Figure 1, which shows different hypothetical evolution scenarios. In all cases, results evolve from 0.2 to 0.8, so the first and the last result of all series coincide (overall error drop was exactly the same). However, the way evolution occurred is different in each case, so evolution trends are not the same. First series (a) shows a clear *stagnation trend*, as no recent improvement occurred after a huge improvement in the past. Second series (b) shows a *continuous evolution* of results, with small improvements every year. Finally, third series (c) shows a huge *recovery from a recent stagnation period*, as recent improvements occurred after many years without any improvement. Therefore, it is interesting that an index for evolution trend measurement can be properly balanced to differentiate these scenarios. In addition, such an index must be consistent in scenarios of complete stagnation (i.e., no evolution since the beginning of the series).

Considering that we are interested in state of the art evolution, it does make sense to discard results which did not overcome the top result achieved so far. Then, the proposed index considers evolution as a monotonically increasing function. Figure 2 shows various examples of actual top results per year, and results selected for trend analysis.

The index we propose considers a series of results from year *i* to year *f* (in this study, *i = 2010* and *f = 2015*). According to the chosen metric and state of development of each task, bias may occur if we observe top results directly. In order to avoid it, we consider relative error drop rate from one year to the next.

Error, in a given year *y*, such that $i \leq y \leq f$, is defined as:

$$e_y = 1 - max(r_i, r_{i+1}, \ldots, r_y) \quad (1)$$

Where r_j is the top result achieved in year *j*. The error drop rates are then computed for each pair of successive years (*y-1* and *y*, such that $i+1 \leq y \leq f$), as:

$$\Delta e_y = 1 - \left({}^{e_y}/_{e_{y-1}}\right) \quad (2)$$

Recent evolution is reinforced by higher weights of error drop rates for recent years, so that recent improvements tend to push WEMI up more than results achieved many years ago, even if the error drop rate in both cases was the same.

Continuous evolution is reinforced with a direct proportionality between WEMI and the number of actual improvements within the time frame. This way, WEMI tends to be higher when continuous evolutions are achieved each year, in comparison with the situation in which the same overall evolution is achieved from one year to the next, at once. Then, WEMI is defined as:

$$WEMI = \left(\frac{\sum_{y=i+1}^{f} w^{f-y+1} \times \Delta e_y}{\sum_{y=i+1}^{f} w^{f-y+1}}\right) + c\frac{q}{f-i} \quad (3)$$

The number of improvements over previous top result between *i+1* and *f* (i.e., the number of times Δe_j is larger

Task	Dataset Used	Metric Observed	YHTR	*q*	OEDR	WEMI
Audio Music Sim. and Retr.	Default	Av. Fine Score Human Eval.	2011	1	0.04	0.0188
Audio Music Mood Classif.	MIREX 2007	Normalized Class. Accuracy	2011	1	0.14	0.0202
Music Structure Segment.	MIREX 2009	Frame Pair Clust. F-Measure	2012	1	0.10	0.0216
Audio Tag Classification	Maj/Min Tag	Tag Classification Accuracy	2011	1	0.11	0.0254
MFFE&T – MF0 Estimat.	MIREX 2009	Chroma Precision	2011	1	0.36	0.0324
Audio Class. Comp. Ident.	MIREX 2009	Normalized Class. Accuracy	2011	1	0.39	0.0341
Symbolic Melodic Simil.	Essen Col.	"Fine" score[1]	2013	2	0.12	0.0398
Audio Key Detection	MIREX 2005	Weighted Key Score	2013	1	0.26	0.0540
Audio Chord Estimation	MIREX 2009[2]	Weigh. Chord Symbol Recall	2011	1	0.87	0.0608
Classic Query-by-Tapping	Roger Jang	Simple Count	2012	1	0.29	0.0738
Audio Onset Detection	MIREX 2005	Average F-Measure	2013	3	0.40	0.0801
Audio Genre Classification	Mixed Popular[3]	Normalized Class. Accuracy	2014	2	0.37	0.0829
Audio Melody Extraction	MIREX 2005	Overall Accuracy	2014	2	0.33	0.0901
Audio Tempo Estimation	MIREX 2006	Average P-Score	2015	2	0.18	0.1014
Audio Beat Tracking	MCK	F-Measure	2015	4	0.20	0.1038
MFFE&T – Note Tracking	MIREX 2009	Average F-Measure[4]	2014	3	0.58	0.1731
Query-by-Singing/Humm.	Roger Jang	Simple Count	2015	1	0.51	0.2353
Score Following[5]	Not identified[6]	Total Precision	2015	4	0.83	0.4225
Audio Cover Song Identif.	Mixed Collec.	Total num. of cov. id. in top 10[7]	2013	1	N/A	N/A

[1] Sum of fine-grained human similarity decisions. | [2] Major/minor triads classification. | [3] Also known as US Pop Music. | [4] For onset only over *chroma*. | [5] Also known as "Real Time Audio to Score Alignment". | [6] MIREX result pages mentions 3 datasets, but we could not identify which one was considered for the results in provided tables. | [7] Metric "mean number of covers identified in top 10 (average performance)" would be preferable, but is not available for all years.

Table 1. Analyzed tasks' general information; WEMI computed for w = 0.6 and c = 0.0713; YHTR stands for "Year of Historical Top Result"; OEDR stands for "Overall Error Drop Rate", computed as $OEDR = 1 - (e_{2015}/e_{2010})$.

than zero, for $i+1 \leq j \leq f$) is called q (if no improvement occurred, WEMI must be zero). Two configurable constants, w and c, are defined such that $0 < w \leq 1$ and $c > 0$. Clearly, the closer w is to zero, the greatest the weight of *recent improvements* on final index, whereas the closer it is to *1*, more equalized weights are used. Also, the closer c is to zero, the lowest the weight of *constant evolution* on final WEMI value. In this study, we computed WEMI for a variety of w and c values (results are available at *https://goo.gl/bxwrDy*). Balancing w and c depends mainly on the importance one gives to *recent* against *continuous* evolution. Therefore, we believe a discussion on MIR community about this trade off would lead to more appropriate balancing of the index for MIREX tasks, considering the goal of identifying bottlenecks of evolution and/or evaluation.

The "Audio Cover Song Identification" task could not have WEMI computed, as expected "total number of covers identified in top 10" (T10) is not available. However, results almost doubled T10 from 908 in 2010 to 1714 in 2013, regardless of the absence of improvements in other MIREX editions.

A total of 18 tasks were analyzed, with "MF0 Estimation & Tracking" comprising two subtasks. This resulted in 19 task/subtasks. Table 1 shows a summary of the analysis, with examples of output for $w = 0.6$ and $c = 0.0713$ (the average weighted sum of error drop rates of all tasks, considering $w = 0.6$).

WEMI is a first proposal and a provocation for a broader discussion about evolution measurement indexes for MIR tasks, especially on MIREX competition. Objective and early identification of stagnation trends may raise earlier discussions in the community about the appropriateness of metrics, datasets or current methods for a given task, probably helping to shorten future stagnation periods or to improve current metrics and/or datasets. Evidently, computing WEMI for a single metric of a task may be misleading. On the other hand, computing it for many metrics of a task will probably lead to a greater understanding of specific bottlenecks in task evaluation and/or evolution.

4. SOME STAGNATION HYPOTHESES

Stagnation on most MIR task results is already acknowledged by MIR community, as in the case of singer identification in polyphonic audio [13], music transcription [2], emotion and genre classification [14][19], music similarity [8][18], and so on. In spite of this acknowledgement, there is not much discussion about possible hypothesis which could explain the phenomena. Sturm [22], among others [23][24], have

recently raised questions about the experimental validity in MIR evaluation, stating reliable evaluation remains neglected in MIR research. Even though, data put together in this research inspire some questions. This paper intends to provoke this kind of questions, and its explanation hypotheses.

In order to encourage this discussion, identifying whether MIREX manages to satisfactorily measure improvements of performance for its various tasks is necessary. If this is true, why so many of them are stagnated? Temporary solution stagnation phenomena are a normal stage in scientific development. However, some mechanisms could be employed to shorten them.

As we said before, since the explanations for stagnation may be task-dependent, it is difficult to provide general explanations for stagnation, and then hints on how to overcome it. Nevertheless, from some discussion on the literature of specific tasks, coupled with our own experience in MIR, we have formulated two hypotheses that may possibly help researchers to move forward. These hypotheses are not meant to be correct, but rather to start a transversal discussion among stagnated tasks.

The first hypothesis is: MIR approaches should perhaps be more musical knowledge-intensive. According to Downie [6], in 2008, community members were becoming aware of the limitations of MIR "generic approaches", i.e. the application of information retrieval solutions for music, without relying on musically meaningful features. However, since then, most of works in MIREX seems to still rely on more generic IR techniques than on an in-depth use of specific music knowledge. It is true that embedding music expert knowledge pawn generality of the approaches, but perhaps this could be path to move away from stagnation.

Let's take "Audio Chord Estimation" as an example. Chord estimation is apparently stuck into a kind of glass ceiling. Very often, approaches are agnostic, neglecting contextual information or musical knowledge after feature selection. The most successful approaches in MIREX often use probabilistic machine learning techniques, mostly through neural networks, as HMM, MLN or Bayesian [3][17][20][25]. A few approaches make use of specialist knowledge, applying it on the lower levels of symbolic information, in order to improve feature vector quality [4][12]. A deeper study on "Audio Chord Estimation" state of the art, performed by McVicar et al. [16], observed advances in feature extraction and modeling stage, as well as expert musical knowledge use for model training, but no musical knowledge use for post-processing, for instance.

Musical creation process is essentially artistic. Then, perplexity of harmonic sequences in real world tends to be high, implying less predictability. In fact, one cannot talk about a correct or wrong chord sequence, as in most classification problems. A composer not only is free to create novel chord sequences, but he or she tends to look for them. Therefore, purely probabilistic approaches are limited by the predictability of analyzed corpus, meaning that uncommon (artistically novel) chord sequences may be misrecognized. In addition, other variables may interfere in harmonic sequences, such as genre (jazz harmony differs strongly from rock harmony) or style (a given musician tends to prefer some chord sequences).

There are evidences that musical knowledge can improve chord estimation [21][1][5][15]. Therefore, we believe that improvements can be achieved using musical knowledge on higher levels of information and contextual information to decide what chord is represented by a given feature vector. By higher levels of information we basically mean musical theory applied over symbolic information. For instance, the use of functional harmonic analysis, which has been proved to add relevant information to chord sequences [21][5], to chose, among candidate chords, the ones which lead to more meaningful chord sequences, even when their feature vector are not the first options provided by a feature vector based classifier.

Music structure information has also been shown to add relevant contextual information for chord estimation [15]. For instance, the classification of "easy" chords first and the use of this information to help classification of "harder" ones, according to the harmonic meaning of the sequence they would lead to, or using harmonically similar pieces already classified of the same song (sometimes with better conditions to feature extraction and classification, such as less noise, transients, arpeggios or ornamentation) may lead to improvement on current results.

The second hypothesis is: the number of techniques employed by the MIREX community is perhaps too limited. It is very difficult to prove that a particular technique is not used by the community, as failed attempts are rarely published. But observing recent ISMIR publications, we noticed that each task presents a small set of often used techniques. For instance, chord estimation tends to rely mostly on HMM, but also on MLN or Bayesian networks, for classification.

To reinforce our hypothesis, we analyze the impact of a specific technique in MIREX results, showing how the use of a new technique can affect results. Chosen technique was Deep Learning, which dates back to the Neo-cognition introduced by K. Fukushima in 1980 [9], but only a few years ago have been found promising for MIR.

In 2012, Humphrey warned about the lack of deep learning approaches in MIR research [10]. Analyzing the top results from 2010 to 2015, among the 19 tasks/subtasks considered in this work, we can notice that, in the last 3 years, 11 tasks had their results improved, but only 3 of the improvements came from approaches using deep learning techniques, according to

the technical reports submitted to MIREX. This may suggest that (1) deep learning is not being extensively explored yet in MIR and (2) if deep learning could improve results in three of the tasks, it is fair to consider there are possibly other techniques, yet not explored, with similar potential.

Regarding the first assertion, it might be due to the lack of enough labeled data, meaning that some tasks are not even eligible for Deep Learning yet. In this case, it would be fair to consider the creation of new datasets or enlargement of existing ones as a possible path to overcome stagnation on these tasks.

Of course, other hypotheses could be deeper investigated. For instance, the lowest WEMI values (even for several different *w* and *c* values) belong to tasks which use human generated ground truth data. Further investigation of this relation could lead to relevant information. Unsuitability or limitation of the datasets and metrics of the stagnated tasks are worth to investigate. However many of these hypotheses are task dependent and such an investigation would be better performed by specialists on each task.

5. CONCLUSIONS

Trying to understand recent practical advances in MIR research, a compilation of the last 6 years of MIREX results for 18 tasks (one of them comprised of two sub-tasks) was performed. Aiming to encourage discussion on how to measure progress in MIR, we propose a configurable quantitative index of improvement, the "Weighted Evolution Measurement Index" (WEMI), in order to objectively measure trends on each task, in a comparable way, reinforcing *recent* and *continuous* advances. We believe such an index may help understanding bottlenecks of evolution or measurement issues, by comparing different datasets and metrics for a given task (intra-task analysis), as well as helping to overcome stagnation, by task comparisons, observing whether methods and strategies of evolving tasks are being applied to stagnated ones (inter-task analysis). The index can be balanced, according to the community's understanding of what is most relevant: *continuous* or *recent* evolution. Also, we raise hypotheses and questions about stagnation affecting many of MIR tasks and we point some possible insights on this matter. We believe that a deeper discussion in MIR community about stagnation phenomena affecting many of MIR tasks may help to find general mechanisms or strategies which will allow overcoming it, as well as improving MIREX interest and relevance.

6. FUTURE WORK

It would be possible to obtain a more detailed overview of MIREX tasks' trends with a number of additional information, for instance: (1) comparing WEMI computed for other metrics or other datasets, in a given task, will probably help understanding if metrics or datasets are biasing observable evolution trends at first sight; and/or (2) a deeper study on discrepant results (for instance, "Classical Composer Identification, 2011" and "Chord Estimation, 2011", as seen in Figure 2) in order to identify overfitting or other distortions of top result will certainly improve accuracy. Another interesting improvement would be adaptation of WEMI so that total weights sum is normalized by the time window, as this would allow more consistent comparisons between time windows of different lengths, if this makes sense in a given context. Finally, a formal evaluation of the index is still missing.

7. REFERENCES

[1] Bello, J. P.; Pickens, J. 2005. A Robust Mid-level Representation for Harmonic Content in Music Signals. *Proc. of the 6th International Society for Music Information Retrieval Conference* (London, United Kingdom, September 11 – 15), ISMIR'05. 304-311.

[2] Benetos, E., et al. 2013. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, 41, 3 (July 2013), 407-434. DOI=10.1007/s10844-013-0258-3.

[3] Chen, R.; Shen, W.; Srinivasamurthy, A.; Chordia, P. 2012. Chord recognition using duration-explicit hidden Markov models. *Proc. of the 13th International Society for Music Information Retrieval Conference* (Porto, Portugal, October 8 – 12, 2012), ISMIR'12. 445-450.

[4] Cho, T.; Bello, J. P. 2011. A feature smoothing method for chord recognition using recurrence plots. *Proc. of the 12th International Society for Music Information Retrieval Conference* (Miami, USA, October 24 – 28, 2011), ISMIR'11. 651-656.

[5] De Haas, W. B.; Rodrigues Magalhães, J. P. and Wiering, F. 2012. Improving Audio Chord Transcription by Exploiting Harmonic and Metric Knowledge. *Proc. of the 13th International Society for Music Information Retrieval Conference* (Porto, Portugal, October 8 – 12, 2012), ISMIR'12. 295-300.

[6] Downie, J. 2008. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29, 4, 247-255. DOI=10.1250/ast.29.247.

[7] Downie, J., et al. 2014. Ten years of MIREX: reflections, challenges and opportunities. In *Proc. of the 15th International Society for Music Information*

Retrieval Conference (Taipei, Taiwan, October 27 – 31, 2014). ISMIR'14. 657-662.

[8] Flexer, A. 2014. On Inter-rater Agreement in Audio Music Similarity. In *Proc. of the 15th International Society for Music Information Retrieval Conference* (Taipei, Taiwan, October 27 – 31, 2014), ISMIR'14. 245-250.

[9] Fukushima, K. 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36, 4, 193–202. DOI = 10.1007/bf00344251.

[10] Humphrey, E. J.; Bello, J. P.; Lecun, Y. 2012. Moving Beyond Feature Design: Deep Architectures and Automatic Feature Learning in Music Informatics. *Proc. of the 13th International Society for Music Information Retrieval Conference* (Porto, Portugal, October 8 – 12, 2012), ISMIR'12. 403-408.

[11] ISMIRSEL. 2016, January 8. *MIREX Home [Online]*. Available: http://www.music-ir.org/mirex/wiki/MIREX_ HOME.

[12] Khadkevich, M.; Omologo, M. 2011. Time-frequency reassigned features for automatic chord recognition. *Proc. of the 36th IEEE International Conference on Acoustics, Speech and Signal Processing* (Prague, Czech Republic, May 22 – 27, 2011), ICASSP'11. 181-184.

[13] Lagrange, M., Ozerov, A., and Vincent, E.. 2012. Robust singer identification in polyphonic music using melody enhancement and uncertainty-based learning. *Proc. of the 13th International Society for Music Information Retrieval Conference* (Porto, Portugal, October 8 – 12, 2012), ISMIR'12. 595-600.

[14] Lidy, T., et al. Improving Genre Classification by Combination of Audio and Symbolic Descriptors Using a Transcription System. *Proc. of the 8th International Society for Music Information Retrieval Conference* (Viena, Austria, September 23 – 27, 2007), ISMIR'07. 61-66.

[15] Mauch, M.; Noland, K. and Dixon, S. 2009. Using Musical Structure to Enhance Automatic Chord Transcription. *Proc. of the 10th International Society for Music Information Retrieval Conference* (Kobe, Japan, October 26 – 30, 2009), ISMIR'10. 231-236.

[16] McVicar, M.; Santos-Rodríguez, R.; Ni, Y. and De Bie, T. 2014. Automatic Chord Estimation from Audio: A Review of the State of the Art. *Proc. of IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22, 2, 556-575. DOI = 10.1109/TASLP.2013.229458.

[17] Ni, Y.; McVicar, M.; Santos-Rodríguez, R.; De Bie, T. 2012. An end-to-end machine learning system for harmonic analysis of music. *Proc. of the IEEE Transactions on Audio, Speech and Language Processing*, 20, 6, 1771-1783. DOI = 10.1109/TASL.2012.2188516.

[18] Pachet, F. and Aucouturier, J. 2004. Improving timbre similarity: How high is the sky?. *Journal of Negative Results in Speech and Audio Sciences*, 1, 1, 1-13.

[19] Panda, R., et al. 2013. Multi-modal music emotion recognition: A new dataset, methodology and comparative analysis. *Proc. of the 10th International Symp. on Computer Music Multidisciplinary Research* (Marseille, France, October 15 – 18, 2013), CMMR'13. 570-582.

[20] Papadopoulos, H.; Tzanetakis, G. 2012. Modeling chord and key structure with Markov logic. *Proc. of the 13th International Society for Music Information Retrieval Conference* (Porto, Portugal, October 8 – 12, 2012), ISMIR'12. 127-132.

[21] Scholz, R.; Vincent, E.; Bimbot, F. 2009. Robust modeling of musical chord sequences using probabilistic N-grams. *Proc. of the 34th IEEE International Conference on Acoustics, Speech and Signal Processing* (Taipei, Taiwan, April 19 – 24, 2009), ICASSP'09. 53–56.

[22] Sturm, B. L. A Simple Method to Determine if a Music Information Retrieval System is a "Horse". 2014. In *IEEE Transactions on Multimedia*, 16, 6 (July 2014), 1636-1644. DOI = 10.1109/TMM.2014.2330697.

[23] Urbano, J. 2011. Information Retrieval Meta-evaluation: Challenges and Opportunities in the Music Domain. In *Proc. of the 12th International Society for Music Information Retrieval Conference* (Miami, USA, October 24 – 28, 2011), ISMIR'11. 609-614.

[24] Urbano, J., Schedl, M. and Serra, X. 2013. Evaluation in Music Information Retrieval. *Journal of Intelligent Information Systems*, 41, 3 (December 2013), 345-369. DOI= 10.1007/s10844-013-0249-4.

[25] Yoshii, K.; Mauch, M.; Goto, M. 2011. A unified probabilistic model of note combinations and chord progressions. *Workshop Book of Neural Information Processing Systems, 4th International Workshop on Machine Learning and Music: Learning from Musical Structure* (Sierra Nevada, USA, December 17, 2011), MML'11. 46.

CROSS-COLLECTION EVALUATION FOR MUSIC CLASSIFICATION TASKS

Dmitry Bogdanov, Alastair Porter, Perfecto Herrera, Xavier Serra
Music Technology Group, Universitat Pompeu Fabra
name.surname@upf.edu

ABSTRACT

Many studies in music classification are concerned with obtaining the highest possible cross-validation result. However, some studies have noted that cross-validation may be prone to biases and that additional evaluations based on independent out-of-sample data are desirable. In this paper we present a methodology and software tools for cross-collection evaluation for music classification tasks. The tools allow users to conduct large-scale evaluations of classifier models trained within the AcousticBrainz platform, given an independent source of ground-truth annotations, and its mapping with the classes used for model training. To demonstrate the application of this methodology we evaluate five models trained on genre datasets commonly used by researchers for genre classification, and use collaborative tags from Last.fm as an independent source of ground truth. We study a number of evaluation strategies using our tools on validation sets from 240,000 to 1,740,000 music recordings and discuss the results.

1. INTRODUCTION

Music classification is a common and challenging Music Information Retrieval (MIR) task, which provides practical means for the automatic annotation of music with semantic labels including genres, moods, instrumentation, and acoustic qualities of music. However, many researchers limit their evaluations to cross-validation on small-sized datasets available within the MIR community. This leaves the question of the practical value of these classifier models for annotation, if the goal is to apply a label to any unknown musical input.

In the case of genre classification, Sturm counts that 42% of studies with experimental evaluation use publicly available datasets, including the famous GTZAN music collection (23%, or more than 100 works) and the ISMIR-2004 genre collection (17.4%), both of which contain no more than 1000 tracks. There is some evidence that such collections sizes are insufficient [7]. The MIREX annual evaluation includes a genre classification task, [1] which is currently run on a collection of 7000 tracks annotated with 10 genres, which is not publicly available to researchers. Some studies employ larger datasets, annotating genre using web-mined sources [11, 17, 18]. It has been shown that some datasets have flaws, including inconsistent and controversial labeling, absence of artist filters, and presence of duplicate examples (for example, GTZAN [20]).

Cross-validation is routinely used as a method for approximating a classifier's performance in real-world conditions, but such estimation is not free from some pitfalls. The ability of trained classifier models to generalize remains under question when following this method [14]. As some studies have noted, typical k-fold cross-validation on small-sized collections is prone to biases and additional evaluations based on independent out-of-sample data are desirable in order to avoid them [2, 8, 12]. Cross-collection validation is also suggested in other domains [1, 3–5, 13].

In order to address these problems and be able to better assess the performance of music classifier models, we propose a cross-collection evaluation process, that is, an evaluation of models on independent sets of music tracks annotated with an independent ground-truth source (which we call *validation sets*). In this paper we present a methodology and software tools for such evaluation for music classification tasks. We use AcousticBrainz, [2] a community-based platform for gathering music information from audio [16]. It contains MIR-related music features for over 3 million recordings including duplicates (we use the term *recording* to refer to a single analysis of music track). It provides the functionality to create datasets consisting of recordings and associated ground truth, training classifier models, and applying them to recordings present in AcousticBrainz. A number of models trained on genre datasets used within MIR are already included. Our tools allow the AcousticBrainz community to conduct cross-collection evaluations of classifier models trained on the AcousticBrainz website given any independent source of ground-truth annotations for recordings and a mapping between a model's classes and the classes within that ground truth.

In order to demonstrate the proposed methodology and tools, we evaluate five genre classifier models trained on MIR genre datasets. We build a genre ground truth for recordings in AcousticBrainz using collaborative tags from

© Dmitry Bogdanov, Alastair Porter, Perfecto Herrera, Xavier Serra. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Dmitry Bogdanov, Alastair Porter, Perfecto Herrera, Xavier Serra. "Cross-collection evaluation for music classification tasks", 17th International Society for Music Information Retrieval Conference, 2016.

[1] http://music-ir.org/mirex
[2] https://acousticbrainz.org

Last.fm [3] and consider various evaluation strategies for mapping the classifier models' outputs to the ground-truth classes. We use our tools on validation sets from 240,000 to 1,740,000 recordings and discuss the obtained results. Finally, we publish our genre ground-truth dataset on our website.

2. CROSS-COLLECTION EVALUATION METHODOLOGY

We define cross-collection evaluation to be an evaluation of a classifier model on tracks in a validation set annotated with an independent ground-truth source. We propose that the validation set is obtained or collected from a different source to the data used for training. This is distinct from holdout validation where a part of a dataset is used to verify the accuracy of the model. Because of the data's different origin it provides an alternative view on the final performance of a system. We develop tools based on this methodology for use in AcousticBrainz, because of the commodity of its infrastructure for building and evaluating classifiers, and the large amount of music recordings which it has analyzed.

2.1 Evaluation strategies

We consider various evaluation strategies concerning the comparison of a classifier's estimates and the ground-truth annotations in a validation set. A direct mapping of classes between the ground truth and the classifier is not always possible due to differences in their number, names, and actual meaning. Therefore, it is necessary to define a mapping between classes. Class names may imply broad categories causing difficulties in determining their actual coverage and meaning, and therefore inspection of the contents of the classes is advised. The following cases may occur when designing a mapping: a) a class in the classifier can be matched directly to a class in the validation set; b) several classes in the classifier can map to one in the validation set; c) one class in the classifier can map to many in the validation set; d) a class in the validation set cannot be mapped to any class in the classifier; e) a class in the classifier cannot be mapped to any class in the validation set. The case d) represents the subset of the validation set which is "out of reach" for the classifier in terms of its coverage, while e) represents the opposite, where the model is able to recognize categories unknown by the ground truth. We show an example of such a mapping in Section 3.3. The design of the mapping will affect evaluation results.

Validation sets may vary in their size and coverage and may contain a wider range of annotations than the classifier being evaluated. We consider the following strategies:

- **S1**: Use only recordings from the validation set whose ground truth has a matching class in the classifier. For example, if a recording is only annotated with the class electronic, and this class does not appear in the classifier, we discard it.
- **S2**: Use all recordings in the validation set and treat recordings from classes that do not exist in the classifier as an incorrect classification.

The validation set may have multiple class annotations per recording (e.g., in case of genre annotations, both pop and rock could be assigned to the same recording). Where the validation set has more than one ground-truth class for a recording we consider different methods of matching these classes to classifiers' estimates:

- **ONLY**: Only use recordings that have one ground-truth class, and discard the rest of the recordings when computing evaluation metrics.
- **ALL**: When a recording has more than one ground-truth class, accept an estimate as correct if it matches any of them, even though for the rest of the classes it would be considered a misclassification.

There may be duplicate recording representing the same music track (as is the case for AcousticBrainz, for which inconsistent classifier estimates have been observed). We consider two ways of dealing with them:

- **D1**: Remove all recordings that have duplicates from the evaluation.
- **D2**: Treat all recordings independently.

2.2 Evaluation metrics

Using class mappings one can compute confusion matrices for a classifier model for all combinations of S1/S2 with ONLY/ALL and D1/D2. The confusion matrix counts the percentage of correct classifier class estimates for each ground-truth class in the validation set. When a recording has more than one ground-truth class in method ALL, the recording is counted in all associated classes. Results are combined in the case when a class in the model is mapped to more than one class in the validation set. We estimate *accuracy*, the percentage of correctly recognized recordings. This value can be skewed due to difference in the sizes of each ground-truth class, and therefore we also compute *normalized accuracy* by scaling match counts according to the number of recordings within each class.

2.3 Tools for cross-collection evaluation

We have developed a set of tools as part of AcousticBrainz which let users train and evaluate classifier models. [4] Our tools let a user evaluate the quality of this model using an independent validation set. They can conduct any of the evaluation strategies mentioned above for any classifier model trained using AcousticBrainz.

To use our tools, a user first creates two datasets in AcousticBrainz. They define one dataset to be used to train a model, and the other to be used as the validation set. To ensure reliability of the accuracy results, the user can perform artist filtering [6, 15] during both the training and the

[3] http://www.last.fm

[4] We use the existing model training process, which selects best SVM parameters in a grid search using cross-validation and trains a model using all the data [10]. More details on the model training process are provided at https://acousticbrainz.org/datasets

cross-collection evaluation process. With artist filtering, during training we randomly select only one recording per artist for the model. On cross-collection evaluation, only recordings by artists different from the ones present in the training data are used. To get the artist of a recording we use the MusicBrainz API to request artist information for a recording using its MBID. Classes are randomly truncated to the same size, with a lower limit of 200 instances per class.

An interface lets the user map classes from the classifier dataset to classes in the validation set. In the case that there are no suitable matches, a class can be discarded from consideration during evaluation. The tools generate a report including statistics on the employed dataset and ground truth, accuracy values and confusion matrices. The results can be exported as HTML, LaTeX, or other machine-readable formats for further use. The tool is integrated into the AcousticBrainz server [5] to make this cross-collection evaluation process available for all AcousticBrainz users.

3. EVALUATION OF GENRE CLASSIFIER MODELS

We applied our evaluation methodology to assess five genre classifiers, three of which were already available within AcousticBrainz. We built two more classifiers using the dataset creator on the AcousticBrainz website with two previously published sources for genre annotation. We built an independent ground truth by mining folksonomy tags from Last.fm.

3.1 Music collections and classifier models

- **GTZAN Genre Collection (GTZAN)**. [6] A collection of 1000 tracks for 10 music genres (100 per genre) [23], including blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock. The GTZAN collection has been commonly used as a benchmark dataset for genre classification, due to it being the first dataset of this kind made publicly available [22].
- **Music Audio Benchmark Data Set (MABDS)**. [7] A collection of 1886 tracks retrieved from an online music community and classified into 9 music genres (46–490 per genre) [9] including alternative, electronic, funk/soul/rnb, pop, rock, blues, folk/country, jazz, and rap/hiphop.
- **AcousticBrainz Rosamerica Collection (ROS)**. A collection of 400 tracks for 8 genres (50 tracks per genre) including classical, dance, hip-hop, jazz, pop, rhythm & blues, rock, and speech (which is more a type of audio than a musical genre). The collection was created by a professional musicologist [7].
- **MSD Allmusic Genre Dataset (MAGD)**. A collection of genre annotations of the Million Song Dataset, derived from AllMusic [17]. We mapped MAGD to the AcousticBrainz collection, [8] which reduced the size of the dataset from 406,427 to 142,969 recordings and applied an artist filter (keeping only one recording per artist). The resulting dataset used for training contained 11 genres (206 tracks per genre) including pop/rock, electronic, rap, jazz, rnb, country, international, latin, reggae, vocal, and blues. [9]
- **Tagtraum genre annotations (TAG)**. A collection of genre annotations for Million Song Dataset, derived from Last.fm and beaTunes [18] (CD2C variation). As with MAGD, we mapped the dataset (reducing the size from 191,408 to 148,960 recordings) and applied an artist filter, resulting in 13 genres including rock, electronic, pop, rap, jazz, rnb, country, reggae, metal, blues, folk, world and latin (228 tracks per genre). [10]

Classifier	Accuracy	Normalized accuracy	Random baseline	Size	Number of classes
GTZAN	75.52	75.65	10	1000	10
MABDS	60.25	43.5	11.1	1886	9
ROS	87.56	87.58	12.5	400	8
MAGD	47.75	47.75	9.09	2266	11
TAG	47.87	47.87	7.69	2964	13

Table 1: Cross-validation accuracies (%) for all classifier models.

The models for GTZAN, MABDS, and ROS are provided in AcousticBrainz as baselines for genre classification but these collections were trained without artist filtering, as the recordings in these datasets are not associated with MBIDs. [11] We inspected available metadata for these collections and gathered non-complete artist lists for artist filtering in our cross-collection evaluation. We were able to identify 245 artists for 912 of 1000 recordings for GTZAN [19], 1239 artists for all 1886 recordings for MABDS, and 337 artists for 365 of 400 recordings for ROS.

Table 1 presents accuracies and normalized accuracies for the considered models obtained from cross-validation on training. The accuracies for GTZAN, MABDS, and ROS models are reported on the AcousticBrainz website. In general we observe medium to high accuracy compared to the random baseline. The results obtained for GTZAN are consistent with 78–83% accuracy observed for a number of other state-of-the-art methods on this collection [21]. Confusion matrices [12] do not reveal any significant misclassifications except for MABDS, for which alternative, funk/soul/rnb, and pop classes were frequently misclassified as rock (more than 35% of cases).

3.2 Preparing an independent ground truth

Last.fm contains tag annotations for a large number of music tracks by a community of music enthusiasts. While these tags are freeform, they tend to include commonly

[5] https://github.com/metabrainz/acousticbrainz-server

[6] http://marsyasweb.appspot.com/download/data_sets

[7] http://www-ai.cs.uni-dortmund.de/audio.html

[8] http://labs.acousticbrainz.org/million-song-dataset-mapping

[9] New age and folk categories were discarded due to insufficient number of instances after artist filtering keeping equal number of recordings per class.

[10] Similarly, new age and punk categories were removed.

[11] https://acousticbrainz.org/datasets/accuracy

[12] Available online: http://labs.acousticbrainz.org/ismir2016

recognized genres. We used the Last.fm API to get tag names and counts for all recordings in the AcousticBrainz database. Tag counts for a track are weighted, where the most applied tag for the track has a weight of 100. As tags are collaborative between users, we expect them to represent the "wisdom of the crowds". We obtained tags for 1,031,145 unique music tracks. Data includes 23,641,136 tag-weight pairs using 781,898 unique tags.

Genre tags as entered by users in Last.fm vary in their specificity (e.g., "rock", "progressive rock"). Meanwhile, the classifiers that we evaluate estimate broad genre categories (according to their class names). Therefore, we matched Last.fm tags to genres, and grouped these genres to broader "top-level" genres which we then matched to the output classes of our classifiers. We used a tree of genres [13] from beets, [14] a popular music tagger which uses data from MusicBrainz, Last.fm, and other sources to organize personal music collections. This genre tree is constructed based on a list of genres on Wikipedia, [15] further moderated by the community to add a wider coverage of music styles. [16] The tree includes 16 top-level genres: african, asian, avant-garde, blues, caribbean and latin american, classical, country, easy listening, electronic, folk, hip hop, jazz, pop, rhythm & blues, rock, and ska. [17] The taxonomy provided by the genre tree may not always be grounded on acoustical/musical similarity. For example, the asian top-level category includes both traditional music and western-influenced styles such as j-pop; jazz includes bebop, free jazz, and ragtime; electronic includes both electroacoustic music and techno. Similar inconsistencies in the contents of classes are not uncommon in genre datasets [18], and have been observed in GTZAN by [20], and also in our informal reviews of other datasets.

We matched tags directly to genres or subgenres in the tree and then mapped them to their top-level genre. Weights were combined for multiple tags mapped to the same top-level genre. Unmatched tags were discarded. We removed tracks where the weight of the top tag was less than 30, normalized the tags so that the top tag had a weight of 100 and again discarded tags with a weight of less than 30. After the cleaning process, we gathered genre annotations for 778,964 unique MBIDs, corresponding to 1,743,674 AcousticBrainz recordings (including duplicates).

3.3 Genre mappings

We created mappings between the classes of the classifier models and the validation set (Table 2). We created no mapping for 'disco' on GTZAN, 'international' and 'vocal' on MAGD, and 'world' on TAG as there were no clear matches. For ROS we did not map 'speech' as it did not represent any musical genre. Recordings estimated by classifiers as these classes were ignored during evaluation.

[13] https://github.com/beetbox/beets/blob/0c7823b4/beetsplug/lastgenre/genres-tree.yaml

[14] http://beets.io

[15] https://en.wikipedia.org/wiki/List_of_popular_music_genres

[16] The list from Wikipedia contains only modern popular music genres

[17] We discarded the comedy and other genres

3.4 Results

Using our tools, we performed an evaluation using sets from 240,000 to 1,740,000 recordings. Table 3 presents the accuracies and number of recordings used for the evaluation of each classifier model. We bring attention to the S2-ONLY-D1 strategy as we feel that it reflects a real world evaluation on a variety of genres, while being relatively conservative in what it accepts as a correct result (the ONLY-D1 variation). Also of note is the S1-ONLY-D1 strategy as it evaluates the classifiers on a dataset which reflects their capabilities in terms of coverage. We present confusion matrices for the S2-ONLY-D1 strategy in Table 4 for MAGD and TAG models (confusion matrices differ little across all evaluation strategies according to our inspection). [18]

Inspection of confusion matrices revealed a few surprising genre confusions. The MABDS model confuses all ground-truth genres with electronic (e.g., blues 64%, folk 62% of recordings misclassified). This tendency is consistent with inspection of this model's estimates in the AcousticBrainz database (81% estimated as electronic). No pattern of such misclassification was present in the confusion matrix during the training stage. Although this model was trained on unbalanced data, electronic was among the smallest sized classes (only 6% of the MABDS collection). Similarly, the GTZAN model tends to estimate all music as jazz (>73% of recordings of all genres are misclassified), which is again consistent with genre estimates in AcousticBrainz (90% estimated as jazz), with no such problems found during training.

The ROS model does not misclassify genres as harshly, confusing pop with rhythm & blues (26%), jazz with classical (21%), electronic with hip hop and rhythm & blues, jazz with rhythm & blues, and rhythm & blues with pop (<20% of cases for all confusions). For the MAGD model we see misclassifications of pop with rhythm & blues (21%), pop with carribean & latin and country, and rhythm & blues with blues (<15%). The TAG model performed better than MAGD, with no genre being misclassified for another more than 15% of the time, though we see a moderate amount of blues, electronic, folk, and pop instances being confused with rock as well as rhythm & blues with blues. The confusions for all three models make sense from musicological and computational points of view, evidencing how controversial genre-tagging can be, and that the computed features may not be specific enough to differentiate between genre labels.

3.5 Discussion

In general, considering exclusion/inclusion of the duplicate recordings in the evaluation (D1/D2), we observed that the differences in accuracy values are less than 4 percentage points for all models. We conclude that duplicates do not create any strong bias in any of our evaluation strategies even though the sizes of D1/D2 testing sets vary a lot.

[18] Complete results for all classifier models are available at: http://labs.acousticbrainz.org/ismir2016

Ground truth	GTZAN	MABDS	ROS	MAGD	TAG
african	-	-	-	-	-
asian	-	-	-	-	-
avant-garde	-	-	-	-	-
blues	blues	blues	-	blues	blues
caribbean and latin american	reggae	-	-	latin, reggae	latin, reggae
classical	classical	-	classical	-	-
country	country	folk/country	-	country	country
easy listening	-	-	-	-	-
electronic	-	electronic	dance	electronic	electronic
folk	-	folk/country	-	-	folk
hip hop	hip-hop	rap/hiphop	hip-hop	rap	rap
jazz	jazz	jazz	jazz	jazz	jazz
pop	pop	pop	pop	pop/rock	pop
rhythm and blues	-	funk/soul/rnb	rnb	rnb	rnb
rock	rock, metal	rock, alternative	rock	pop/rock	metal, rock
ska	-	-	-	-	-

Table 2: Mapping between classifier classes and ground-truth genres.

Model	Strategy	Recordings	Accuracy	Normalized accuracy
GTZAN	S1-ALL-D1	274895	13.68	12.78
	S1-ALL-D2	1235692	10.72	12.73
	S1-ONLY-D1	242346	13.27	12.95
	S1-ONLY-D2	1053670	10.35	12.91
	S2-ALL-D1	373886	10.06	6.39
	S2-ALL-D2	1623809	8.16	6.37
	S2-ONLY-D1	**292840**	**8.99**	**6.42**
	S2-ONLY-D2	1214253	6.93	6.37
MABDS	S1-ALL-D1	361043	31.76	17.52
	S1-ALL-D2	1660333	31.13	16.75
	S1-ONLY-D1	292220	28.39	18.44
	S1-ONLY-D2	1277695	27.44	17.96
	S2-ALL-D1	386945	29.63	9.86
	S2-ALL-D2	1743034	29.65	9.42
	S2-ONLY-D1	**302448**	**26.41**	**10.40**
	S2-ONLY-D2	1302343	25.82	10.15
ROS	S1-ALL-D1	320398	51.73	47.36
	S1-ALL-D2	1518024	50.12	45.32
	S1-ONLY-D1	269820	50.46	52.52
	S1-ONLY-D2	1229136	48.82	51.72
	S2-ALL-D1	379302	43.70	20.72
	S2-ALL-D2	1683696	45.19	19.83
	S2-ONLY-D1	**296112**	**43.12**	**23.58**
	S2-ONLY-D2	1252289	45.19	23.24
MAGD	S1-ALL-D1	323438	59.35	42.13
	S1-ALL-D2	1505105	59.91	40.41
	S1-ONLY-D1	265890	59.56	48.34
	S1-ONLY-D2	1184476	60.83	48.40
	S2-ALL-D1	347978	55.92	23.70
	S2-ALL-D2	1590395	57.36	22.73
	S2-ONLY-D1	**272426**	**56.36**	**27.35**
	S2-ONLY-D2	1187287	58.94	27.56
TAG	S1-ALL-D1	327825	59.85	44.46
	S1-ALL-D2	1482123	60.58	42.12
	S1-ONLY-D1	265280	59.51	52.97
	S1-ONLY-D2	1139297	60.49	52.77
	S2-ALL-D1	342544	57.35	27.79
	S2-ALL-D2	1532129	58.67	26.32
	S2-ONLY-D1	**268543**	**56.94**	**33.13**
	S2-ONLY-D2	1147231	58.62	33.10

Table 3: Cross-collection evaluation accuracies (%) for all classifier models.

Similar conclusions can be made with respect to the inclusion of recordings with conflicting genre ground truth (ONLY/ALL). Conflicting cases of genre annotations only account for 21% of our validation set. In the case of our Last.fm annotations, multiple labeling of ground truth does not affect our results, still, one should explore both strategies to ensure that the same holds for other ground truths.

The only large difference in accuracy values is observed when comparing S1 and S2 strategies—S1 yields higher accuracies as all additional recordings in S2 are considered incorrect no matter what the classifier selects. Normalized accuracy allows us to assess the performance of a classifier given a hypothetical validation set with an equal number of instances per class. In our S2 strategy, many validation set classes not matched to a classifier class, and therefore considered incorrect, contained a small number of recordings (e.g., african, asian, avant-garde, and easy listening; see Table 4). Because of this we observe a larger difference in normalized accuracies between S1 and S2.

Based on the results we conclude that the models for ROS, MAGD and TAG perform the best. Their normalized accuracies are two times better than other classifiers under any condition. Interestingly, the ROS model is trained on the smallest collection (400 tracks, compared to 1000 tracks for GTZAN and 1886 tracks for MABDS, and over 2000 for MAGD an TAG), while we expected that it would suffer from insufficient training size.

What can be the reason for such a differing performances of models? MAGD uses as its source genre annotations made by experts from Allmusic, while the ROS collection was created by a musicologist specifically for the task of content-based music classification, which may be the reason for their better performance. The annotations in TAG were taken from two different sources, and were only used when both sources agreed on the genre [18].

4. CONCLUSIONS

The majority of studies on music classification rely on estimating cross-validation accuracy on a single ground truth, while it has been criticized as being shortsighted to shed light on the real capacity of a system to recognize music categories [21]. In our study we go beyond this approach and show an additional way to ascertain the capacity of classifiers by evaluating across collections. We believe that cross-collection generalization is an interesting metric to

Ground-truth genre	size (%)	Estimated genre: blues (blues)	carribean & latin (latin, reggae)	country (country)	electronic (electronic)	hip hop (rap)	jazz (jazz)	pop, rock (pop/rock)	rhythm & blues (rnb)
blues	2.7	**48.30**	8.54	11.27	2.88	1.31	8.94	13.16	5.61
carribean & latin	1.9	6.98	**57.11**	4.70	5.64	6.88	7.44	3.04	8.20
country	4.3	14.09	8.28	**61.89**	1.00	0.43	2.24	8.38	3.67
electronic	20.1	2.01	4.57	0.90	**59.28**	6.22	4.35	16.22	6.45
hip hop	2.3	1.22	10.39	0.14	8.77	**65.63**	1.08	3.80	8.97
jazz	7.7	9.39	7.00	3.68	2.87	1.02	**67.22**	4.60	4.22
pop	5.8	4.63	18.65	18.23	7.12	3.47	2.87	**23.98**	21.05
rhythm & blues	3.4	16.36	13.87	11.88	3.66	5.93	5.00	10.74	**32.56**
rock	43.9	7.60	6.26	6.10	7.14	0.82	2.48	**67.38**	2.23
african	0.3	13.09	33.54	10.22	8.38	6.13	9.20	4.50	14.93
asian	1.2	2.13	16.85	7.72	8.35	2.58	3.66	35.34	23.36
avant-garde	0.2	14.33	8.26	5.51	12.40	5.92	20.11	30.72	2.75
classical	2.2	4.85	4.05	5.73	4.30	0.45	66.05	13.86	0.70
easy listening	0.4	9.52	13.79	27.59	6.62	2.21	18.07	14.06	8.14
folk	3.1	15.55	14.80	28.92	5.75	0.89	10.71	20.14	3.24
ska	0.7	4.80	39.28	5.71	13.90	9.21	2.16	19.66	5.28

(a) MAGD. Columns for pop and rock are summed together as they match the same model class

Ground-truth genre	size (%)	Estimated genre: blues (blues)	carribean & latin (latin, reggae)	country (country)	electronic (electronic)	folk (folk)	hip hop (rap)	jazz (jazz)	pop (pop)	rhythm & blues (rnb)	rock (metal, rock)
blues	2.7	**48.09**	4.70	6.57	1.08	7.17	0.63	9.55	3.45	6.72	12.04
carribean & latin	1.9	3.47	**59.11**	3.26	2.93	4.37	5.56	6.96	4.47	6.77	3.12
country	4.3	11.58	5.29	**51.90**	0.37	11.15	0.15	2.55	6.13	3.18	7.70
electronic	20.1	0.82	4.53	0.81	**53.66**	5.45	5.82	2.62	9.33	3.33	13.63
folk	3.1	6.03	4.46	11.23	3.20	**47.22**	0.35	5.30	7.53	2.59	12.09
hip hop	2.3	0.83	9.14	0.11	7.03	0.31	**67.00**	1.52	2.30	8.75	3.00
jazz	7.7	7.68	5.19	2.15	1.43	4.93	0.71	**65.56**	3.28	6.05	3.02
pop	5.8	3.76	9.24	11.43	4.02	8.56	2.40	4.10	**35.78**	8.38	12.32
rhythm & blues	3.4	12.59	11.40	8.37	1.96	3.73	4.45	5.29	11.82	**31.94**	8.46
rock	43.9	4.19	2.65	3.61	2.95	5.15	0.57	1.59	7.54	1.99	**69.76**
african	0.3	12.43	29.94	5.37	5.08	14.12	5.65	7.34	7.34	8.19	4.52
asian	1.2	1.63	6.64	3.47	4.52	3.94	2.67	1.98	52.76	6.12	16.26
avant-garde	0.2	8.38	5.67	2.47	9.25	14.43	4.19	16.40	4.07	4.93	30.21
classical	2.2	5.78	1.64	4.76	1.73	30.75	0.16	41.08	5.38	2.19	6.53
easy listening	0.4	6.81	7.07	17.38	2.97	21.05	0.79	14.76	15.20	5.94	8.03
ska	0.7	3.00	42.24	4.00	9.71	0.75	9.66	1.15	5.36	3.55	20.57

(b) TAG

Table 4: Confusion matrices for S2-ONLY-D1 strategy. Original class names for the classifiers are listed in parentheses. Misclassifications >10% are shaded light gray and >20% dark gray.

take into account for validating the robustness of classifier models. We propose a methodology and software tools for such an evaluation. The tools let researchers conduct large-scale evaluations of classifier models trained within AcousticBrainz, given an independent source of ground-truth annotations and a mapping between the classes.

We applied our methodology and evaluated the performance of five genre classifier models trained on MIR genre collections. We applied these models on the AcousticBrainz dataset using between 240,000 and 1,740,000 music recordings in our validation sets and automatically annotated these recordings by genre using Last.fm tags. We demonstrated that good cross-validation results obtained on datasets frequently reported in existing research may not generalize well. Using any of the better-performing models on AcousticBrainz, we can only expect a 43–58% accuracy according to our Last.fm ground truth when presented with any recording on AcousticBrainz. We feel that this is still not a good result, and highlights how blurred the concept of genres can be, and that these classifiers may be "blind" with respect to some important musical aspects defining some of the genres. More research effort is required in designing musically meaningful descriptors and making them error-resistant, as well as understanding the relationships between different genre taxonomies.

Importantly, the application of the proposed methodology is not limited to genres and can be extended to other classification tasks. In addition to the proposed methodology and tools, we release a public dataset of genre annotations used in this study. [19] In our future work we plan to investigate and publish more independent sources of ground-truth annotations, including annotations by genre and mood, that will allow researchers to conduct more thorough evaluations of their models within AcousticBrainz.

5. ACKNOWLEDGEMENTS

This research has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 688382. We also thank Gabriel Viglensoni for help with mining Last.fm tags.

[19] https://labs.acousticbrainz.org/lastfm-genre-annotations

6. REFERENCES

[1] J. Bekios-Calfa, J. M Buenaposada, and L. Baumela. Revisiting linear discriminant techniques in gender recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):858–864, 2011.

[2] D. Bogdanov, J. Serrà, N. Wack, P. Herrera, and X. Serra. Unifying low-level and high-level music similarity measures. *IEEE Transactions on Multimedia*, 13(4):687–701, 2011.

[3] V. Chudáček, G Georgoulas, L. Lhotská, C Stylios, M. Petrík, and M Čepek. Examining cross-database global training to evaluate five different methods for ventricular beat classification. *Physiological measurement*, 30(7):661–677, 2009.

[4] N. Erdogmus, M. Vanoni, and S. Marcel. Within-and cross-database evaluations for gender classification via befit protocols. In *International Workshop on Multimedia Signal Processing (MMSP'14)*, pages 1–6, 2014.

[5] C. Fernández, I. Huerta, and A. Prati. A comparative evaluation of regression learning algorithms for facial age estimation. In *Face and Facial Expression Recognition from Real World Videos*, pages 133–144. Springer, 2015.

[6] A. Flexer and D. Schnitzer. Effects of album and artist filters in audio similarity computed for very large music databases. *Computer Music Journal*, 34(3):20–28, 2010.

[7] E. Guaus. *Audio content processing for automatic music genre classification: descriptors, databases, and classifiers*. PhD thesis, Universitat Pompeu Fabra, 2009.

[8] P. Herrera, A. Dehamel, and F. Gouyon. Automatic labeling of unpitched percussion sounds. In *AES 114th Convention*, 2003.

[9] H. Homburg, I. Mierswa, B. Möller, K. Morik, and M. Wurst. A benchmark dataset for audio classification and clustering. In *International Conference on Music Information Retrieval (ISMIR'05)*, pages 528–531, 2005.

[10] C. W. Hsu, C. C. Chang, and C. J. Lin. A practical guide to support vector classification. 2003.

[11] D. Liang, H. Gu, and B. O'Connor. Music genre classification with the Million Song Dataset. Technical report, Carnegie Mellon University, 2011.

[12] A. Livshin and X. Rodet. The importance of cross database evaluation in musical instrument sound classification: A critical approach. In *International Conference on Music Information Retrieval (ISMIR'03)*, 2003.

[13] M. Llamedo, A. Khawaja, and J. P. Martínez. Cross-database evaluation of a multilead heartbeat classifier. *IEEE Transactions on Information Technology in Biomedicine*, 16(4):658–664, 2012.

[14] A. Y Ng. Preventing" overfitting" of cross-validation data. In *International Conference on Machine Learning (ICML'97)*, pages 245–253, 1997.

[15] E. Pampalk, A. Flexer, and G. Widmer. Improvements of audio-based music similarity and genre classificaton. In *International Conference on Music Information Retrieval (ISMIR'05)*, pages 628–633, 2005.

[16] A. Porter, D. Bogdanov, R. Kaye, R. Tsukanov, and X. Serra. AcousticBrainz: a community platform for gathering music information obtained from audio. In *International Society for Music Information Retrieval Conference (ISMIR'15)*, 2015.

[17] A. Schindler, R. Mayer, and A. Rauber. Facilitating comprehensive benchmarking experiments on the million song dataset. In *International Society for Music Information Retrieval Conference (ISMIR'12)*, pages 469–474, 2012.

[18] H. Schreiber. Improving genre annotations for the million song dataset. In *International Society for Music Information Retrieval Conference (ISMIR'15)*, 2015.

[19] B. L Sturm. The state of the art ten years after a state of the art: Future research in music information retrieval. *Journal of New Music Research*, 43(2):147–172, 2014.

[20] B.L. Sturm. An analysis of the GTZAN music genre dataset. In *International ACM Workshop on Music Information Retrieval with User-centered and Multimodal Strategies (MIRUM'12)*, pages 7–12, 2012.

[21] B.L. Sturm. Classification accuracy is not enough. *Journal of Intelligent Information Systems*, 41(3):371–406, 2013.

[22] B.L. Sturm. A survey of evaluation in music genre recognition. In *Adaptive Multimedia Retrieval: Semantics, Context, and Adaptation*, pages 29–66. Springer, 2014.

[23] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

DOWNBEAT DETECTION WITH CONDITIONAL RANDOM FIELDS AND DEEP LEARNED FEATURES

Simon Durand, Slim Essid
LTCI, CNRS, Télécom ParisTech
Université Paris-Saclay, 75013, Paris, France
simon.durand@telecom-paristech.fr

ABSTRACT

In this paper, we introduce a novel Conditional Random Field (CRF) system that detects the downbeat sequence of musical audio signals. Feature functions are computed from four deep learned representations based on harmony, rhythm, melody and bass content to take advantage of the high-level and multi-faceted aspect of this task. Downbeats being dynamic, the powerful CRF classification system allows us to combine our features with an adapted temporal model in a fully data-driven fashion. Some meters being under-represented in our training set, we show that data augmentation enables a statistically significant improvement of the results by taking into account class imbalance. An evaluation of different configurations of our system on nine datasets shows its efficiency and potential over a heuristic based approach and four downbeat tracking algorithms.

1. INTRODUCTION

Musical rhythm can often be organized in several hierarchical levels. These levels don't always correspond to musical events and have a regular temporal interval that can change over time to follow the musical tempo. One of these levels is the tatum level and is at the time scale of onsets. The next one is often the beat level and can be intuitively understood as the hand clapping or foot tapping level. Then in several music traditions there is the bar level that groups beats of different accentuation together. The first beat of the bar is called the downbeat. The aim of this work is to automatically find the downbeat positions from musical audio signals. The downbeat is useful to musicians, composers and conductors to segment, navigate and understand music more easily. Its automatic estimation is also useful for various applications in music information retrieval, computer music and computational musicology.

This task is receiving more attention recently. With the increasing number of annotated music files and refined learning strategies, methods using probabilistic models and machine learning algorithms tend to be the most successful [4, 13, 18]. Once a downbeat detection function has been extracted, most systems use a temporal model to take advantage of the structured organization of downbeats and output the downbeat sequence. It includes heuristics [3], dynamic programming [25], hidden Markov models [23] and particle filters [18] among others.

In this work, we propose for the first time a Conditional Random Field (CRF) framework for the task of downbeat tracking. First, four complementary features related to harmony, rhythm, melody and bass content are extracted and the signal is segmented at the tatum level. Adapted convolutional neural networks (CNN) to each feature characteristics are then used for feature learning. Finally, a feature representation concatenated from the last and/or penultimate layer of those networks is used to define observation feature functions and is fed into a Markovian form of CRF that will output the downbeat sequence.

1.1 Related work

A CRF framework is used in [7] and [16] for the field of beat tracking. However, the optimal weights of the observations and transitions feature functions are not directly learned from the data.

The system presented in [14] uses an interesting idea of limiting engineered hypotheses by segmenting the data in onsets and learning the activation of downbeats with a Support Vector Machine classifier. Contrary to our work, it requires manual annotation of either the first part of the tested song or of a very similar song and outputs an intermediary downbeat activation function as opposed to the final downbeat positions.

In [6], the same segmentation, low-level feature extraction and complementary CNNs are used. However, the proposed system includes three main differences:

- We are not only using an individual output per downbeat candidate but a detailed high-level representation also coming from the penultimate layer of the neural networks. Besides, we don't optimize individual features on isolated downbeat occurrences, but features from all the deep networks simultaneously on a whole structured downbeat sequence.

- We are using another type of classifier, namely CRF, known to be more effective than Hidden Markov

© Simon Durand, Slim Essid. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Simon Durand, Slim Essid. "Downbeat Detection with Conditional Random Fields and Deep Learned Features", 17th International Society for Music Information Retrieval Conference, 2016.

Models especially in high dimensional settings due to being a discriminative classifier.

- A fully data driven approach, after extracting low-level features, is adopted. It takes advantage of data augmentation procedures to allow for a proper training of the CRF classifiers, limiting the use of ad-hoc heuristics and hand-crafted data transformations.

2. FEATURE LEARNING

The feature learning part of our system is the same as in [6]. We first segment the audio signal in tatums as seen in figure 1. We then simplify the downbeat detection task to a classification problem where the goal is to find which tatums are at a downbeat position. Human perception of downbeats depending on several musical cues, we then extract four low-level features related to melody, rhythm, harmony and bass content. Each low-level feature input, shown in figure 2, is fed to a convolutional neural network adapted to its characteristics. The bass content neural network (BCNN) targets melodic and percussive bass instruments. The melodic neural network (MCNN) targets relative melodic patterns which are known to play a role in human perception of meter regardless of their absolute pitch [29] with max pooling. The harmonic neural network (HCNN) learns how to detect harmonic change in the input and is trained on all different harmony transposition by data augmentation. Finally, the rhythmic neural network (RCNN) aims at learning length specific rhythmic patterns with multi-label learning, instead of sudden changes in the rhythm feature that are not very indicative of a downbeat position. For more details about the motivations behind the design choices made for each network the interested reader is referred to [6].

3. CRF SYSTEM FOR DOWNBEAT TRACKING

Two high-level feature representations coming from the last and penultimate layer of each network are then used as input to a Conditional Random Field (CRF) classifier.

3.1 CRF-based classification

CRF [19] are a powerful class of discriminative classifiers for structured input–structured output data prediction, which have proven successful in a variety of real-world classification tasks [26, 27] and also in combination with neural networks [24]. They model directly the posterior probabilities of output sequences $\underline{\mathbf{y}} = (y_1, \cdots, y_n)$ given input observation sequences $\underline{\mathbf{x}} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)$ according to:

$$p(\underline{\mathbf{y}}|\underline{\mathbf{x}};\boldsymbol{\theta}) = \frac{1}{Z(\underline{\mathbf{x}},\boldsymbol{\theta})} \exp \sum_{j=1}^{D} \theta_j G_j(\underline{\mathbf{x}},\underline{\mathbf{y}})$$

where $G_j(\underline{\mathbf{x}},\underline{\mathbf{y}})$ are feature functions describing the observations, θ_j are the model parameters (assembled as $\boldsymbol{\theta} = [\theta_j]_{1\leq j\leq D}$), and $Z(\underline{\mathbf{x}})$ is a normalizing factor that guarantees that $p(\underline{\mathbf{y}}|\underline{\mathbf{x}})$ is a well defined probability, which sums to 1.

Owing to the sequential nature of the downbeat classification problem, we use a Markovian form of CRF, where the transition feature functions, denoted by t_j, are defined on two consecutive labels, in a linear-chain fashion, and observation feature functions, denoted by v_j, depend on single labels, so that:

$$p(\underline{\mathbf{y}}|\underline{\mathbf{x}};\boldsymbol{\theta}) = \frac{1}{Z(\underline{\mathbf{x}},\boldsymbol{\theta})} \exp \left\{ \sum_{i=1}^{n} \sum_{j=1}^{D_o} \theta_j v_j(y_i, \underline{\mathbf{x}}, i) + \sum_{i=1}^{n} \sum_{j=1}^{D_t} \theta_j t_j(y_{i-1}, y_i, \underline{\mathbf{x}}, i) \right\}. \quad (1)$$

More specifically, the transition feature functions we use are such that $t_j(y_{i-1} = k, y_i = l, \underline{\mathbf{x}}, i) = \mathbb{I}(y_i = l)\mathbb{I}(y_{i-1} = k)$, where $\mathbb{I}(.)$ is the indicator function (equal to 1 if its argument is true and otherwise equal to 0). As for the observation feature functions they are chosen to be of the form $v_j(y_i = l, \underline{\mathbf{x}}, i) = e_j \mathbb{I}(y_i = l)$ where e_j are obtained by the feature representation learned by the networks presented in section 2. Actually, two schemes are envisaged here. In the first variant, the e_j features are taken to be directly the final outputs of the bass, melodic, harmonic and rhythmic networks. Alternatively, we added the output of the penultimate layer [1] which can be viewed as lower level features that were optimized, as part of the network training processes, to discriminate downbeats from tatums. The deep network penultimate layer output is a powerful feature representation that can be used as an input to a dedicated classifier to improve accuracy [9]. The last layer of our networks being essentially a linear combination of the penultimate layer features followed by a normalization to map them to probabilities, the CRF classifier is a good fit for the final weighting of those features, based on the more optimal output-sequence level maximum a posteriori criterion $p(\underline{\mathbf{y}}|\underline{\mathbf{x}};\theta)$, compared to the static criterion optimized in the last layer of the networks. The harmonic network penultimate layer dimension is of 1000 and each of the other networks penultimate layer dimension is of 800.

3.2 Defining the output-space

The set of output labels Y_i^j represents the position i of a tatum in a j tatum-long bar, with $i \in \{1...j\}$ and $j \in \{3, 4, 5, 6, 7, 8, 9, 10, 12, 16\}$. We consider an additional label for bars containing more than 16 tatums for a total of 81 labels. This way, the feature function weights depend on the bar length, in tatums, and the position inside the bar. For instance, the sixth tatum of a 6 tatum-long bar Y_6^6 and the sixth tatum of a 8 tatum-long bar Y_6^8 have different musical properties. In the first case, we want the transition feature functions to emphasize the next output to be the first tatum of a 6 tatum-long bar Y_1^6. In the second case, we want to emphasize the next output to be the seventh tatum of a 8 tatum-long bar instead Y_7^8. The observation features, taking into account one or two bars of

[1] before the ReLU to keep information about negative units

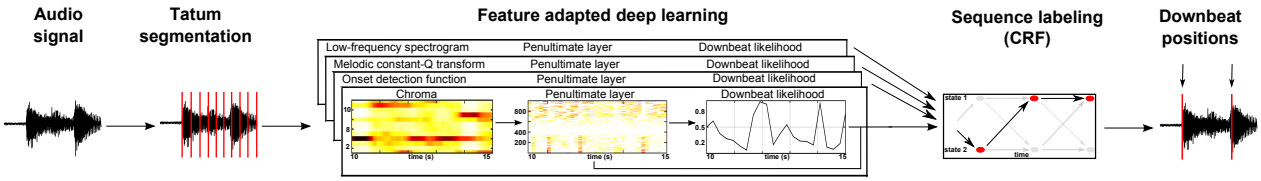


Figure 1. Model overview. The signal is quantized in tatums. Four low-level features related to harmony, rhythm, melody and bass content are extracted. High-level feature representations are learned with four convolutional networks adapted to each feature characteristics. The networks penultimate layer, along with the downbeat likelihood, are fed in a CRF to find the downbeat sequence among the tatums.

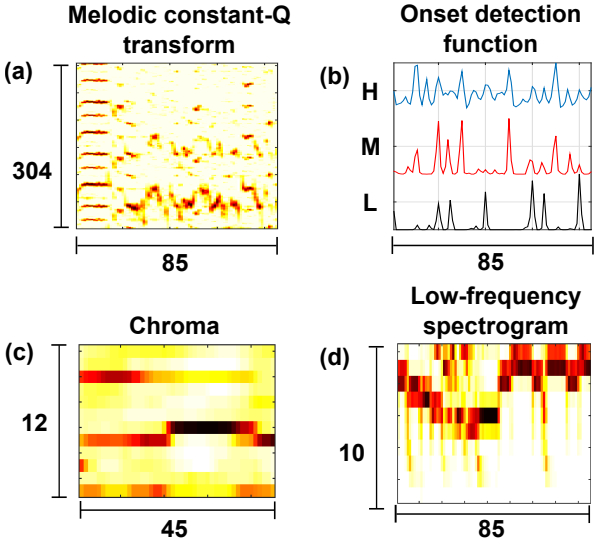


Figure 2. Low-level features with their temporal and spectral dimension, used as input of the melodic neural network (a), rhythmic neural network (b), harmonic neural network (c) and bass content neural network (d).

temporal context, will also be rather different and are better treated separately. It is therefore important to distinguish those two outputs for a consistent decoding.

3.3 Labeling the training data

The training data being only annotated in beat and downbeat, defining its labels is not straightforward. First, bars of 2, 11, 13, 14 and 15 tatums are not present in our model for efficiency, robustness and because they are barely present in most music datasets but they can't be ignored to train the model efficiently. They are then annotated to the most common neighbor bar-length: 14 and 15 tatum-long bars are annotated as 16 tatum-long bars. 11 and 13 tatum-long bars are annotated as 12 tatum-long bars. The last state of those metrical levels is either removed or repeated to do so. 2 tatum-long bars are annotated as 3 tatum-long bars if the following bar is a 3 tatum-long bar for continuity or as 4 tatum-long bar otherwise as they are the most common neighbor for a duple meter.

Second, the beginning and end of songs are sometimes not properly estimated or annotated and considering or ignoring all observations before the first or after the last downbeat can lead to training problems. For the beginning of songs, we removed the samples that where more than one bar before the first annotated downbeat as they were not reliable enough. We then annotated the bar preceding the first downbeat with the same classes than the bar containing the first downbeat for continuity, and finally removed samples in this first bar randomly. It allows the initialization of the position inside the bar to be randomized. The procedure is applied in reverse for the end of songs.

Although extensive tests were not performed, we obtain a gain in performance by about 4 percent points (pp) by using this annotation process compared to a simple representation of all these non conventional cases by an additional label.

3.4 Handling class-imbalance with data augmentation

Not all metrical level are well represented in the used datasets. In fact, {3,4,6,8,12,16} tatum-long bars, i.e. bars of 3 and 4 beats, represent more than 96% of the data and will be the focus of the CRF model. In this subset, bars of 3 beats will be represented by 3, 6, and roughly half of 12 tatum-long bars. This represents approximately 15% of the data. Those metrical levels are then non negligible but under-represented. Such data imbalance is known to create difficulties while training classifiers like CRFs. We therefore balance our dataset with data augmentation. We use time-stretching by a factor of 1.1 and 0.9 and pitch shifting by ± 1 semitone on 3-beats-per-bar songs to do so. The implementation is done thanks to the *muda* package presented in [20]. We will study in the experiments the added value of the data augmentation.

4. EXPERIMENTAL SETUP

4.1 Evaluation methods

We use the F-measure and a statistical test to assess the performance of our system:

F-measure: The F-measure is the harmonic mean of the precision (ratio of detected downbeat that are relevant) and the recall (the ratio of relevant downbeat detected). It is an instantaneous measure of performance that is used in the MIREX downbeat tracking evaluation [2]. We use a

[2] http://www.music-ir.org/mirex/wiki/2016:

tolerance window of $\pm$70ms. The configuration with the best F-measure with be highlighted in bold. We do not take into account the first 5 seconds and last 3 seconds of audio in our evaluation metric since the annotation is sometimes missing or not very reliable there.

Statistical tests: To assess statistical significance, we perform a Friedman's test and a Tukey's honestly significant criterion (HSD) test with a 95% confidence interval. System(s) with a statistically significant improvement over the rest on the whole dataset will be underlined.

4.2 Databases

We use nine different databases in this work, for a total of 1511 audio tracks of about 43 hours of audio music. Using multiple datasets allows us to see the performance of our system on different music styles and be robust to different annotation strategies.

RWC Classical [10]: 60 western classical pieces, from 1 to 10 minutes. We removed the last track as the annotation seemed inconsistent.

RWC Jazz [10]: 50 jazz tracks from 2 to 7 minutes.

RWC music genre [11]: 92 music tracks from various music styles, from 1 to 10 minutes. We removed the traditional Japanese songs and the a Capella song as we don't have the corresponding audio.

RWC Pop [10]: 80 Japanese Pop music and 20 American Pop music tracks from 3 to 6 minutes.

Beatles [3] **:** 179 songs from The Beatles.

Ballroom [4] **:** 698 30-second long excerpts from various ballroom dance music.

Hainsworth [12]: 222 excerpts from 30 second to 1 minute from various music styles. It is to note that the current downbeat annotation can significantly be improved.

Klapuri subset [15]: The downbeat annotations for this dataset are lacking in some files. Full cleaning will be done in future work but we use a subset of 4 relatively difficult genres for downbeat tracking : Jazz, Electronic music, Classical and Blues with 10 randomly selected excerpts for each genre.

Quaero [5] **:** 70 songs from various Pop, Rap and Electronic music hits.

4.3 General train/test procedure

We use a leave-one-dataset-out approach, meaning that we train and validate our system on all but one dataset and test it on the remaining one. Compared to standard cross-validation, this procedure was chosen to be more fair to non machine learning methods that are blind to the test set and to supervised algorithms using the same approach. However, it is limiting the ability of the deep networks and the CRF model to work on test data from styles not often seen in the training set. Two notable examples are the RWC Classical and RWC Jazz music datasets.

Audio_Downbeat_Estimation

[3] http://isophonics.net/datasets

[4] http://www.ballroomdancers.com

[5] http://www.quaero.org

Dataset	ll	ll + da	pl	pl + da
RWC Jazz	65.3	66.0	65.5	**66.1**
RWC Class	44.3	44.3	43.8	**45.9**
Hainsworth	62.9	65.9	64.5	**66.0**
RWC Genre	66.2	68.1	69.1	**69.3**
Klapuri subset	67.1	71.2	67.4	**71.5**
Ballroom	78.0	77.3	79.0	**80.9**
Quaero	83.5	**83.8**	83.1	82.7
Beatles	84.0	84.1	84.4	**85.2**
RWC Pop	87.2	85.1	86.7	**87.4**
Mean	70.9	71.8	71.5	**72.8**

Table 1. F-measure results for different configurations of the presented system. ll means the features come from the network last layer and pl means that features from the penultimate layer were also used. da means data augmentation was used.

4.4 CRF training

For CRF training we use the Pycrfsuite toolbox [21]. The CRF parameters are learned as classically done in a maximum likelihood sense using both ℓ_2 and ℓ_1-regularisation, thus in an elastic-net fashion, so as to promote sparse solutions, and solved for using the L-BFGS algorithm. The optimal values of the regularisation parameters were selected by a 4-fold cross-validation on the training set. For the last layer features, the grid for the optimal ℓ_2 value is [100,10,1.0,0.1,0.01,0.001,0.0001]. Since there are only four features out of the networks, we don't need feature selection and the ℓ_1 parameter was set to 0. When adding the penultimate layer features, the grids for the optimal ℓ_1 and ℓ_2 values were [10,100] and [0.1,0.01,0.001] respectively.

5. RESULTS AND DISCUSSION

5.1 Impact of the data augmentation:

Configuration using data augmentation will be abbreviated by "da", and their F-measure results for each dataset is shown in table 1. We can see an improvement on all datasets, except on the RWC Pop and Quaero datasets. Indeed, the number of songs containing 3 or 6 tatums per bar is very limited there. Overall the F-measure improvement is of +0.9 percent point using last layer features (abbreviated by "ll") and of +1.3 pp using penultimate layer features (abbreviated by "pl").

5.2 Impact of the penultimate layer:

F-measure results of configurations adding the penultimate layer output as features is also shown in table 1. Using the penultimate layer increases the results overall by 0.6 pp with the non augmented data and by 1.0 pp with the

Dataset	[23]	[22]	[3]	[17]	[6]	pl + da
RWC Jazz	39.6	47.2	42.1	51.5	**70.9**	66.1
RWC Class	29.9	21.6	32.7	33.5	**51.0**	45.9
Hainsworth	42.3	47.5	44.2	51.7	65.0	**66.0**
RWC Genre	43.2	50.4	49.3	47.9	66.1	**69.3**
Klapuri	47.3	41.8	41.0	50.0	67.4	**71.5**
Ballroom	45.5	50.3	50.0	52.5	80.1	**80.9**
Quaero	57.2	69.1	69.3	71.3	81.2	**82.7**
Beatles	53.3	66.1	65.3	72.1	83.8	**85.2**
RWC Pop	69.8	71.0	75.8	72.1	**87.6**	87.4
Mean	47.6	51.7	52.2	55.8	72.6	**72.8**

Table 2. F-measure results for compared algorithms. [23], [22] and [3] are unsupervised. [17] and [6] are supervised algorithms also trained with a leave-one-dataset-out approach. [6] uses the same training sets and [17] uses similar training sets, with the addition of the Boeck [1, 2], Rock [28] and Robbie Williams [8] datasets and the subtraction of the Klapuri subset and the Quaero dataset.

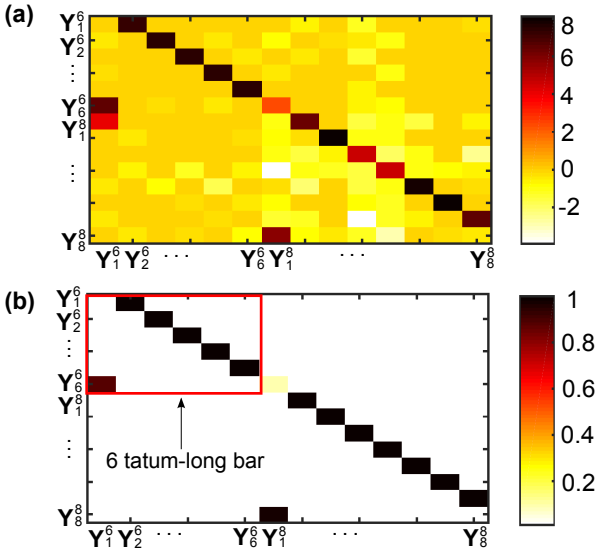


Figure 3. Selected transition weights for the 6 and 8 tatum-long bars. It corresponds to the output labels Y_1^6 to Y_6^6 and Y_1^8 to Y_8^8. (a) Weight of the transition feature function in the presented CRF model. (b) Coefficients of the transition matrix in [6]. As an illustration, inside the red rectangle pointed by an arrow are all the coefficients corresponding to a transition inside a 6 tatum-long bar. There is a weight at the bottom left corner of this rectangle with a value close to 1. It corresponds to the weight of the transition from Y_6^6 to Y_1^6.

augmented data. Its impact on the bigger datasets (Ballroom, Beatles, RWC Pop, RWC Genre, Hainsworth), representing 85% of the songs is more important than for the smaller datasets. Besides, using both the data augmentation and the penultimate layer allows the CRF model to have the best performance on all datasets but one, and to have a statistically significant improvement over the other configurations.

5.3 Comparison to other algorithms:

We compared our best system to the ones of Krebs et al. [17], Peeters et al. [23], Davies et al. [3] and Papadopoulos et al. [22]. We also compared to a system using the same neural networks but with a different feature combination and temporal model [6]. In this system, the output of the four networks is averaged and a Viterbi model with hand-crafted transition and emission probabilities is used to decode the downbeat sequence. Due to space constraints, we do not add [4] and [5] since they are close to [6] in terms of architecture and produce worse results. Results are shown in the table 2. With the new CRF system proposed here, the improvement is substantially better in all datasets compared to [17], [23], [3] and [22]. While the improvement averaged across datasets is moderate compared to [6], we observe a statistically significant improvement [6] . Overall results are held back by the performance on RWC Classical and RWC Jazz. The used training sets barely contain these music styles while we are exploiting a fully data-driven approach. The leave-one-dataset-out approach might be too restrictive when dealing with very distinctive music datasets. However, when more appropriate training data is available, the CRF model has a better potential, as results on RWC Genre indicates. This dataset includes more than 30% of Jazz and Classical music songs and has a significantly better performance with the new temporal model (69.3% F-measure compared to 66.1% in [6]). In this case the RWC Jazz and RWC Classical datasets were part of the training set and the CRF system was able to model these styles more accurately. In fact, the performance on Classical and Jazz music pieces on RWC Genre is improved by 6.8 pp, which is even better than the 3.1 pp overall. It highlights the potential of the data-driven proposed system, where relevant annotated data has a big impact on performance.

5.4 Analysis of the transition features:

The output space being similar with the one defined in [6], we can compare the transition coefficients. Due to space constraints, we limit our analysis to bars of 6 and 8 tatum of the $pl + da$ CRF model. They correspond to the most common bars in the used datasets. The transition coefficients can be seen in figure 3. The first observation is that the general intuition of moving circularly inside a bar is indeed learned by the CRF model as seen with the stronger weights of the transition feature function close to the diagonal of the figure. We also see that the proposed learned CRF model transition coefficients are more detailed while they seem more binary in [6]. The proposed system is less restrictive in metrical changes as can be seen by the coefficient of the output transitions $Y_6^6 \rightarrow Y_1^8$ and $Y_8^8 \rightarrow Y_1^6$ in particular. It can be because the observation features are

[6] It is to note that the comparison between the data augmented system and [6] is fair since the networks were trained on the same data, and the feature combination and temporal model steps of the heuristic model is blind to any data.

I_{HCNN}	I_{RCNN}	I_{MCNN}	I_{BCNN}
11.3	42.1	9.0	37.7

Table 3. Mean impact of each feature representation on the pl + da CRF model.

reliable enough to avoid false metrical changes between 6 and 8 tatum-long bars. Finally, we see that some transitions have strong negative weights in the CRF model. $Y_4^8 \rightarrow Y_1^8$ and $Y_7^8 \rightarrow Y_5^8$ have the top negative weights, both at -3.9. In the first case, it corresponds to going back to the downbeat after 4 tatums and in the second case to going back to the downbeat after 16 tatums while being in a 8 tatum-long bar. Finding the difference between a 4, 8 and 16 tatum-long bar is indeed quite difficult perceptively and for the networks. There can be one part of the song where the chords or the rhythmic patterns change twice as fast or twice as slow, which could misled the observation features. The negative weights can therefore emphasize a metrical continuity in the decoding.

5.5 Ability to find the correct metrical level:

To evaluate the ability of the system to find the correct metrical level, we use the continuity-based metric focusing on the total proportion of correct regions at the correct metrical level (CMLt) with a tolerance window of $\pm 17.5\%$ of the inter-beat-interval [7]. The proposed system obtains a CMLt of 61.5% while [6] obtains a CMLt of 56.6%. The CRF model is therefore more efficient to find the correct metrical level compared to [6]. It can be explained by the fact that every downbeat and non downbeat outputs have a different observation features while all the non downbeat states and all the downbeat states had the same observation feature respectively in the compared system. Besides, as seen above, the transition coefficients of the CRF model are better to avoid octave errors on duple meters while the compared system makes more errors there.

5.6 Analysis of the selected features:

We looked at the weight of the $pl + da$ CRF model to see if a feature representation had more impact than others to detect the downbeat sequence. To do so we calculated the sum of the absolute learned weight value belonging to each feature representation:

$$I_{XCNN} = \sum_{j \in XCNN} |\theta_j| \quad (2)$$

with $X \in \{H, R, M, B\}$. Results are shown in table 3 after a normalization inside and across datasets. It is to note that they are consistent for each dataset and each label. We can see that the rhythmic and bass content networks have a larger impact on the CRF model. It can be surprising knowing that the harmonic network is the best performing network in [6]. However, the rhythmic and bass content networks were trained to recognize the downbeat sequence on the whole input and not a single downbeat per input only. It allows them to encode information about the metrical level that is useful for the CRF model.

[7] We don't consider $\pm 17.5\%$ of the inter-downbeat-interval since it would be too permissive.

6. CONCLUSIONS

We presented a Conditional Random Field system based on multiple deep learned feature representations for the task of downbeat tracking. Using the networks penultimate layer feature representation with 3 beats per bar augmented data, we outperformed 5 compared downbeat tracking algorithms overall. While we need the training and test data to come from similar music styles to make full use of our powerful temporal model, it holds more potential compared to heuristic based approaches and could be more easily adapted to different music styles.

Future work will focus on learning the deep networks and the conditional random field models jointly and on refining the initial temporal segmentation.

7. REFERENCES

[1] J. P. Bello and J. Pickens. A robust mid-level representation for harmonic content in music signals. volume 19, 2005.

[2] S. Böck, F. Krebs, and M. Schedl. Evaluating the online capabilities of onset detection methods. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2012.

[3] M. E. P Davies and M. D. Plumbley. A spectral difference approach to extracting downbeats in musical audio. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, 2006.

[4] S. Durand, J. P. Bello, B. David, and G. Richard. Downbeat tracking with multiple features and deep neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[5] S. Durand, J. P. Bello, B. David, and G. Richard. Feature adapted convolutional neural networks for downbeat tracking. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[6] S. Durand, J. P. Bello, B. David, and G. Richard. Robust downbeat tracking using an ensemble of convolutional networks. *arXiv preprint arXiv:1605.08396*, 2016.

[7] T. Fillon, C. Joder, S. Durand, and S. Essid. A conditional random field system for beat tracking. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[8] B. D. Giorgi, M. Zanoni, A. Sarti, and S. Tubaro. Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony. In *Proceedings*

of the International Workshop on Multidimensional Systems (nDS), 2013.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[10] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical and jazz music databases. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, volume 2, pages 287–288, 2002.

[11] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Music genre database and musical instrument sound database. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, volume 3, pages 229–230, 2003.

[12] S. Hainsworth and M. D. Macleod. Particle filtering applied to musical tempo tracking. *EURASIP Journal on Applied Signal Processing*, 2004:2385–2395, 2004.

[13] A. Holzapfel, F. Krebs, and A. Srinivasamurthy. Tracking the "odd": Meter inference in a culturally diverse music corpus. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 425–430, 2014.

[14] T. Jehan. Downbeat prediction by listening and learning. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 267–270, 2005.

[15] A. Klapuri, A. Eronen, and J. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–355, 2006.

[16] P. Korzeniowski, S. Böck, and G. Widmer. Probabilistic extraction of beat positions from a beat activation function. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2014.

[17] F. Krebs, S. Böck, and G. Widmer. An efficient state-space model for joint tempo and meter tracking. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 72–78, 2015.

[18] F. Krebs, A. Holzapfel, A. T. Cemgil, and G. Widmer. Inferring metrical structure in music using particle filters. *IEEE Transactions on Audio, Speech and Language Processing*, 23(5):817–827, 2015.

[19] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICLM*, 2001.

[20] B. McFee, E.J. Humphrey, and J.P. Bello. A software framework for musical data augmentation. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2015.

[21] N. Okazaki. Crfsuite: a fast implementation of conditional random fields (crfs), 2007.

[22] H. Papadopoulos and G. Peeters. Joint estimation of chords and downbeats from an audio signal. *IEEE Transactions on Audio, Speech and Language Processing*, 19(1):138–152, 2011.

[23] G. Peeters and H. Papadopoulos. Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6), 2011.

[24] J. Peng, L. Bo, and J. Xu. Conditional neural fields. In *Advances in neural information processing systems*, pages 1419–1427, 2009.

[25] A. Srinivasamurthy, A. Holzapfel, and X. Serra. In search of automatic rhythm analysis methods for turkish and indian art music. *Journal of New Music Research*, 43(1):94–114, 2014.

[26] C. Sutton and A. McCallum. Dynamic Conditional Random Fields : Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. In *Proceedings of ICML*, 2004.

[27] C. Sutton and A. McCallum. *An introduction to Conditional Random Fields for relational learning*, chapter 4, pages 93–128. MIT Press, 2006.

[28] D. Temperley and T. d. Clercq. Statistical analysis of harmony and melody in rock music. *Journal of New Music Research*, 42(3):187–204, 2013.

[29] J. Thomassen. Melodic accent: Experiments and a tentative model. *Journal of the Acoustical Society of America*, 71:1596, 1982.

EXPLOITING FREQUENCY, PERIODICITY AND HARMONICITY USING ADVANCED TIME-FREQUENCY CONCENTRATION TECHNIQUES FOR MULTIPITCH ESTIMATION OF CHOIR AND SYMPHONY

Li Su, Tsung-Ying Chuang and Yi-Hsuan Yang
Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan
lisu@citi.sinica.edu.tw, jasonmail04@gmail.com, yang@citi.sinica.edu.tw

ABSTRACT

To advance research on automatic music transcription (AMT), it is important to have labeled datasets with sufficient diversity and complexity that support the creation and evaluation of robust algorithms to deal with issues seen in real-world polyphonic music signals. In this paper, we propose new datasets and investigate signal processing algorithms for multipitch estimation (MPE) in choral and symphony music, which have been seldom considered in AMT research. We observe that MPE in these two types of music is challenging because of not only the high polyphony number, but also the possible imprecision in pitch for notes sung or played by multiple singers or musicians in unison. To improve the robustness of pitch estimation, experiments show that it is beneficial to measure pitch saliency by jointly considering frequency, periodicity and harmonicity information. Moreover, we can improve the localization and stability of pitch by the multi-taper methods and nonlinear time-frequency reassignment techniques such as the Concentration of Time and Frequency (ConceFT) transform. We show that the proposed unsupervised methods to MPE compare favorably with, if not superior to, state-of-the-art supervised methods in various types of music signals from both existing and the newly created datasets.

1. INTRODUCTION

The ability to identify concurrent pitches in polyphonic music is considered admirable by most people. Throughout history, such an ability has been symbolic of a music genius, with the most popular legendary story possibly being Mozart's transcription of Allegri's *Miserere* at the age of fourteen. An interesting question is then whether computers can also possess the ability and perform automatic music transcription (AMT). A great deal of research has been done in the music information retrieval (MIR) community to develop AMT algorithms, but to date it is still an unsolved problem [4].

AMT is challenging for multiple reasons. One such challenge has to do with the creation of labeled multipitch data with diversity, for the labeling process requires considerable expertise and is usually time-consuming [24]. Existing multipitch datasets are often small in size and limited in diversity, and in combination they still cannot represent the rich variety found in music performances. For example, to our knowledge, there is no labeled multipitch data for choir, one of the most common type of music through the ages and cultures and also known as the theme featuring the legendary story of Mozart. As the evaluation of AMT algorithms requires labeled data, the transcription of choir music remains largely unexplored.

The rich variety of music also poses challenges in designing features robust to variations in timbre, genre, and type of performance. For example, it is difficult to design a feature that performs equally well in characterizing the pitch information in both piano and choir music, for they are fairly different — the latter involves a group of people singing in unison but each having her or his own vocal characteristics and control of pitch. This specific issue of possible imprecision in pitch has rarely been dealt with in the literature, possibly due to the scarcity of related labeled data. The shift-invariant Probabilistic Latent Component Analysis (PLCA) algorithm [3] can support non-ideal tuning and frequency so might be able to partially address this issue, but such an evaluation has not been reported before. Moreover, while PLCA is a supervised algorithm that demands the availability of data, we are interested in unsupervised algorithms.

This work attempts to address the aforementioned issues for multipitch estimation (MPE), a sub-task of AMT. Specifically, we propose new datasets and discuss the characteristics and distinct technical issues of MPE for choir and symphony music. Besides, by extending a previous work [23] we propose an unsupervised approach that interprets a pitch event in three dimensions — frequency, periodicity and harmonicity. Moreover, we introduce recent advance in time-frequency (TF) analysis, including the Synchrosqueezing Transform (SST) and the Concentration of Time and Frequency (ConceFT) method, to improve the stablization and localization of pitch information in our feature representation. Result shows that that the proposed unsupervised methods compare favorably with state-of-

© Li Su, Tsung-Ying Chuang and Yi-Hsuan Yang. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Li Su, Tsung-Ying Chuang and Yi-Hsuan Yang. "Exploiting frequency, periodicity and harmonicity using advanced time-frequency concentration techniques for multipitch estimation of choir and symphony", 17th International Society for Music Information Retrieval Conference, 2016.

the-art supervised methods in various types of music. Finally, a simple decision fusion framework also shows the effectiveness of combining multiple MPE methods.

2. PROBLEM DEFINITION

To facilitate our discussion on feature representation, we focus on the feature design for *frame-level* transcription of polyphonic music, namely the MPE sub-task. Other transcription sub-tasks such as note tracking and timbre streaming [7] are not discussed in this paper.

We refer to a multipitch signal as a superposition of multiple "perceptually mono-pitch" signals. This particular type of mono-pitch signal can be produced either by a single performer, with rather well-defined pitch and loudness, or by a group of performers playing instruments or singing in unison. The latter case, often referred to as "chorus" or "ensemble" sounds, has quite different signal-level characteristics from the former. The major difference lies in the small, independent variations in the fundamental frequency (F0), a.k.a. the *voice flutter* phenomenon [25]. For example, early research in choral music showed that the "dispersion of F0" (measured as the bandwidths of partial tones) among three reputable choirs varied typically in the range of 20–30 cents [17]. It is also found that the *pitch scatter* (i.e., the standard deviation of F0 across singers, averaged over the duration of each tone [17, 25]) among choir basses is 10–15 cents [26]. Previous work on the synthesis of chorus/ensemble effect also adjusted pitch scatter parameters in similar ranges [12, 18].

We assume that every sound contributing to the mono-pitch signal of interest is composed of a series of sinusoidal components which are with *nearly* integer multiples of the F0, i.e., every sound has *low inharmonicity*. In this way, the bandwidth of each partial is mainly determined by the amount of the frequency variations of every sound. Besides, we ignore issues of *missing fundamental* or *stacked harmonics* found in real-world polyphonic signals, for they both have been discussed in our previous work [23]. In summary, we assume the signal under analysis has discernable F0s and small but nonzero degree of inharmonicity and pitch scatter.

3. RELATED WORK

3.1 Pitch saliency features

For a signal $x(t)$ with multiple periodicities, a pitch candidate is determined by 1) a *frequency representation* $V(t, f)$ that reveals the saliency of every fundamental frequency and its harmonic frequencies (i.e., its integer multiples) in a signal, 2) a *periodicity representation* $U(t, q)$ that reveals the saliency of every fundamental period and its integer multiples in a signal, [1] and 3) the *constraints on harmonicity* described as follows: at a specific time t_0, a pitch candidate $f_0 = 1/q_0$ is the true pitch when there exists $M_v, M_u \in \mathbb{N}$ such that [23]:

1. A sequence of prominent peaks found at $V(t_0, f_0)$, $V(t_0, 2f_0), \ldots, V(t_0, M_v f_0)$.
2. A sequence of prominent peaks found at $U(t_0, q_0)$, $U(t_0, 2q_0), \ldots, U(t_0, M_u q_0)$.

An MPE algorithm following this approach has been found useful in transcribing a wide variety of music, including complicated music signals like symphony [23]. The frequency representation being used is the short-time Fourier transform (STFT). For a window function $h(t)$, the STFT of $x(t)$ is formulated by

$$V_x^{(h)}(t, f) = \int x(\tau) h(\tau - t) e^{-j2\pi f(\tau - t)} \mathrm{d}\tau \,. \quad (1)$$

For periodicity representations, an important one for MPE is the the *generalized cepstrum* [16, 28]:

$$U_x^{(h,g_\xi)}(t, q) = \int g_\xi(V_x^{(h)}(t, f)) e^{-j2\pi q u} \mathrm{d}u \,, \quad (2)$$

where q is referred to as *lag* or *quefrency*, and $g_\xi(\cdot)$ is a nonlinear scaling function defined by either $g_1(y) = |y|^\gamma$ or $g_2(y) = (|y|^\gamma - 1)/\gamma$, $0 < \gamma \leq 2$. We remark that when $\gamma = 2$, $U_x^{(h,g_1)}$ becomes the ACF according to the Wiener-Khinchin theorem, and when $\gamma \to 0$, $U_x^{(h,g_2)}$ approximates to the *real cepstrum*. $U_x^{(h,g_1)}$ and $U_x^{(h,g_2)}$ are different merely in the scale and the zero-quefrency term, so for simplicity we use $U_x^{(h,g_1)}$ in this paper. We will omit ξ in the notation and simply denote $U_x^{(h,g_\xi)}(t, q)$ as $U_x^{(h)}(t, q)$.

3.2 Multi-taper time-frequency analysis

In conventional STFT, the spectrum is estimated by only one window function. In contrast, multi-taper TF analysis estimates the spectrum by averaging of the spectral estimation of multiple windows (i.e., tapers) [27]. The main purpose of multi-tapering is to stabilize the spectrum estimation by reducing the variance due to noises or boundary of the segments. The tapers are basically orthogonal to each other, and their estimates are approximately uncorrelated. Therefore, the average of them can reduce the variance. To be more specific, given $\mathbf{z} = [\nu_1, \ldots, \nu_J]$, a set of J-taper window with good concentration in the TF plane, where $\nu_i \in \mathbb{R}^T$ is a window of length T for $i = 1, 2, \ldots, J$, and $\mathbf{z}$ forms an orthonormal basis in $\mathbb{R}^T$, the multi-taper STFT is given by $\frac{1}{J} \sum_{j=1}^{J} V_x^{(\nu_j)}(t, f)$, the average of every $V_x^{(\nu_j)}(t, f)$.

Although rarely seen in the literature of MIR, the multi-tapering method has been found useful in several different applications in speech processing such as speaker identification and emotion recognition [1, 14], because of its stable output of feature representation.

3.3 Synchrosqueezing Transform (SST)

SST is a special case of *time-frequency reassignment* [2], a class of nonlinear TF analysis techniques. In a nutshell, it aims at moving the spectral-leakage terms caused by Heisenberg-Gabor uncertainty to the center of mass

[1] The fundamental period of a periodic signal $x(t)$ is defined as the smallest q such that $x(t + q) = x(t)$. Since q is measured in time, we refer to q as in the *lag domain*, to distinguish it from t in the *time domain*.

of true component, and therefore sharpens the harmonic peaks and achieves high localization [5]. SST uses the *frequency reassignment vectors* estimated by the *instantaneous frequency deviation* (IFD). In music processing, such a method can better discriminate closely-located components, and applications have be found in music processing tasks such as chord recognition, synthesis, and melody extraction [11, 13, 20, 22].

Let $V_x^{(h)} = |V_x^{(h)}|e^{\Phi_x^h}$. The IFD, $\Omega_x^{(h,\theta_v)}$, is defined as the time derivative of the instantaneous phase term Φ_x^h:

$$\Omega_x^{(h,\theta_v)}(t,\eta) := \frac{\partial \Phi_x^{h_n}}{\partial t} = -\Im \left. \frac{V_x^{(\mathcal{D}h)}(t,\eta)}{V_x^{(h)}(t,\eta)} \right|_{\mathfrak{N}_v}, \quad (3)$$

where $\Im$ means the imaginary part and $\mathfrak{N}_v := \{f : |V_x^{(h_n)}(t,f)| > \theta_v\}$ gives a threshold so as to avoid computation instability when $|V_x^{(h_n)}(t,f)|$ is very small. This formulation (3) can be derived by definition. The SST is therefore represented as

$$S_x^{(h,\theta_v)}(t,f) = \int_{\mathfrak{N}_v} V_x^{(h)}(t,\eta)\delta\left(|f - \Omega_x^{(h)}(t,\eta)|\right) \mathrm{d}\eta\,. \quad (4)$$

As the result of our analysis is not sensitive to the value of the parameter θ_v, we set θ_v to 10^{-6} of the root mean square energy of the signal under analysis throughout the paper. For convenience, we also omit θ_v in the notation and simply denote $\Omega_x^{(h,\theta_v)}$ and $S_x^{(h,\theta_v)}$ as $\Omega_x^{(h)}$ and $S_x^{(h)}$.

3.4 ConceFT

The main drawback of the TF reassignment is the spurious terms contributed by inaccurate IFD estimation resulting from correlations between noise and the window function. To achieve both localization and stability at the same time, a solution is the *multi-taper SST*, the average of multiple SST computed by a finite set of orthogonal windows [30]. Recently, Daubechies, Wang and Wu improved this idea and proposed the ConceFT method [6]. ConceFT emphasizes the use of *over-complete* windows rather than merely orthogonal windows, by assuming that a spurious term in a specific TF location just appears sparsely for the TF representations using different windows. Theoretical analysis proves that the ConceFT leads to sharper estimates of the instantaneous frequencies for signals that are corrupted by noise. Experiments also showed that ConceFT is useful in estimating the instantaneous frequencies with a fluctuated trajectory [6], a case similar to pitch scatter.

The over-complete window functions for ConceFT is generated from $\mathbf{z}$. This set of window functions $\mathbf{h} = [h_1, \ldots, h_N]$ with N windows is constructed as $h_1(t) = \nu_1(t)$ and $h_n(t) = \sum_{j=1}^{J} r_{nj}\nu_j(t)$ for $j = 2, \ldots, J$, $n = 2, \ldots, N$, and $\mathbf{r}_n = [r_{n1}, \ldots, r_{nJ}]$ is a random vector with unit norm. In ConceFT we need $J > 1$ and $N \geq J$. In contrast, a single window TF analysis requires $J = 1$ and $N = 1$, where $\mathbf{h} = h_1$. ConceFT is represented by

$$C_x^{(\mathbf{h})}(t,f) = \frac{1}{N}\sum_{n=1}^{N} S_x^{(h_n)}(t,f)\,. \quad (5)$$

We refer the reader with interest to [6] for a summary of the current progress in this direction.

For simplicity, we use $J = 2$ in this paper. Specifically, we use the Hamming window $0.54 + 0.46\cos(2\pi t/T)$ for ν_1, and the sine window $\sin(2\pi t/T)$ for ν_2. Obviously, ν_1 is orthogonal to ν_2, and the spectrum of ν_1 is concentrated to zero frequency whereas ν_2 has a zero at $f = 0$.

4. PROPOSED METHOD

4.1 Combining frequency and periodicity

An intuitive way to combine the frequency and periodicity representations is to multiply $V_x^{(h_n)}$ and $U_x^{(h_n)}$, after mapping the latter from time-quefrency into the TF domain:

$$W_x^{(h_n)}(t,f) = |V_x^{(h_n)}(t,f)|U_x^{(h_n)}\left(t, \frac{1}{f}\right). \quad (6)$$

This approach has been mentioned in previous work on single pitch detection [19], where the F0 is determined simply by $f_0(t) = \arg\max_f W_x^{(h_n)}(t,f)$. Please note that here we only consider the co-occurrence of salience in the frequency and periodicity representations, and so far the constraints of harmonicity have not been included. A threshold on either $V_x^{(h_n)}$ or $U_x^{(h_n)}$ for removing unwanted terms is also critical to system performance. We will consider these issues below.

4.2 Constraints on harmonicity

To identify the location of the harmonic components in the STFT, one may use *pseudo-whitening*, a preprocessing step of estimating the spectrum envelope [15]. This method, however, is unreliable for a spectrum whose envelope is not smoothly varying or not supported by a large number of harmonics. This happens to be the case in choral music, since the singers tend to sing with more power in the F0 region rather than in the singer's format region [21].

To address this issue, we propose to assess whether a component at (t,f) is a sinusoidal component by using the IFD, instead of the spectral envelope. The rational is: as small $|\Omega_x^{(h)}|$ implies that the corresponding component in STFT is close to the true component, we can assume that $|\Omega_x^{(h)}|$ is bounded by a positive value θ_s around the harmonic components in the STFT. We have accordingly the *constraint on harmonicity in frequency representation*:

$$\mathfrak{N}_s := \{f : \max\left[|\Omega_x^{(h_n)}(t,(1:M_v)f)|\right] < \theta_s\}\,, \quad (7)$$

where we use $(t,(1:M_v)f)$ as the shorthand for the set of points $\{(t,f),(t,2f),\ldots,(t,M_vf)\}$. That is, for an F0 at (t_0,f_0), we require that $|\Omega_x^{(h)}(t_0,f_0)|$ is smaller than θ_s at (t_0,f_0) and its integer multiples. Similarly, for a fundamental period event at (t_0,q_0), the amplitude of $U_x^{(h_n)}$ should be above a threshold θ_c not only at (t_0,q_0) but also the multiples of its period. This leads to the *constraint on sub-harmonicity in periodicity representation*:

$$\mathfrak{N}_c := \{f : \min\left[U_x^{(h_n)}(t,(1:M_u)q)\right] > \theta_c\}\,. \quad (8)$$

With (3), (7) and (8), we have a more succinct feature representation $Y_x^{(h_n)}(t,f)$ by removing most of the non-harmonic-related terms in $W_x^{(h_n)}(t,f)$:

$$Y_x^{(h_n)}(t,f) = W_x^{(h_n)}(t,f)\Big|_{\mathfrak{N}}, \tag{9}$$

where $\mathfrak{N} := \mathfrak{N}_v \cap \mathfrak{N}_s \cap \mathfrak{N}_c$. Moreover, to enhance localization of this multipitch feature, we consider synchrosqueezing operation on $Y_x^{(h_n)}$ (instead of on $V_x^{(h_n)}$):

$$S_x^{(h_n)}(t,f) := \int_{\mathfrak{N}} Y_x^{(h_n)}(t,\eta)\delta\left(|f-\Omega_x^{(h_n)}(t,\eta)|\right)\mathrm{d}\eta\,, \tag{10}$$

Finally, by modifying (4), the multi-taper or ConceFT-based multipitch feature is obtained from averaging either $Y_x^{(h_n)}$ or $S_x^{(h_n)}$ over $n = 1, 2, \ldots, N$, respectively:

$$B_x^{(\mathbf{h})}(t,f) = \frac{1}{N}\sum_{n=1}^{N} Y_x^{(h_n)}(t,f)\,, \tag{11}$$

$$C_x^{(\mathbf{h})}(t,f) = \frac{1}{N}\sum_{n=1}^{N} S_x^{(h_n)}(t,f)\,. \tag{12}$$

In the experiments we will compare the performance of the four features formulated in (9)–(12).

4.3 Implementation issues

There are several ways to sample the value of $U_x^{(h_n)}(t,1/f)$ from $U_x^{(h_n)}(t,q)$, the simplest way being assigning every components in q to the bin closest to $f = 1/q$. However, the problem is there are usually insufficient low-quefrency points in $U_x^{(h_n)}(t,q)$ to represent the high-frequency part in $U_x^{(h_n)}(t,1/f)$. For example, there are only 34 points in $U_x^{(h_n)}$ to represent frequencies ranging from 1 kHz to 4 kHz for a signal sampled at 44.1 kHz. A simple yet effective solution is to linearly interpolate $U_x^{(h_n)}(t,q)$ into a fine grid with 0.4 Hz spacing,[2] and then have $U_x^{(h_n)}(t,1/f) = \sum_{j\in\mathfrak{P}(f)} U_x^{(h_n)}(t,q_j)$, where $\mathfrak{P}(f) := \{j : 1/(f+0.5/T) < q_j < 1/(f-0.5/T)\}$. A short-pass lifter described in [23] is also applied to $U_x^{(h_n)}(t,q)$.

Another issue of this mapping scheme is that the low-frequency part could be overemphasized since the summation is over a wide quefrency range and thereby cannot reveal true salience of pitch. This is not a critical issue if the dynamic information of each note is not required in transcription, but such dynamic information is needed here for late fusion. To address this, in our implementation we use a *binarized* version of $U_x^{(h_n)}$ to treat it as a *mask* in filtering out unwanted harmonic peaks in $V_x^{(h_n)}$, by setting $U_x^{(h_n)}(t,q) = 1$ if $U_x^{(h_n)}(t,q) > \theta_c$ and 0 otherwise.

5. EXPERIMENT

5.1 Datasets

To provide available source for the research on transcribing music with pitch scatter, we propose two new datasets, directly named Choir and Symphony here, which contain 5 excerpts of choral music from 3- to 8-part, and 5 excerpts of symphony, respectively. The length of the excerpts ranges from 18 to 108 seconds, totaling 5 minutes and 40 seconds. Information of each note events, including onset, offset, pitch name and instrument, are annotated by a professional pianist using the annotation methodology proposed in [24]. The audio, annotation, and other detailed information will be made public through a website. [3]

To test the generalizability of the proposed method, we also experiment on another two commonly used MPE datasets — Bach10 [8] and TRIOS [9]. The former contains ten quartets of four different instruments, while the latter consists of five pieces of fully synthesized music of piano and two other pitched instruments. The sampling rate of all the audio files is 44.1 kHz.

5.2 Numerical illustration

For all the proposed features, we empirically set the window length T to 0.14 second, hop size H to 0.01 second and use $\theta_c = 2\times10^{-4}$. For $Y_x^{(h_1)}$ and $B_x^{(\mathbf{h})}$, we set $N = 1$ and $\theta_s = 3$ bins (21.43 Hz); and for $S_x^{(h_1)}$ and $C_x^{(\mathbf{h})}$, we set $N = 10$ and $\theta_s = 1$ bin (7.14 Hz), a narrower bandwidth to enhance localization. The nonlinear scaling factor of g_ξ in (2) is set to 0.15.

Figure 1 shows in row the ground truth, $V_x^{(h_1)}(t,f)$, $U_x^{(h_1)}(t,1/f)$, $W_x^{(h_1)}(t,f)$ and $C_x^{(\mathbf{h})}(t,f)$ of four 5-second excerpts sampled respectively from the four datasets introduced in Section 5.1. Notice that $U_x^{(h_1)}(t,1/f)$ shown here is after thresholding of (7) to avoid negative values. We apply a power scale $(\cdot)^{0.1}$ in drawing the figures.

The second and third columns show that $V_x^{(h_1)}(t,f)$ or $U_x^{(h_1)}(t,1/f)$ alone is not a good multipitch feature since the former suffers from unwanted harmonic terms and the latter from "sub-harmonic" ones. However, most of these terms are removed in $W_x^{(h_1)}$, as seen in the fourth column. Furthermore, in the rightmost column, $C_x^{(\mathbf{h})}$ achieves very sharp components with few noises; it also nicely localizes the pitches. Notably, we see that the STFT components in Choir and Symphony spread widely and are much more fluctuated than those in Bach10 and TRIOS (due to severe pitch scatter), but the sharpness of the components in $C_x^{(\mathbf{h})}$ of the four samples are almost the same.

It can also be seen that there are still some challenging cases in the symphony due to its high complexity. For example, the short pitch activation above 1kHz in the ground truth (*pizzicato* of the 1st violin) remains unrecalled even in $C_x^{(\mathbf{h})}$. This is a subject of future work.

5.3 Piano roll output and post-processing

To obtain the piano roll output, all the features are processed first by a moving median filter with length 0.21 second to enhance smoothness, and then by a peak peaking process to pertain local maxima and discard other non-peak terms. Then, the MPE result, represented in piano

[2] Pilot studies show that finer grid spacing results in smoother feature representation but provides no significant empirical gains in MPE.

[3] https://sites.google.com/site/lisupage/research/new-methodology-of-building-polyphonic-datasets-for-amt

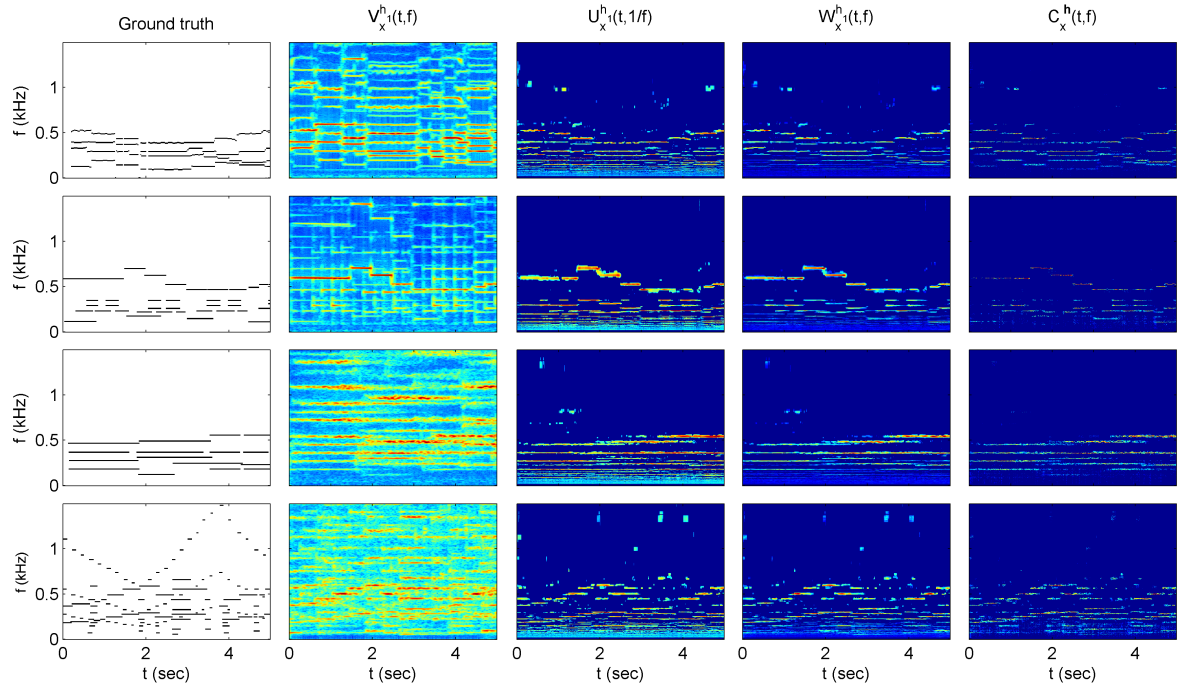


Figure 1. Illustration of the ground truth, $V_x^{(h_1)}(t,f)$, $U_x^{(h_1)}(t,1/f)$, $W_x^{(h_1)}(t,f)$ and $C_x^{(\mathbf{h})}(t,f)$ of four 5-second excerpts. $V_x^{(h_1)}$: STFT; $U_x^{(h_1)}$: generalized cepstrum; $W_x^{(h_1)}$: combination of STFT and generalized cepstrum; $C_x^{(\mathbf{h})}$: the proposed representation with ConceFT. First row: '01-AchGottundHerr.wav' (i.e. Bach's *Ach Gott und Herr, wie gros und schwer*, BWV 255) in quartet (violin, clarinet, saxophone and bassoon). Second row: Mozart's Trio in Eb major 'Kegelstatt', K.498, in piano, clarinet and viola. Third row: William Byrd, *Ave Verum Corpus*, in SATB choir. Fourth row: Tchaikovsky, Symphony No.6, Op.74 (*Pathetique*), Mov.2., in flute, oboe, clarinet, bassoon, horn, trumpet and strings.

roll $O(t,p)$, where $p = 13, 14, \ldots, 76$ is the piano roll number from `A1` (55 Hz) to `C7` (2,093 Hz), is obtained by $O(t,p) = \sum_{\mathfrak{F}(p)} X(t,f)$, where $\mathfrak{F}(p) = \{f : 440 \times 2^{(p-49-0.5)/12} \leq f < 440 \times 2^{(p-49+0.5)/12}\}$ (notice that A4 is the 49th key on the piano), and X denotes one of the feature representations described in (9)–(12).

The results of the proposed and baseline algorithms are refined by the same post-processing steps. The first step removes isolated pitches that are above `C5` and leave any other pitches in the affinity of 0.1 second by more than an octave. This is done because composers usually prefer smaller intervals within one octave in the high-pitch range. The second step is again a moving median filter with length also 0.21 second for smoothness.

5.4 Baselines and evaluation

Three baseline methods are considered. The first baseline is the *unsupervised* method proposed in our previous work [23], which also combines information of frequency, periodicity and harmonicity, but its harmonicity constraint was performed on the piano roll representation rather than directly on the TF representation. We did not use advanced TF analysis such as SST and ConceFT in the prior work [23]. Moreover, the method of computing the adaptive threshold of spectral representation is also different. We use the parameters suggested in [23], and the nonlinear scaling factor for generalized cepstrum is also set to 0.15.

The second one is an *unsupervised* method based on the Constrained Non-negative Matrix Factorization (C-NMF) algorithm proposed by Vincent *et al.* [29].[4] For the experiment on all datasets, we set $\beta = 0.5$ for computing the β-divergence and the value of $\vartheta = -32$ dB for thresholding the activation patterns in C-NMF.

The third baseline is a *supervised* method based on the shift-invariant PLCA proposed by Benetos *et al.* [3].[5] This approach uses labeled data to learn five templates for each pitch of each instrument and voice. The templates are learned from the single notes of the RWC instrument dataset [10], which contains various instruments as well as five vowels of human voice including soprano, alto, tenor and bass. We set the parameter for instrument activation $s_z = 1.3$, the parameter for source contribution $s_u = 1.1$ and the parameter for pitch shifting $s_h = 1.1$, all similar to [3]. To facilitate the comparison between the supervised and unsupervised approaches to MPE, we employ an *instrument-informed* setting that uses templates learned from different instruments for different music pieces, which might have given PLCA some advantages.

Moreover, to investigate cross-model behaviors of the algorithms, we experiment with a *late fusion* scheme that combines our method with PLCA. We first normalized

[4] `http://www.irisa.fr/metiss/members/evincent/multipitch_estimation.m`

[5] `https://code.soundsoftware.ac.uk/projects/amt_mssiplca_fast`

Table 1. Experiment result. $Y_x^{(h_1)}$, $B_x^{(\mathbf{h})}$, $S_x^{(h_1)}$ and $C_x^{(\mathbf{h})}$ are described in (9), (10), (11), (12), respectively. $Y_x^{(h_1)}$: single-window, without synchrosqueezing; $B_x^{(\mathbf{h})}$: single-window, with synchrosqueezing; $S_x^{(h_1)}$: overcomplete-window, without synchrosqueezing; $C_x^{(\mathbf{h})}$: overcomplete-window, with synchrosqueezing

Dataset	Proposed				Baseline			Proposed+PLCA (Late fusion)			
	$Y_x^{(h_1)}$	$B_x^{(\mathbf{h})}$	$S_x^{(h_1)}$	$C_x^{(\mathbf{h})}$	[23]	C-NMF	PLCA	$Y_x^{(h_1)}$	$B_x^{(\mathbf{h})}$	$S_x^{(h_1)}$	$C_x^{(\mathbf{h})}$
Bach10	**83.96**	83.29	79.18	82.13	81.97	79.78	70.57	82.39	82.14	**82.69**	82.04
TRIOS	**66.30**	**66.30**	60.23	66.26	64.09	59.40	64.93	**71.10**	70.79	70.35	70.57
Choir	57.44	59.71	51.29	**61.18**	44.98	45.62	61.07	64.36	64.88	64.06	**65.31**
Symphony	49.14	**50.44**	46.95	50.33	48.82	40.34	47.04	51.73	**52.46**	51.02	51.86

every frame of the piano roll output by its l_2 norm, then combine them through linear superposition, and finally discard the terms which are smaller than a threshold ϵ:

$$\bar{O}_{fusion} = (\alpha \bar{O}_{PLCA} + (1-\alpha)\bar{O}_{proposed} - \epsilon)_+ , \quad (13)$$

where $\bar{O}$ is the normalized piano roll output, $\alpha \in [0, 1]$ controls the relative weights of the two methods, and $(x)_+ = \max(0, x)$ is a hard thresholding function.

We evaluate the accuracy of MPE using the micro-average frame-level F-score, which counts the number of true positives, false positives and false negatives over all the frames within a dataset and then calculates the harmonic mean of the precision and recall rates.

6. RESULT

Table 1 lists the F-scores on the four datasets using the proposed methods using features (9)–(12), three baseline methods and late fusion of proposed features with PLCA. The main findings are reported below.

First, the four proposed methods outperform the three baselines in general. Although the method [23] adopted the same approach of combining frequency and periodicity information as the proposed methods do, it was reported to be sensitive to γ, the nonlinear scaling factor in computing the generalized cepstrum. Besides, the method [23] cannot benefit from the constraint on harmonics, especially in the case of Choir, as the estimation of the spectral threshold and noise terms is rather inaccurate without IFD information. C-NMF also performs poorly for such challenging musical signals. In comparison to the two unsupervised baselines, PLCA performs fairly well in Choir and Symphony, perhaps because it uses supervised templates and allows template shifting in pitch [3].

Second, among the proposed methods, we find the multi-taper ones $B_x^{(\mathbf{h})}$ and $C_x^{(\mathbf{h})}$ do perform better than those use only one window, i.e., $Y_x^{(h_1)}$ and $S_x^{(h_1)}$, for datasets with pitch scatter (i.e., Choir and Symphony). However, for Bach10 and TRIOS, where most of the pitches are played with only one instrument, multi-tapering and SST do not give better performance, as there is no need to reduce the variance of a spectral peak of a single source.

Moreover, the method using single-window SST $S_x^{(h_1)}$ performs the worst among the proposed methods, as its nonlinearity usually gives rise to unwanted speckle terms, a major known drawback of SST [6]. This problem is nicely solved by ConceFT, as we can see that $C_x^{(\mathbf{h})}$ outperforms $S_x^{(h_1)}$ by around 10% in Choir and 3% in Symphony. This suggests the need to introduce multi-tapering to stabilize the estimation, when feature localization is an important requirement for the system.

Finally, a grid search over the four datasets shows that the optimal result of late fusion is achieved by setting $\alpha = 0.05$ (i.e. emphasizing the proposed method) and $\epsilon = 3 \times 10^{-5}$. Combining PLCA and $C_x^{(\mathbf{h})}$ achieves 65.31% for Choir, which amounts to more than 4% improvement over $C_x^{(\mathbf{h})}$. Combining PLCA and $Y_x^{(\mathbf{h})}$ further improves the F-score by 4.8%. However, less improvement is found in Bach10 and Symphony, possibly because that the former already has limited space for improvement and the latter is rather complicated such that some information cannot be well captured by both method. Although the weighting on PLCA is small, PLCA does capture some critical information missed by the proposed methods. This suggests the importance of fusing different MPE models, in particular unsupervised and supervised ones.

7. CONCLUSION

To improve the robustness of MPE algorithms in dealing with diverse music signals, we introduce and incorporate novel TF analysis tools including SST and ConceFT to enhance the stability and localization of multipitch features. The proposed unsupervised methods also measure pitch saliency by jointing considering frequency, periodicity and harmonicity. Result on two newly created datasets of choral and symphony music demonstrates the superiority of the proposed methods for MPE in music signals featuring pitch scatter. Slightly better result can be obtained by combining our methods and the supervised method PLCA.

8. ACKNOWLEDGEMENT

We thank Patricia Hsu for data labeling and Dr. Hau-Tieng Wu for fruitful discussions. Dr. Li Su was supported by the postdoctoral fellowship of Academia Sinica.

9. REFERENCES

[1] Y. Attabi, M. J. Alam, P. Dumouchel, P. Kenny, and D. O'Shaughnessy. Multiple windowed spectral features for emotion recognition. In *Proc. ICASSP*, pages 7527–7531, 2013.

[2] F. Auger and P. Flandrin. Improving the readability of time-frequency and time-scale representations by the reassignment method. *IEEE Trans. Signal Process.*, 43(5):1068 –1089, may 1995.

[3] E. Benetos and S. Dixon. A shift-invariant latent variable model for automatic music transcription. *Computer Music Journal*, 36(4):81–94, 2012.

[4] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: challenges and future directions. *J. Intelligent Information Systems*, 41(3):407–434, 2013.

[5] I. Daubechies, J. Lu, and H.-T. Wu. Synchrosqueezed wavelet transforms: An empirical mode decomposition-like tool. *Appl. Numer. Harmon. Anal.*, 30:243–261, 2011.

[6] I. Daubechies, Y. Wang, and H.-T. Wu. Conceft: Concentration of frequency and time via a multitapered synchrosqueezed transform. *Philosophical Transactions A*, 374(2065), 2016.

[7] Z. Duan, J. Han, and B. Pardo. Multi-pitch streaming of harmonic sound mixtures. *IEEE/ACM Trans. Audio, Speech, Language Process.*, 22(1):138–150, 2014.

[8] Z. Duan, B. Pardo, and C. Zhang. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE/ACM Trans. Audio, Speech, Language Process.*, 18(8):2121–2133, 2010.

[9] J. Fritsch. High quality musical audio source separation. Master's thesis, Queen Mary Centre for Digital Music, 2012.

[10] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Music genre database and musical instrument sound database. In *Proc. ISMIR*, pages 229–230, 2003.

[11] S. W. Hainsworth. *Techniques for the Automated Analysis of Musical Audio*. PhD thesis, University of Cambridge, UK, 2003.

[12] D. Kahlin and S. Ternström. The chorus effect revisited-experiments in frequency-domain analysis and simulation of ensemble sounds. In *Proc. EUROMICRO Conference*, volume 2, pages 75–80, 1999.

[13] M. Khadkevich and M. Omologo. Time-frequency reassigned features for automatic chord recognition. In *Proc. ICASSP*, pages 181–184, 2011.

[14] T. Kinnunen, R. Saeidi, F. Sedlák, K. A. Lee, J. Sandberg, M. Hansson-Sandsten, and H. Li. Low-variance multitaper mfcc features: a case study in robust speaker verification. *IEEE Trans. Audio, Speech, Language Proc.*, 20(7):1990–2001, 2012.

[15] Anssi P. Klapuri. Multiple fundamental frequency estimation based on harmonicity and spectral smoothness. *IEEE Speech Audio Process.*, 11(6):804–816, 2003.

[16] T. Kobayashi and S. Imai. Spectral analysis using generalized cepstrum. *IEEE Trans. Acoust., Speech, Signal Process*, 32(5):1087–1089, 1984.

[17] W. Lottekmoser and Fr. J. Meyer. Frequenzmessungen an gesungenen akkorden. *Acta Acustica united with Acustica*, 10(3):181–184, 1960.

[18] J. Pätynen, S. Tervo, and T. Lokki. Simulation of the violin section sound based on the analysis of orchestra performance. In *Proc. WASPAA*, pages 173–176, 2011.

[19] G. Peeters. Music pitch representation by periodicity measures based on combined temporal and spectral representations. In *Proc. ICASSP*, 2006.

[20] G. Peeters and X. Rodet. Sinola: A new analysis/synthesis method using spectrum peak shape distortion, phase and reassigned spectrum. In *Proc. ICMC*, 1999.

[21] T. D. Rossing, J. Sundberg, and S. Ternström. Acoustic comparison of voice use in solo and choir singing. *J. Acoust. Soc. Am.*, 79(6):1975–1981, 1986.

[22] J. Salamon and E. Gòmez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Trans. Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[23] L. Su and Y.-H. Yang. Combining spectral and temporal representations for multipitch estimation of polyphonic music. *IEEE/ACM Trans. Audio, Speech, Language Process.*, 23(10):1600–1612, 2015.

[24] L. Su and Y.-H. Yang. Escaping from the abyss of manual annotation: New methodology of building polyphonic datasets for automatic music transcription. In *Int. Symposium on Computer Music Multidisciplinary Research*, 2015.

[25] S. Ternström. Perceptual evaluations of voice scatter in unison choir sounds. *Journal of Voice*, 7(2):129–135, 1993.

[26] S. Ternström and J. Sundberg. Intonation precision of choir singers. *J. Acoust. Soc. Am.*, 84(1):59–69, 1988.

[27] D. J. Thomson. Spectrum estimation and harmonic analysis. *Proc. IEEE*, 70:1055–1096, 1982.

[28] T. Tolonen and M. Karjalainen. A computationally efficient multipitch analysis model. *IEEE Speech Audio Process.*, 8(6):708–716, 2000.

[29] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Trans. Audio, Speech, Language Process.*, 18(3):528–537, 2010.

[30] J. Xiao and P. Flandrin. Multitaper time-frequency reassignment for nonstationary spectrum estimation and chirp enhancement. *IEEE Trans. Signal Process.*, 55:2851–2860, 2007.

GENRE ONTOLOGY LEARNING: COMPARING CURATED WITH CROWD-SOURCED ONTOLOGIES

Hendrik Schreiber
tagtraum industries incorporated
hs@tagtraum.com

ABSTRACT

The Semantic Web has made it possible to automatically find meaningful connections between musical pieces which can be used to infer their degree of similarity. Similarity in turn, can be used by recommender systems driving music discovery or playlist generation. One useful facet of knowledge for this purpose are fine-grained genres and their inter-relationships.

In this paper we present a method for learning genre ontologies from crowd-sourced genre labels, exploiting genre co-occurrence rates. Using both lexical and conceptual similarity measures, we show that the quality of such learned ontologies is comparable with manually created ones. In the process, we document properties of current reference genre ontologies, in particular a high degree of disconnectivity. Further, motivated by shortcomings of the established taxonomic precision measure, we define a novel measure for highly disconnected ontologies.

1. INTRODUCTION

In the 15 years since Tim Berners-Lee's article about the *Semantic Web* [2], the *Linking Open Data Community Project* [1] has successfully connected hundreds of datasets, creating a universe of structured data with DBpedia [2] at its center [1, 3]. In this universe, the de facto standard for describing music, artists, the production workflow etc. is *The Music Ontology* [15]. Examples for datasets using it are MusicBrainz/LinkedBrainz [3, 4] and DBTune [5]. While in practice taking advantage of *Linked Open Data* (LOD) is not always easy [9], semantic data has been used successfully, e.g. to build recommender systems. Passant et al. outlined how to use LOD to recommend musical content [14]. An implementation of this concept can be found in [13]. Tatlı et al. created a context-based music recommendation system, using genre and instrumentation information from DBpedia [18]. Di Noia et al. proposed a movie recommender based on LOD from DBpedia, Freebase [6], and LinkedMDB [7] [8]. And recently, Oramas et al. created a system for judging artist similarity based on biographies linked to entities in LOD-space [11]. Many of these approaches are trying to solve problems found in recommender systems relying on collaborative filtering, like cold start or popularity bias [4].

Among other data, genre ontologies are a basis for these systems. They allow the determination of degree of similarity for musical pieces (e.g. via the length of the shortest connecting path in the ontology graph), even if we have no other information available. Surprisingly, we know little about the genre ontologies contained in repositories like DBpedia. How large and deep are they? How well do they represent genre knowledge? Are they culturally biased? How interconnected are genres in these ontologies?

While editors of LOD ontologies often follow established rules, it is an inherent property of any ontology that its quality is subjective. An alternative are learned ontologies. Naturally, they do not represent objective truth either, but instead of relying on design principles, they use empirical data. An interesting question is: How do curated genre ontologies compare with learned ontologies?

In the following we are attempting to answer some of these questions. Section 2 starts with proposing a method for building a genre ontology from user-submitted genre tags. In Section 3, we describe the existing genre ontologies DBpedia and WikiData as well as two new ontologies created with the method from Section 2. In Section 4, we describe evaluation measures loaned from the field of ontology learning. Our results are discussed in Section 5, and our conclusions are presented in Section 6.

2. BUILDING THE GENRE GRAPH

As shown in [16], it is possible to create genre taxonomy trees from user-submitted genre labels. These trees have been proven useful for inferring a single top-level genre for a given sub-genre. Unfortunately, taxonomy trees are insufficient when attempting to model the complex inter-genre relations found in the real world. The concept of a fusion-genre for example, i.e. a connection between two

[1] http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData
[2] http://wiki.dbpedia.org/
[3] http://musicbrainz.org/
[4] http://linkedbrainz.org/
[5] http://dbtune.org/

[6] http://www.freebase.com/ — to be shut-down soon.
[7] http://www.linkedmdb.org/

© Hendrik Schreiber. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Hendrik Schreiber. "Genre Ontology Learning: Comparing curated with crowd-sourced ontologies", 17th International Society for Music Information Retrieval Conference, 2016.

otherwise separate taxonomy trees, is impossible to represent. Therefore, an ontology is commonly regarded as the preferred structure to model genres and their relations.

Similar to [5], we define a genre ontology as a structure $\mathcal{O} = (\mathcal{C}, root, \leq_{\mathcal{C}})$ consisting of a set of concepts $\mathcal{C}$, a designated $root$ concept and the partial order $\leq_{\mathcal{C}}$ on $\mathcal{C} \cup \{root\}$. This partial order is called concept hierarchy. The equation $\forall c \in \mathcal{C} : c \leq_{\mathcal{C}} root$ holds for this concept hierarchy. For the sake of simplicity, we treat the relation between genre names and genre concepts as a bijection, i.e. we assume that each genre name corresponds to exactly one genre concept and vice versa.

To construct a genre ontology based on suitably normalized labels, we first create a genre co-occurrence matrix M as described in [16]. The set $\mathcal{C} = \{c_1, c_2, ..., c_n\}$ contains n genres. Each user submission is represented by a sparse vector $u \in \mathbb{N}^n$ with

$$u_i = \begin{cases} 1, & \text{if } c_i = \text{user-submitted genre} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Each song is represented by a vector $s \in \mathbb{R}^n$. Each s is defined as the arithmetic mean of all user submissions u associated with a given song. Thus s_i describes the relative strength of genre c_i. Co-occurrence rates for a given genre c_i with all other genres can be computed by element-wise averaging all s for which $s_i \neq 0$ is true:

$$M_i = \bar{s}, \quad \forall s \text{ with } s_i \neq 0; M \in \mathbb{R}^{n \times n} \quad (2)$$

Unlike [16], we normalize the co-occurrence rates from M so that the maximum co-occurrence rate of one genre with another is 1. This normalized co-occurrence matrix is called N. Just like M, N is asymmetric. For example, `alternative` strongly co-occurs with `rock`, but `rock` co-occurs not as strongly with `alternative`. We take advantage of this by defining a rule that helps us find sub-genres: If a genre c_i co-occurs with another genre c_j more than a minimum threshold τ, c_j co-occurs with c_i more than a minimum threshold υ, and c_i co-occurs with c_j more than the other way around, then we assume that c_i is a sub-genre of c_j. More formally:

$$\begin{aligned} &\forall c_i, c_j \in \mathcal{C} : c_i <_{\mathcal{C}} c_j \text{ iff} \\ &c_i \neq c_j \wedge N_{i,j} > \tau \wedge N_{j,i} > \upsilon \wedge N_{i,j} > N_{j,i} \end{aligned} \quad (3)$$

Note, that this rule allows one genre to be the sub-genre of multiple other genres. τ controls the co-occurrence rate it takes to be recognized as sub-genre. A low τ leads to more sub-genres and fewer top-level genres. υ ensures that the relationship is not entirely one-sided. As an extreme example, a negative υ would require no co-occurrence of genre c_j with c_i, but c_i could still be a sub-genre of c_j.

Applying (3) makes it easy to find top-level genres, but the resulting hierarchy is rather flat. If a genre is more than one node away from $root$, the rule does not perform well, when it comes to deciding whether a genre is either a sub-genre or a sibling. The reason lies in the fixed parameters τ and υ, which are suitably chosen to find top-level genres, but not sub-genres two or more levels deep. To better determine deep sub-genre relationships starting from a given top-level genre, we apply (3) recursively on each hierarchical sub-structure. So if $\mathcal{C}' \subset \mathcal{C}$ is the set of sub-genres for a $c_k \in \mathcal{C}$, then the co-occurrence matrix N' for $\mathcal{C}'$ can be computed just like N. Because N' is normalized, the same τ and υ are suitable to find $\mathcal{C}'$'s top-level genres, i.e. c_k's direct children. Recursion stops, when the sub-structure consists of at most one genre.

3. ONTOLOGIES

In order to evaluate learned ontologies, we need at least one ontology that serves as reference. This is different from a ground truth, as it is well known that a single truth does not exist for ontologies: Different people create different ontologies, when asked to model the same domain [6, 10, 12, 17]. We chose DBpedia and WikiData as references, which are described in Sections 3.1 and 3.2. Using the definitions and rules from Section 2, we constructed two ontologies. One based on submissions by English speaking users and another based on submissions by international users. They are described in Sections 3.3 and 3.4.

3.1 DBpedia Genre Ontology

DBpedia is the suggested genre extension for *The Music Ontology* and therefore a natural choice for a reference ontology. [8] The part of DBpedia related to musical genres is created by extracting Wikipedia's genre infoboxes. For this to succeed, the DBpedia creation process requires that such infoboxes exist, and that there is a defined mapping from localized infobox to ontology properties. Informally we found that for the English edition of Wikipedia both conditions are usually met. This is not always true for other language editions, e.g. German.

Wikipedia's guidelines [9] define three possible hierarchical relations between genres:

- *Sub-genre*: `heavy metal` < `thrash metal`, `black metal`, `death metal`, etc.
- *Fusion*: `industrial` < `industrial metal` ∧ `heavy metal` < `industrial metal`.
- *Derivative*: `post punk` < `house`, `alternative rock`, `dark wave`, etc.

The *derivative* relation differs from *sub-genre* and *fusion* in that derivative genres are considered "separate or developed enough musicologically to be considered parent/root genres in their own right". As the relation does not fit the general concept of sub-genre or sub-class, we excluded it when building the ontology. Further, we were unable to find a formal definition for the DBpedia relation *stylistic origin*. Based on sample data we interpreted it as the inverse of *derivative*. As such it was also excluded. While this made sense for most genres, it did not for some. The `hip hop` infobox for example, lists `East`

[8] As source for this work, we used DBpedia Live, `http://live.dbpedia.org`.

[9] `https://en.wikipedia.org/wiki/Wikipedia:WikiProject_Music/Music_genres_task_force/Guidelines#Genrebox`

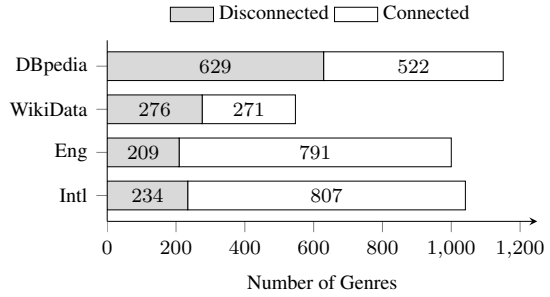


Figure 1. Connected vs. disconnected genres in the four used ontologies. Parameters for generated ontologies: $\tau = 0.17$, $\upsilon = 0.0001$, $|\mathcal{C}_{\text{Eng}}| = 1000$, $|\mathcal{C}_{\text{Intl}}| = 1041$.

`Coast hip hop` and `West Coast hip hop` as regional scenes, but not as sub-genres or derivatives. Unfortunately, in DBpedia, *regional scene* is not defined as a special genre relation, like sub-genre, but just as a plain property. In contrast, both Wikipedia articles on `East Coast hip hop` and `West Coast hip hop` start with assuring a sub-genre relationship to `hip hop`. Also, both DBpedia entries list `hip hop` as the stylistic origin. We found similar issues with `techno` and `Detroit techno`, and other genres.

At the time of writing, the DBpedia-based ontology, created as described above, consisted of 1151 genres with a maximum hierarchy depth of 6. 629 genres (54.6%) did not have any super- or sub-genres (Figure 1). We will refer to it as $\mathcal{O}_{\text{DBpedia}}$. In order to increase the chances of finding corresponding genres in other ontologies, we normalized the raw genre names as well as their aliases found via DBpedia *wikiPageRedirects* (Wikipedia entries for conceptually identical terms).

Loaning from graph theory, we call genres without super- or sub-genres *disconnected*. Ontologies consisting exclusively of disconnected genres we call *trivial*.

3.2 WikiData Genre Ontology

Unlike DBpedia, WikiData is not a parsed version of Wikipedia, but an independent database of structured data for anyone to edit. Currently, WikiData defines just one relation between musical genres: *sub-class*.

In an informal evaluation, we found that, with regard to genres, WikiData is still evolving. While `East Coast hip hop` for example is listed as a sub-genre of `hip hop`, `West Coast hip hop` had no parent at the time of writing. Another example is `techno` and `Detroit techno`. `Detroit techno` existed as an entity, but was not of type music genre, and `techno` was not connected to it in any way. On the plus side, translations of genre names are easily accessible via localized labels for each genre. For matching we used normalized versions of these labels.

At the time of writing, the WikiData-based genre ontology consisted of 547 genres, 276 (50.5%) genres were disconnected, and the hierarchy-depth was 5. We will refer to this ontology as $\mathcal{O}_{\text{WikiData}}$.

3.3 English Language Ontology

Using the rules defined in Section 2, we constructed an ontology based on the top n genre labels submitted by users to the central database of beaTunes [10], a consumer music application [16]. Given the relevance of English in Western pop-culture and the fact that our reference $\mathcal{O}_{\text{DBpedia}}$ offers data based on the English edition of Wikipedia, we only considered submissions by users with English as their system language. We will refer to this ontology as $\mathcal{O}_{\text{Eng}}$. Naturally, $\mathcal{O}_{\text{Eng}}$ is strongly biased towards English culture and contains English genre names almost exclusively. Also, as it is generated from user submitted labels, it contains noise.

Using $\tau = 0.17$ and $\upsilon = 0.0001$ for the top 1000 English genres, we found 209 (20.9%) disconnected genres and the maximum hierarchy-depth was 4.

Because we mentioned `hip hop` and `techno` as problematic examples before, here is what we found for $\mathcal{O}_{\text{Eng}}$: While neither `East Coast hip hop` nor `West Coast hip hop` occur in the top 1000 English genres, `East Coast rap` and `West Coast rap` do. They both have `rap` as a parent, which in turn is a child of `hip hop`. `Techno` does occur as genre, but `Detroit techno` is not in the top 1000 (rank 1557). When using the top 1600 genres as source, `Detroit techno` has `techno` and `electronica` as parents.

3.4 International Ontology

In addition to $\mathcal{O}_{\text{Eng}}$, we generated an international ontology named $\mathcal{O}_{\text{Intl}}$ based on submissions by users with the system languages French, German, Spanish, Dutch, or English. These are the five languages with the most submissions in the beaTunes database. The ontology was created with the goal of being less anglocentric.

Because, the database contains different numbers of submissions per language, we normalized each submission's weight on a per language basis to ensure equal influence. To represent the chosen languages in the selection of the most used genres, we used the intersection of the top n language-specific genres. For $n = 400$ this resulted in a set of 1041 genres, 534 of which also occur in the English top 1000 genres. The sub-set of non-English genre names mostly consists of genuinely new additions like `Kölsch` and `Deutsch Pop`, and translations like `Kindermuziek` and `psychedelische Rock`. The situation regarding `hip hop` and `techno` is similar to $\mathcal{O}_{\text{Eng}}$. Using $\tau = 0.17$ and $\upsilon = 0.0001$ we found that 234 (22.5%) genres were disconnected and the maximum hierarchy-depth was 5.

4. EVALUATION MEASURES

Ontologies can be compared on different levels. In the following, we are going to concentrate on lexical (Section 4.1) and conceptual (Section 4.2) aspects. For both viewpoints measures have been established in the ontology learning community (see e.g. [7, 19]).

[10] `http://www.beatunes.com/`

4.1 Lexical Measures

Let $\mathcal{O}_{\mathrm{R}}$ denote a reference ontology, and $\mathcal{O}_{\mathrm{C}}$ an ontology we wish to evaluate. Correspondingly, $\mathcal{C}_{\mathrm{R}}$ is the set of concepts contained in $\mathcal{O}_{\mathrm{R}}$, and $\mathcal{C}_{\mathrm{C}}$ the concepts in $\mathcal{O}_{\mathrm{C}}$. As we assume a bijective relation between lexical terms and concepts, *lexical precision* (LP) is defined as the ratio between the number of concepts in both ontologies and the number of concepts in $\mathcal{O}_{\mathrm{C}}$:

$$LP(\mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) = \frac{|\mathcal{C}_{\mathrm{C}} \cap \mathcal{C}_{\mathrm{R}}|}{|\mathcal{C}_{\mathrm{C}}|} \tag{4}$$

Lexical recall (LR) is defined as the ratio between the number of concepts in both ontologies and the number of concepts in $\mathcal{O}_{\mathrm{R}}$ [5]:

$$LR(\mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) = \frac{|\mathcal{C}_{\mathrm{C}} \cap \mathcal{C}_{\mathrm{R}}|}{|\mathcal{C}_{\mathrm{R}}|} \tag{5}$$

Finally, the *lexical F-measure* (LF) is defined by:

$$LF(\mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) = \frac{2 \cdot LP \cdot LR}{LP + LR} \tag{6}$$

4.2 Conceptual Measures

The similarity of two concepts $c_i \in \mathcal{C}_{\mathrm{C}}$ and $c_j \in \mathcal{C}_{\mathrm{R}}$ can be measured by comparing their *semantic cotopies* [10]. A basic semantic cotopy is defined as the set containing all super- and sub-concepts for a given concept including itself. The *common semantic cotopy* (*csc*) is similar, but only takes concepts into account that are members of both ontologies we wish to compare. Additionally, the concept for which we are building the cotopy is excluded ($<_{\mathcal{C}}$ instead of $\leq_{\mathcal{C}}$). Both modifications are intended to minimize the influence of lexical similarity [5]:

$$\begin{aligned}&\mathrm{csc}(c_i, \mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) \\ &\quad = \{c_j \in \mathcal{C}_{\mathrm{C}} \cap \mathcal{C}_{\mathrm{R}} | c_j <_{\mathcal{C}_{\mathrm{C}}} c_i \vee c_i <_{\mathcal{C}_{\mathrm{C}}} c_j\}\end{aligned} \tag{7}$$

The *local taxonomic precision* (tp_{csc}) is defined as the ratio between the size of the intersection of the cotopies for two concepts, and the size of the cotopy of just the concept to evaluate:

$$\begin{aligned}&tp_{\mathrm{csc}}(c_i, c_j, \mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) \\ &\quad = \frac{|\mathrm{csc}(c_i, \mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) \cap \mathrm{csc}(c_j, \mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}})|}{|\mathrm{csc}(c_i, \mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}})|}\end{aligned} \tag{8}$$

tp_{csc} is undefined for $|\mathrm{csc}(c_i, \mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}})| = 0$ (division by zero). In the spirit of [5], i.e. to avoid unjustifiably high values for trivial ontologies, we define $tp_{\mathrm{csc}} = 0$ for this case. Based on the local tp_{csc}, we define a *global taxonomic precision* (TP_{csc}) as the mean tp_{csc} for all concepts in $\mathcal{C}_{\mathrm{C}} \cap \mathcal{C}_{\mathrm{R}}$ [7]:

$$\begin{aligned}&TP_{\mathrm{csc}}(\mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) \\ &\quad = \frac{1}{|\mathcal{C}_{\mathrm{C}} \cap \mathcal{C}_{\mathrm{R}}|} \sum_{c \in \mathcal{C}_{\mathrm{C}} \cap \mathcal{C}_{\mathrm{R}}} tp_{\mathrm{csc}}(c, c, \mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}})\end{aligned} \tag{9}$$

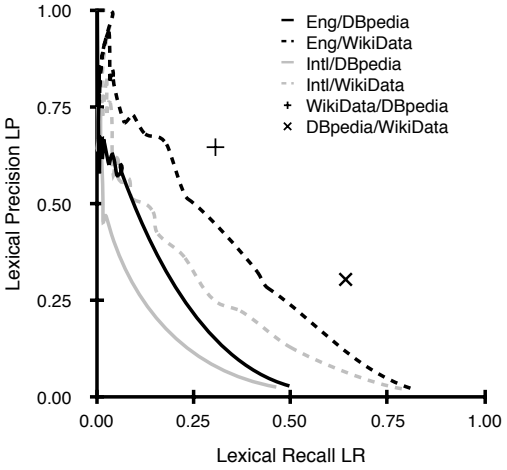


Figure 2. Lexical precision LP and recall LR for learned ontologies $\mathcal{O}_{\mathrm{Eng}}$ and $\mathcal{O}_{\mathrm{Intl}}$ based on different genre numbers. $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$ serve as reference ontologies (with a fixed number of genres).

$\mathcal{O}_{\mathrm{C}}$	$\mathcal{O}_{\mathrm{WikiData}}$	$\mathcal{O}_{\mathrm{Eng}}$	$\mathcal{O}_{\mathrm{Intl}}$
LP	0.644	0.260	0.183
LR	0.306	0.226	0.166
LF	0.415	0.242	0.174
TP_{csc}	0.098	0.187	0.193
TR_{csc}	0.114	0.220	0.212
TF_{csc}	0.105	0.202	0.202
TP_{con}	0.266	0.237	0.240
TR_{con}	0.319	0.278	0.268
TF_{con}	0.290	0.256	0.253

Table 1. Results for $\mathcal{O}_{\mathrm{R}} = \mathcal{O}_{\mathrm{DBpedia}}$, $\tau = 0.17$, $\upsilon = 0.0001$, $|\mathcal{C}_{\mathrm{Eng}}| = 1000$, $|\mathcal{C}_{\mathrm{Intl}}| = 1041$.

The *taxonomic recall* (TR_{csc}) is:

$$TR_{\mathrm{csc}}(\mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) = TP_{\mathrm{csc}}(\mathcal{O}_{\mathrm{R}}, \mathcal{O}_{\mathrm{C}}) \tag{10}$$

Finally, the *taxonomic F-measure* (TF_{csc}) is defined by:

$$TF_{\mathrm{csc}}(\mathcal{O}_{\mathrm{C}}, \mathcal{O}_{\mathrm{R}}) = \frac{2 \cdot TP_{\mathrm{csc}} \cdot TR_{\mathrm{csc}}}{TP_{\mathrm{csc}} + TR_{\mathrm{csc}}} \tag{11}$$

5. RESULTS AND DISCUSSION

We measured the similarity of all four ontologies using varying parameters for the learned ones. Section 5.1 reports lexical results, Section 5.2 conceptual results. In Section 5.3 we discuss our findings.

5.1 Lexical Results

How similar are the ontologies on the lexical level? For the reference ontologies $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$ this is easy to answer: $LP/LR/LF(\mathcal{O}_{\mathrm{WikiData}}, \mathcal{O}_{\mathrm{DBpedia}}) = 0.64/0.31/0.42$ (Table 1). Given their respective sizes, the highest possible values for this pairing are $1.00/0.48/0.64$ (if $\mathcal{C}_{\mathrm{WikiData}} \subset \mathcal{C}_{\mathrm{DBpedia}}$).

$\mathcal{O}_{\mathrm{C}}$	$\mathcal{O}_{\mathrm{DBpedia}}$	$\mathcal{O}_{\mathrm{Eng}}$	$\mathcal{O}_{\mathrm{Intl}}$
LP	0.303	0.259	0.202
LR	0.638	0.473	0.384
LF	0.411	0.335	0.264
TP_{csc}	0.114	0.174	0.181
TR_{csc}	0.098	0.151	0.149
TF_{csc}	0.105	0.162	0.163
TP_{con}	0.319	0.305	0.357
TR_{con}	0.266	0.274	0.303
TF_{con}	0.290	0.288	0.328

Table 2. Results for $\mathcal{O}_{\mathrm{R}}$= $\mathcal{O}_{\mathrm{WikiData}}$, $\tau = 0.17$, $\upsilon = 0.0001$, $|\mathcal{C}_{\mathrm{Eng}}|= 1000$, $|\mathcal{C}_{\mathrm{Intl}}|= 1041$.

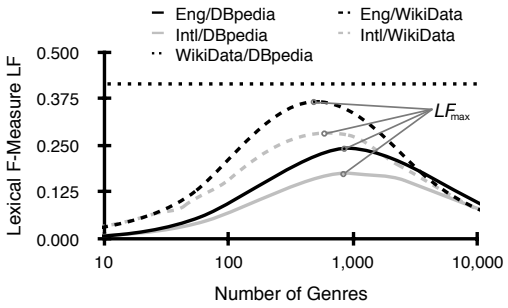


Figure 3. Lexical F-measure LF for learned ontologies $\mathcal{O}_{\mathrm{Eng}}$ and $\mathcal{O}_{\mathrm{Intl}}$ based on different genre numbers. $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$ serve as reference ontologies (with a fixed number of genres).

For the learned ontologies, the answer depends on the number of genres used during generation. Not surprisingly, we observed that recall increases with the number of genres, while precision decreases. When comparing precision/recall values for the learned ontologies with $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$, values for $\mathcal{O}_{\mathrm{WikiData}}$ are predominantly higher, indicating a greater similarity with the learned ontologies (dashed lines in Figure 2). This is also reflected in the lexical F-measure shown in Figure 3. While $LF_{\max}(\mathcal{O}_{\mathrm{Eng}}, \mathcal{O}_{\mathrm{DBpedia}})$ is only 0.24, $LF_{\max}(\mathcal{O}_{\mathrm{Eng}}, \mathcal{O}_{\mathrm{WikiData}})$ is 0.37—just 0.05 below $LF(\mathcal{O}_{\mathrm{WikiData}}, \mathcal{O}_{\mathrm{DBpedia}})$, shown as dotted line. For $\mathcal{O}_{\mathrm{Intl}}$, the $LF_{\max}$ values are lower than their $\mathcal{O}_{\mathrm{Eng}}$ counterparts: $LF_{\max}(\mathcal{O}_{\mathrm{Intl}}, \mathcal{O}_{\mathrm{DBpedia}})$ is 0.18 and $LF_{\max}(\mathcal{O}_{\mathrm{Intl}}, \mathcal{O}_{\mathrm{WikiData}})$ is 0.28. In all cases, the number of genres needed to achieve $LF_{\max}$ approximately equals the number of genres in the reference ontology.

When generated for very few genres, both learned ontologies reach $LP = 1.0$ for either reference ontology, as they all contain the top genres `rock`, `pop`, etc. The achievable LR values however, differ significantly. At a very low precision level, both learned ontologies reach no more than $LR = 0.5$ with $\mathcal{O}_{\mathrm{DBpedia}}$ as reference. In contrast, at the same precision level, with $\mathcal{O}_{\mathrm{WikiData}}$ as reference, LR is greater than 0.74 (Figure 2). We investigated what might be the reason for the low recall for $\mathcal{O}_{\mathrm{DBpedia}}$ and came to the conclusion that it contains many genres that are unlikely to be found in standard genre tags, e.g. `Music of Innsbruck` or `Music of Guangxi`.

5.2 Conceptual Results

Just like the lexical results, conceptual results depend on the number of genres considered and of course the reference used. Additionally, τ and υ influence the outcome.

We found that values for $\upsilon \leq 0.0001$ hardly affect $TP/TR/TF$ results, when the learned ontology is compared with $\mathcal{O}_{\mathrm{DBpedia}}$ or $\mathcal{O}_{\mathrm{WikiData}}$. However, inspection of the learned ontologies shows, that a very low υ causes some genres to have significantly more parents than the average genre. Consequently, they connect unrelated parts of the ontology. Examples for this are `canadian` and `seventies`. We argue that neither is really a musical genre, but rather an orthogonal concept—a region and an era, respectively. This also explains why $TP/TR/TF$ are unaffected, as by definition they are only influenced by genres that appear in both the learned and the reference ontology. Being orthogonal to the genre concept, they never occur in a reference ontology. We further observed, that υ values greater than 0.0001 affect $TP/TR/TF$ negatively. The following data are therefore based on $\upsilon = 0.0001$.

We investigated how τ influences $TP/TR/TF$ by calculating TF_{csc} for $\mathcal{O}_{\mathrm{Eng}}$ ($|\mathcal{C}_{\mathrm{Eng}}|$=1000) and $\mathcal{O}_{\mathrm{Intl}}$ ($|\mathcal{C}_{\mathrm{Intl}}|$=1041) with $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$ as reference ontologies. Based on Figure 4, we chose $\tau = 0.17$ as a value reasonably suited for all ontologies.

Keeping τ and υ constant, how are taxonomic results influenced by the number of genres? $TF_{\mathrm{csc}}(\mathcal{O}_{\mathrm{Eng}}, \mathcal{O}_{\mathrm{DBpedia}})$ peaks around 160 genres with $TF_{\max} = 0.31$. The same is true for $TF_{\mathrm{csc}}(\mathcal{O}_{\mathrm{Eng}}, \mathcal{O}_{\mathrm{WikiData}})$ with $TF_{\max} = 0.32$. For $TF_{\mathrm{csc}}(\mathcal{O}_{\mathrm{Intl}}, \mathcal{O}_{\mathrm{DBpedia}})$ we found $TF_{\max}$ around 285 genres with a value of 0.26 and for $TF_{\mathrm{csc}}(\mathcal{O}_{\mathrm{Intl}}, \mathcal{O}_{\mathrm{WikiData}})$ around 411 genres with 0.28 (Figure 5a). In all cases, TF_{csc} peaks for genre numbers that are well below the number of genres in the reference ontology. This makes sense as all ontologies, to a large degree, consist of disconnected genres that cannot contribute to a higher TF_{csc}. But even for most non-$TF_{\max}$ genre numbers, TF_{csc} values involving the learned ontologies are higher than $TF_{\mathrm{csc}}(\mathcal{O}_{\mathrm{WikiData}}, \mathcal{O}_{\mathrm{DBpedia}}) = 0.12$, depicted as the dotted line in Figure 5a. It appears, as if both $\mathcal{O}_{\mathrm{Eng}}$ and $\mathcal{O}_{\mathrm{Intl}}$ are taxonomically more similar to $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$ than $\mathcal{O}_{\mathrm{DBpedia}}$ to $\mathcal{O}_{\mathrm{WikiData}}$ or the other way around. Upon closer inspection, we attributed this to the greater intersection of disconnected genres from $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$. 47.6% of the genres in the lexical intersection $\mathcal{C}_{\mathrm{DBpedia}} \cap \mathcal{C}_{\mathrm{WikiData}}$ are disconnected in at least one of the two ontologies. But only 36.9% of the $\mathcal{O}_{\mathrm{WikiData}}$ intersection with $\mathcal{O}_{\mathrm{Eng}}$ and 38.6% of the intersection with $\mathcal{O}_{\mathrm{Intl}}$ are disconnected. Even lower are the values for $\mathcal{O}_{\mathrm{DBpedia}}$: Just 17.7% of the intersection with $\mathcal{O}_{\mathrm{Eng}}$ and 16.7% of the intersection with $\mathcal{O}_{\mathrm{Intl}}$ are disconnected. As defined in Section 4.2, disconnected genres lead to zero tp_{csc} values. This significantly contributes to $\mathcal{O}_{\mathrm{DBpedia}}$ and $\mathcal{O}_{\mathrm{WikiData}}$ achieving lower

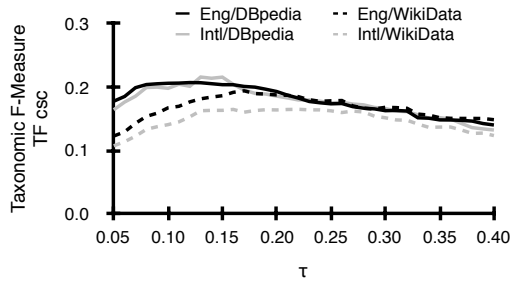


Figure 4. Taxonomic F-measure TF_{csc} for $\mathcal{O}_{\text{Eng}}$ ($|\mathcal{C}_{\text{Eng}}|$=1000) and $\mathcal{O}_{\text{Intl}}$ ($|\mathcal{C}_{\text{Intl}}|$=1040) compared with $\mathcal{O}_{\text{DBpedia}}$ and $\mathcal{O}_{\text{WikiData}}$ for varying τ values.

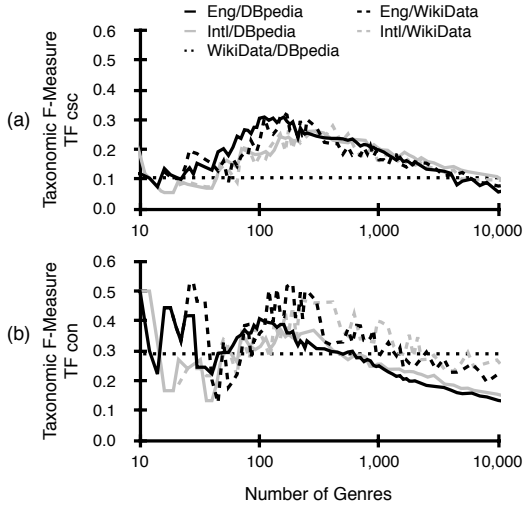


Figure 5. Taxonomic F-measures TF_{csc} and TF_{con} for learned ontologies and different genre numbers compared with $\mathcal{O}_{\text{DBpedia}}$ and $\mathcal{O}_{\text{WikiData}}$. $\tau = 0.17$, $\upsilon = 0.0001$.

$TP/TR/TF$ values, when compared with each other, than a pairing that does not have as many disconnected genres in common. By removing all disconnected genres from $\mathcal{C}_{\text{C}} \cap \mathcal{C}_{\text{R}}$ before calculating TP_{csc}, we calculated the *connected taxonomic precision* (TP_{con}), which results in higher values for all pairings, and especially for ($\mathcal{O}_{\text{DBpedia}}$,$\mathcal{O}_{\text{WikiData}}$) (Figure 5b). The problem with genre ontologies is, that from a taxonomic point of view, the reference ontologies are, to a large degree, trivial. TP_{con} attempts to work around the problem by accepting that there are disconnected genres and ignores them when calculating TP.

5.3 Discussion

The results show that, using the proposed method, it is possible to create an ontology that is almost as similar to $\mathcal{O}_{\text{WikiData}}$ as the alternative reference ontology $\mathcal{O}_{\text{DBpedia}}$—on both the lexical and conceptual level. When comparing learned ontologies with the more comprehensive $\mathcal{O}_{\text{DBpedia}}$, the results are not quite as good: while it is possible to generate an ontology that is as similar to $\mathcal{O}_{\text{DBpedia}}$ as $\mathcal{O}_{\text{WikiData}}$ on the conceptual level, it was not possible on the lexical level due to the many uncommon genres contained in DBpedia.

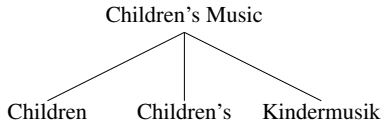


Figure 6. Declinations and translations in $\mathcal{O}_{\text{Intl}}$.

Sourcing genre tags from international instead of just English users has proven detrimental to lexical similarity, when comparing with either $\mathcal{O}_{\text{DBpedia}}$ or $\mathcal{O}_{\text{WikiData}}$. When inspecting $\mathcal{O}_{\text{Intl}}$, we noted translations and declinations of genre names. They are often close relatives in the generated hierarchy (e.g. Figure 6). On one hand, this clearly contributed to worse lexical results. On the other hand, we see this as a potentially useful property. Different crowd-sourced notations in a reference ontology simplify lookups, because there is no mismatch between the names that are really being used and the names that occur in the ontology. Furthermore, it allows easy measurement of semantic similarity for unknown notations or translations, e.g. via the length of the shortest connecting path. It also adds a cultural dimension, as `children's music` and `Kindermusik` are clearly the same genre, but a parent looking for music may prefer music from its own culture and chooses one genre over the other.

All differences put aside, one must not forget that the mentioned ontologies can be linked and thus complement each other. A missing connection in one ontology, may be made through another one. The generated ontologies can be found at `http://www.tagtraum.com/learned_ontologies.html` and contain *sameAs*-relations with WikiData and DBpedia.

6. CONCLUSION AND FUTURE WORK

DBpedia and WikiData both consist of two parts: The first part contains disconnected genres that have neither parents nor sub-genres. It has little value in a taxonomic sense, but can still serve as linkable data in LOD-space. The second part is an imperfect, but rich, interconnected hierarchy of relatively popular genres that can be used for similarity estimation and therefore recommender systems. Because of the way DBpedia is created, not all language editions are represented equally well.

By exploiting co-occurrence rates of user submitted genre labels, we were able to learn new genre ontologies. Using established lexical and conceptual similarity measures, we successfully demonstrated the validity of the proposed learning method. Further, to improve conceptual similarity measures with largely trivial reference ontologies, we proposed an additional measure, the connected taxonomic precision.

Future work may add translation recognition and improve genre name normalization. Taking advantage of learned genre ontologies may lead to interesting new music information retrieval applications.

7. REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *DBpedia: A nucleus for a web of open data*. Springer, 2007.

[2] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):28–37, 2001.

[3] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. DBpedia–a crystallization point for the web of data. *Web Semantics: science, services and agents on the world wide web*, 7(3):154–165, 2009.

[4] Òscar Celma and Pedro Cano. From hits to niches? or how popular artists can bias music recommendation and discovery. In *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, page 5. ACM, 2008.

[5] Philipp Cimiano, Andreas Hotho, and Steffen Staab. Learning concept hierarchies from text corpora using formal concept analysis. *Journal of AI Research (JAIR)*, 24:305–339, 2005.

[6] Philipp Cimiano, Alexander Mädche, Steffen Staab, and Johanna Völker. Ontology learning. In *Handbook on ontologies*, pages 245–267. Springer, 2009.

[7] Klaas Dellschaft and Steffen Staab. On how to perform a gold standard based evaluation of ontology learning. In *The Semantic Web-ISWC 2006*, pages 228–241. Springer, 2006.

[8] Tommaso Di Noia, Roberto Mirizzi, Vito Claudio Ostuni, Davide Romito, and Markus Zanker. Linked open data to support content-based recommender systems. In *Proceedings of the 8th International Conference on Semantic Systems*, pages 1–8. ACM, 2012.

[9] Ben Fields, Sam Phippen, and Brad Cohen. A case study in pragmatism: exploring the practical failure modes of linked data as applied to classical music catalogues. In *Proceedings of the 2nd International Workshop on Digital Libraries for Musicology*, pages 21–24. ACM, 2015.

[10] Alexander Mädche and Steffen Staab. Measuring similarity between ontologies. In *Knowledge engineering and knowledge management: Ontologies and the semantic web*, pages 251–263. Springer, 2002.

[11] Sergio Oramas, Mohamed Sordo, Luis Espinosa-Anke, and Xavier Serra. A semantic-based approach for artist similarity. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 100–106, Málaga, Spain, 2015.

[12] François Pachet and Daniel Cazaly. A taxonomy of musical genres. In *Proceedings of the Content-based Multimedia Information Access Conference (RIAO)*, pages 1238–1245, Paris, France, 2000.

[13] Alexandre Passant. dbrec–music recommendations using DBpedia. In *The Semantic Web–ISWC 2010*, pages 209–224. Springer, 2010.

[14] Alexandre Passant and Yves Raimond. Combining social music and semantic web for music-related recommender systems. In *The 7th International Semantic Web Conference*, volume 19, 2008.

[15] Yves Raimond, Samer Abdallah, Mark Sandler, and Frederick Giasson. The music ontology. In *Proceedings of the 8th International Society for Music Information Retrieval Conference (ISMIR)*, pages 417–422, Vienna, Austria, 2007.

[16] Hendrik Schreiber. Improving genre annotations for the million song dataset. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 241–247, Málaga, Spain, 2015.

[17] Mohamed Sordo, Òscar Celma, Martín Blech, and Enric Guaus. The quest for musical genres: Do the experts and the wisdom of crowds agree? In *Proceedings of the 9th International Society for Music Information Retrieval Conference (ISMIR)*, pages 255–260, 2008.

[18] İpek Tatlı and Ayşenur Birtürk. Using semantic relations in context-based music recommendations. In *Workshop on Music Recommendation and Discovery (WOMRAD), ACM Conference on Recommender Systems.*, pages 14–17, Chicago, IL, USA, 2011.

[19] Wilson Wong, Wei Liu, and Mohammed Bennamoun. Ontology learning from text: A look back and into the future. *ACM Computing Surveys (CSUR)*, 44(4):20, 2012.

GENRE SPECIFIC DICTIONARIES FOR HARMONIC/PERCUSSIVE SOURCE SEPARATION

Clément Laroche[1,2] **Hélène Papadopoulos**[2] **Matthieu Kowalski**[2,3] **Gaël Richard**[1]

[1] LTCI, CNRS, Télécom ParisTech, Univ Paris-Saclay, Paris, France
[2] Univ Paris-Sud-CNRS-CentraleSupelec, L2S, Gif-sur-Yvette, France
[3] Parietal project-team, INRIA, CEA-Saclay, France

[1]name.lastname@telecom-paristech.fr, [2]name.lastname@lss.supelec.fr

ABSTRACT

Blind source separation usually obtains limited performance on real and polyphonic music signals. To overcome these limitations, it is common to rely on prior knowledge under the form of side information as in *Informed Source Separation* or on machine learning paradigms applied on a training database. In the context of source separation based on factorization models such as the *Non-negative Matrix Factorization*, this supervision can be introduced by learning specific dictionaries. However, due to the large diversity of musical signals it is not easy to build sufficiently compact and precise dictionaries that will well characterize the large array of audio sources. In this paper, we argue that it is relevant to construct genre-specific dictionaries. Indeed, we show on a task of harmonic/percussive source separation that the dictionaries built on genre-specific training subsets yield better performances than cross-genre dictionaries.

1. INTRODUCTION

Source separation is a field of research that seeks to separate the components of a recorded audio signal. Such a separation has many applications in music such as up-mixing [9] (spatialization of the sources) or automatic transcription [35] (it is easier to work on single sources). The separation task is difficult due to the complexity and the variability of the music mixtures.

The large collection of audio signals can be classified into various musical genres [34]. Genres are labels created and used by humans for categorizing and describing music. They have no strict definitions and boundaries but particular genres share characteristics typically related to instrumentation, rhythmic structure, and pitch content of the music. This resemblance between two pieces of music has been used as an information to improve chord transcription [23, 27] or downbeat detection [13] algorithms. Genre information can be obtained using annotated labels. When the genre information is not available, it can be retrieved using automatic genre classification algorithms [26, 34]. Such classification have never been used to guide a source separation problem and this may be due to the lack of annotated databases. The recent availability of large evaluation databases for source separation that integrate genre information motivates the development of such approaches. Furthermore, Most datasets used for Blind Audio Source Separation (BASS) research are small in size and they do not allow for a thorough comparison of the source separation algorithms. Using a larger database is crucial to benchmark the different algorithms.

In the context of BASS, Non-negative Matrix Factorization (NMF) is a widely used method. The goal of NMF is to approximate a data matrix $V \in \mathbb{R}_+^{n \times m}$ as

$$V \approx \tilde{V} = WH \tag{1}$$

with $W \in \mathbb{R}_+^{n \times k}$, $H \in \mathbb{R}_+^{k \times m}$ and where k is the rank of factorization [21]. In audio signal processing, the input data is usually a Time-Frequency representation such as a Short Time Fourier Transform (STFT) or a constant-Q transform spectrogram. Blind source separation is a difficult problem and the plain NMF decomposition does not provide satisfying results. To obtain a satisfying decomposition, it is necessary to exploit various features that make each source distinguishable from one another. Supervised algorithms in the NMF framework exploit training data or prior information in order to guide the decomposition process. For example, information from the scores or from midi signals can be used to initialize the learning process [7]. The downside of these approaches is that they require well organized prior information that is not always available. Another supervised method consists in performing prior training on specific databases. A dictionary matrix W_{train} can be learned from database in order to separate the target instrument [16, 37]. Such method requires minimum tuning from the user. However, within different music pieces of an evaluation database, the same instrument can sound differently depending on the recording conditions and post processing treatments.

© Clément Laroche, Hélène Papadopoulos, Matthieu Kowalski, Gaël Richard. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Clément Laroche, Hélène Papadopoulos, Matthieu Kowalski, Gaël Richard. "Genre specific dictionaries for harmonic/percussive source separation", 17th International Society for Music Information Retrieval Conference, 2016.

In this paper, we focus on the task of Harmonic Percussive Source Separation (HPSS). HPSS has numerous applications as a preprocessing step for other audio tasks. For example the HPSS algorithm [8] can be used as a preprocessing step to increase the performance for singing pitch extraction and voice separation [14]. Similarly, beat tracking [6] and drum transcription algorithms [29] are more accurate if the harmonic instruments are not part of the analyzed signal.

We built our algorithm using the method developed in [20]: an unconstrained NMF decomposes the audio signal in a sparse orthogonal part that are well suited for representing the harmonic component, while the percussive part is represented by a regular nonnegative matrix factorization decomposition. In [19], we have adapted the algorithm using a trained drum dictionary to improve the extraction of the percussive instruments. As the user databases typically cover a wide variety of genres, instrumentation may strongly differ from one piece to another. In order to better manage the variability and to build effective dictionaries, we propose here to use genre specific training data.

The main contribution of this article is that we develop a genre specific method to build NMF drum dictionaries that gives consistent and robust results on a HPSS task. The *genre specific dictionaries* are able to improve the separation score compared to a *universal dictionary* trained from all available data (i.e. a cross-genre dictionary).

The rest of the paper is organized as follows. Section 2 defines the context of our work, Section 3 presents the proposed algorithm while Section 4 describes the construction of specific dictionaries. Finally Section 5 details the results of the HPSS on 65 audio files and we suggest some conclusions in Section 6.

2. TOWARD GENRE SPECIFIC INFORMATION

2.1 Genre information

Musical genre is one of the most prominent high level music descriptors. Electronic Music Distribution has become more and more popular in recent years and music catalogues never stop to increase (the biggest online services now propose around 30 million tracks). In that context, associating a genre to a musical piece is crucial to help users finding what they are looking for. As mentioned in the introduction, genre information has been used as a cue to improve some content-based music information retrieval algorithms. If an explicit definition of musical genres is not really available [3], musical genre classification can be performed automatically [24].

Source separation has been used extensively in order to help the genre classification process [18,30] but, at the best of our knowledge, the genre information has never been exploited to guide source separation algorithm.

2.2 Methods for dictionary learning

Audio data is largely redundant as it often contains multiple correlated versions of the same physical event (note, drum hits...) [33] hence the idea to exploit this redundancy to reduce the amount of information necessary for the representation of a musical signal.

Many rank reduction methods, such as Single Value Decomposition (K-SVD) [1], Vector Quantization (VQ) [10], Principal Component Analysis (PCA) [15], or Non negative matrix factorization (NMF) [32] are based on the principle that our observations can be described by a sparse subset of atoms taken from a redundant representation. These methods provide a small subset of relevant templates that are later used to guide the extraction of a target instrument.

Building a dictionary using K-SVD has been a successful approach in image processing [39]. However this method does not scale well to process large audio signals as the computational time is unrealistic. Thus a genre specific dictionary scenario cannot be considered in this framework.

VQ has been mainly used for audio compression [10] and PCA has been used for voice extraction [15]. However these methods have not been used yet as a pre-processing step to build a dictionary.

Finally, in the NMF framework, some work has been done to perform a decomposition with learned dictionaries. In [12], a dictionary is built using a physical model of the piano. This method is not adapted to build genre specific dictionaries as the model cannot easily take into account the genre information. A second way to build a dictionary is to directly use the STFT of an instrument signal [37]. This method does not scale well if the training data is large, thus it is not possible to use it to build genre specific dictionaries. Finally, another method to build a dictionary is to compute a NMF decomposition on a large training set specific to the target source [31]. After the optimization process of the NMF, the W matrix from this decomposition is used as a fixed dictionary matrix W_{train}. This method does not give satisfying results on pitched instruments (i.e., harmonic instruments) and the dictionary needs to be adaptated for example using linear filtering on the fixed templates [16]. Compared to state of the art methods, fixed dictionaries provide good results for HPSS [19]. However, the results have a high variance because the dictionaries are learned on general data that do not take into account the large variability of drum sounds. A nice property of the NMF framework is that the rank of the factorization determines the final size of the dictionary and it can be chosen small enough to obtain a strong compression of the original data. The limitations of the current methods motivated us to build genre specific data using NMF in order to obtain relevant compact dictionaries.

2.3 Genre information for HPSS

Current state-of-the-art unsupervised methods for HPSS such as complementary diffusion [28] and constrained NMF [5] cannot be easily adapted to use genre information. We will not discuss these methods in this article.

However supervised methods can be modified to utilize genre information. In [17] the drum source sepa-

ration is done using a Non-Negative Matrix Partial Co-Factorization (NMPCF). The spectrogram of the signal and the drum-only data (obtained from prior learning) are simultaneously decomposed in order to determine common basis vectors that capture the spectral and temporal characteristics of the drum sources. The percussive part of the decomposition is constrained while the harmonic part is completely unconstrained. As a result, the harmonic part tends to decompose a lot of information from the signal and the separation is not satisfactory (i.e., the harmonic part contains some percussive instruments). A drawback of this method is that it does not scale when the training data is large and the computation time is significantly larger compared to other methods.

By contrast, the approach introduced and detailled in [19, 20] appears to be a good candidate to test the genre specific dictionaries: they can be easily integrated to the algorithm without increasing the computation time.

3. STRUCTURED PROJECTIVE NMF (SPNMF)

3.1 Principle of the SPNMF

Using a similar model as in our preliminary work [20], let V be the magnitude spectrogram of the input data. The model is then given by

$$V \approx \tilde{V} = V_H + V_P, \tag{2}$$

with V_P the spectrogram of the percussive part and V_H the spectrogram of the harmonic part. V_H is approximated by the projective NMF decomposition [38] while V_P is decomposed by NMF components which leads to:

$$V \approx \tilde{V} = W_H W_H^T V + W_P H_P. \tag{3}$$

The data matrix is approximated by an almost orthogonal sparse part that codes the harmonic instruments $V_H = W_H W_H^T V$ and a non constrained NMF part that codes the percussive instruments $V_P = W_P H_P$. As a fully unsupervised SPNMF model does not allow for a satisfying harmonic/percussive source separation [20], we propose here to use a fixed genre specific drum dictionary W_P in the percussive part of the SPNMF.

3.2 Algorithm optimization

In order to obtain such a decomposition, we can use a measure of fit $D(x|y)$ between the data matrix V and the estimated matrix $\tilde{V}$. $D(x|y)$ is a scalar cost function and in this article, we use the Itakura Saito (IS) divergence. A discussion about the possible use of other divergences can be found in [19].

The SPNMF model gives the optimization problem:

$$\min_{W_H, W_P, H_P \geq 0} D(V|W_H W_H^T V + W_P H_P) \tag{4}$$

A solution to this problem can be obtained by iterative multiplicative update rules following the same strategy as in [22, 38]. Using formula from Appendix 7, the optimization process is given in Algorithm 1, where $\otimes$ is the Hadamard product and all division are element-wise operation.

Input: $V \in \mathbb{R}_+^{m \times n}$ and $W_{train} \in \mathbb{R}_+^{m \times e}$ Output: $W_H \in \mathbb{R}_+^{m \times k}$ and $H_P \in \mathbb{R}_+^{e \times n}$ Initialization;
while $i \leq$ *number of iterations* **do**
  $H_P \leftarrow H_P \otimes \frac{[\nabla_{H_P} D(V|\tilde{V})]^-}{[\nabla_{H_P} D(V|\tilde{V})]^+}$
  $W_H \leftarrow W_H \otimes \frac{[\nabla_{W_H} D(V|\tilde{V})]^-}{[\nabla_{W_H} D(V|\tilde{V})]^+}$
  $i = i + 1$
end
$X_P = W_{train} H_P$ and $X_H = W_H W_H^T V$

Algorithm 1: SPNMF with a fixed trained drum dictionary matrix.

3.3 Signal reconstruction

The percussive signal $x_p(t)$ is synthesized using the magnitude percussive spectrogram $X_P = W_P H_P$. To reconstruct the phase of the percussive part, we use a Wiener filter [25] to create a percussive mask as:

$$\mathcal{M}_P = \frac{X_P^2}{X_H^2 + X_P^2} \tag{5}$$

To retrieve the percussive signal as:

$$x_p(t) = \text{SFTF}^{-1}(\mathcal{M}_P \otimes X). \tag{6}$$

Where X is the complex spectrogram of the mixture. We use a similar procedure for the harmonic part.

4. CONSTRUCTION OF THE DICTIONARY

In this section we detail the building process of the drum dictionary. We present in Section 4.1 tests conducted on the SiSEC 2010 database [2] in order to find the optimal size to build the genre specific dictionaries. In Section 4.2 we describe the training and the evaluation database. Finally, in Section 4.3, we detail the protocol to build the genre specific dictionaries.

4.1 Optimal size for the dictionary

The NMF model is given by (1). If V is the power spectrum of a drum signal, The matrix W is a *dictionary* or a set of *patterns* that codes the frequency information of the drum. The first step to build a NMF drum dictionary is to select the rank of factorization. In order to avoid overfitting, the algorithm is optimized using databases different from the database used for evaluation, described in Section 4.2.

We run the optimization tests on the public SiSec database [2]. The database is composed of four polyphonic real-world music excerpts and each music signal contains percussive, harmonic instruments and vocals. The duration of the recordings is ranging from 14 to 24 s. In the context of HPSS, following the same protocol as in [5], we do not consider the vocal part and we build the mixture signals from the percussive and harmonic instruments only. The signals are sampled at 44.1 kHz. We compute the STFT with a 2048 sample long Hann window with a 50%

overlap. Furthermore, the rank of factorization of the harmonic part of the SPNMF algorithm is set to $k = 100$, as in [19].

A fixed drum dictionary is built using the database ENST-Drums [11]. For this, we concatenate 30 files where the drummer is playing a *drum phrase* that result in an excerpt of around 10 min duration. We then compute an NMF decomposition with different ranks of factorization ($k = 12$, $k = 50$, $k = 100$, $k = 200$, $k = 300$, $k = 500$, $k = 1000$ and $k = 2000$) on the drum signal alone to obtain 8 drum dictionaries.

These dictionaries are then used to perform a HPSS on the four songs of the SiSEC database using the SPNMF algorithm (see Algorithm 1). The results are compared by means of the Signal-to-Distortion Ratio (SDR), the Signal-to-Interference Ratio (SIR) and the Signal-to-Artifact Ratio (SAR) of each of the separated sources using the BSS Eval toolbox provided in [36].

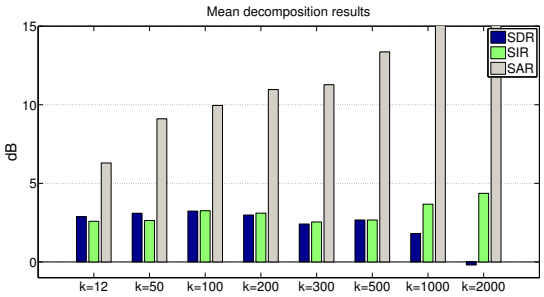


Figure 1: Influence of k on the S(D/I/A)R on the SiSEC database.

The results in Figure 1 show that the optimal value for the SDR and SIR is reached for $k = 100$, then the SDR decreases for $k \geqslant 200$. For $k \geqslant 500$ the harmonic signal provided by the algorithm contains most of the original signal therefore the SAR is very high but the decomposition quality is poor. For the rest of the article, the size of the drum dictionaries will be $k = 100$.

4.2 Training and evaluation database

The evaluation tests are conducted on the Medley-dB database [4] composed of polyphonic real-world music excerpts. It consists in 122 music signals and 85 of them contain percussive instruments, harmonic instruments and vocals. The signals that do not contain a percussive part are excluded from evaluation. The genres are distributed as follows: *Classical* (8 songs), *Singer/Songwriter* (17 songs), *Pop* (10 songs), *Rock* (20 songs), *Jazz* (11 songs), *Electronic/Fusion* (13 songs) and *World/Folk* (6 songs). It is important to note that, because the notion of genre is quite subjective (see Section 2), the Medley-dB database uses general genre labels that cannot be considered to be precise. There are many instances where a song could have fallen in multiple genres, and the choices were made so that each genre would be as acoustically homogeneous as possible. Moreover, as we are only working with the instrumental part of the song (the vocals are omitted), the *Pop* label (for example) is similar to the *Singer/Songwriter*. We separate the database into training and evaluation files, as detailed in the next section.

Genre	Artist Song
Classical	JoelHelander Definition MatthewEntwistle AnEveningWithOliver MusicDelta Beethoven
Electronic/Fusion	EthanHein 1930sSynthAndUprightBass TablaBreakbeatScience Animoog TablaBreakbeatScience Scorpio
Jazz	CroqueMadame Oil MusicDelta BebopJazz MusicDelta ModalJazz
Pop	DreamersOfTheGhetto HeavyLove NightPanther Fire StrandOfOaks Spacestation
Rock	BigTroubles Phantom Meaxic TakeAStep PurlingHiss Lolita
Singer/Songwriter	AimeeNorwich Child ClaraBerryAndWooldog Boys InvisibleFamiliars DisturbingWildlife
World/Folk	AimeeNorwich Flying KarimDouaidy Hopscotch MusicDelta ChineseYaoZu
Non specific	JoelHelander Definition TablaBreakbeatScience Animoog MusicDelta BebopJazz DreamersOfTheGhetto HeavyLove BigTroubles Phantom AimeeNorwich Flying MusicDelta ChineseYaoZu

Table 1: Song selected for the training database.

4.3 Genre specific dictionaries

Seven genre-specific drum dictionaries are built using 3 songs of each genre. In addition, a cross-genre drum dictionary is built using half of one song of each genre. Finally, a dictionary is built using the 10 min excerpt of pure drum signals from the ENST-Drums database described in Section 4.1. The Medley-dB files selected for training are given in Table 1 and excluded from evaluation.

With the results from Section 4.1 the dictionaries are built as follows: for every genre specific subset of the training database, we perform a NMF on the drum signals with $k = 100$. The resulting W matrices of the NMF are then used in the SPNMF algorithm as the W_P matrix (see Algorithm 1).

5. RESULTS

In this section, we present the results of the SPNMF with the genre specific dictionaries on the evaluation database from Medley-dB.

5.1 Comparison of the dictionaries

We perform a HPSS on the audio files using the SPNMF algorithm with the 9 dictionaries built in Section 4.3. The results on each song are then sorted by genres and the average results are displayed using box-plots. Each box-plot is made up of a central line indicating the median of

the data, upper and lower box edges indicating the 1^{st} and 3^{rd} quartiles while the whiskers indicate the minimum and maximum values.

Figures 2, 3 and 4 show the SDR, SAR and SIR results for all the dictionaries on the *Pop* subset, giving an overall idea of the performance of the dictionaries inside a specific sub-database. The *Pop* dictionary leads to the highest SDR and SIR and the non specific dictionaries are not performing as well. On this sub-database, the genre specific data gives relevant information to the algorithm. As stated in Section 4.2, some genres are similar to others, explaining why the *Rock* and the *Singer* dictionaries are also providing good results. An interesting result is that compared to the non specific dictionaries, the *Pop* dictionary has a lower variance. Genre information allows for a higher robustness to the variety of the songs within the same genre. Samples of the audio results can be found on the website [1] .

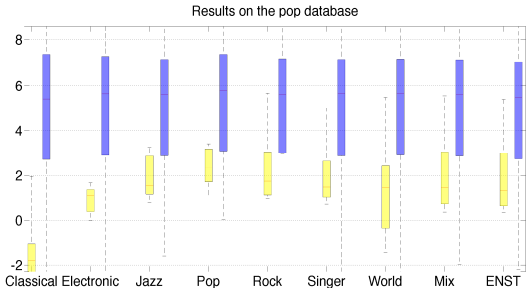


Figure 2: Percussive (left bar)/Harmonic (right bar) SDR results on the *Pop* sub-database using the SPNMF with the 9 dictionaries.

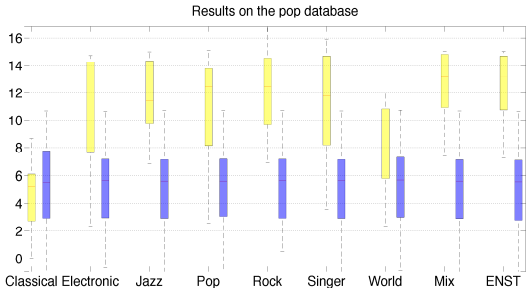


Figure 3: Percussive (left bar)/Harmonic (right bar) SIR results on the *Pop* sub-database using the SPNMF with the 9 dictionaries.

On Table 2, we display the mean separation score for all the genre specific dictionaries compared to the non specific dictionary. The dictionary built on the ENST-drums is giving results very similar to the universal dictionary built on the Medley-dB database. For the sake of concision we only display the results using the universal dictionary from Medley-dB. On the database *Singer/Songwriter*, *Pop*, *Rock*, *Jazz* and *World/Folk*, the genre specific dictionaries outperform the universal dictionary on the harmonic and percussive separation.

[1] https://goo.gl/4X2jk5

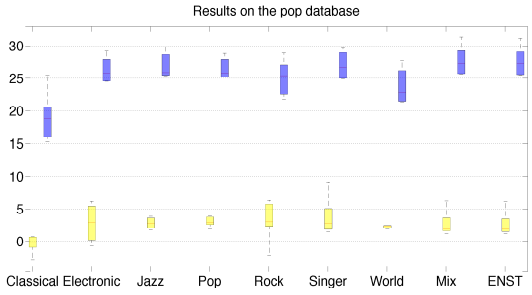


Figure 4: Percussive (left bar)/Harmonic (right bar) SAR results on the *Pop* sub-database using the SPNMF with the 9 dictionaries.

5.2 Discussion

The cross-genre dictionary as well as the ENST-drum dictionary are outperformed by the genre specific dictionaries. The information from the music of the same genre is not altered by the NMF compression and provides drum templates closer to the target drum. The databases *Classical* and *Electronic/Fusion* are composed of songs where the drum is only playing for a few moments. Similarly on some songs of the *Electronic/Fusion* database, the electronic drum reproduces the same pattern during the whole song making the drum part very redundant. As a result, in both cases the drum dictionary does not contain a sufficient amount of information to outperform the universal dictionary. Because of these two factors, the genre specific dictionaries are not performing correctly.

It can be noticed that overall the harmonic separation is giving much better results than the percussive extraction. The fixed dictionaries are creating artefact as the percussive templates do not correspond exactly to the target drum signal. A possible way to alleviate this problem would be to adapt the dictionaries but this would require the use of hyper parameters and that is not the philosophy of this work [20].

6. CONCLUSION

Using genre specific information in order to build more relevant drum dictionaries is a powerful approach to improve the HPSS. The dictionaries still have an imprint of the genre after the NMF decomposition and the additional information is properly used by the SPNMF to improve the source separation quality. This is a first step in order to produce dictionaries capable of separating a wide variety of audio signal.

Future work will be dedicated into building a blind method to select the genre specific dictionary in order to perform the same technique on database where the genre information is not available.

Genre	Classical	Electronic/Fusion	Jazz	Pop	Rock	Singer/Songwriter	World/Folk
Percussive separation							
Genre specific (dB)							
SDR	-1.6	-0.6	**0.4**	**2.5**	**-0.2**	**0.6**	**0.4**
SIR	8.2	15.2	**9.6**	12.3	**19.8**	11.5	**6.1**
SAR	5.9	0.3	**2.1**	**3.4**	0.3	**4.5**	**16.3**
Non specific (dB)							
SDR	**-0.0**	**-0.3**	-0.7	2.0	-2.2	-0.0	-3.6
SIR	**11.3**	**17.0**	9.6	**12.6**	18.3	**13.0**	2.8
SAR	**8.1**	**0.4**	0.9	2.7	**2.3**	1.8	12.1
Harmonic Separation							
Genre specific (dB)							
SDR	**7.5**	**1.6**	**13.0**	**5.1**	**2.1**	7.2	**4.9**
SIR	**10.6**	**1.8**	**13.3**	**5.0**	2.2	**11.5**	**13.5**
SAR	18.2	23.5	28.5	24.5	**36.0**	28.5	**22.7**
Non specific (dB)							
SDR	6.0	1.3	12.7	4.8	1.9	**7.5**	4.6
SIR	7.1	1.4	12.8	4.9	**2.9**	7.5	13.3
SAR	**27.2**	**27.7**	**29.9**	**26.2**	34.3	**31.9**	21.6

Table 2: Average SDR, SIR and SAR results on the Medley-dB database.

7. APPENDIX: SPNMF WITH THE IS DIVERGENCE

The Itakura Saito divergence gives us the problem,

$$\min_{W_H, W_P, H_P \geq 0} \frac{V}{\tilde{V}} - log(\frac{V}{\tilde{V}}) - 1.$$

The gradient wrt W_H gives

$$[\nabla_{W_H} D(V|\tilde{V})]^-_{i,j} = (ZV^T W_H)_{i,j} + (VZ^T W_H)_{i,j},$$

with $Z_{i,j} = (\frac{V}{W_H W_H^T V + W_P H_P})_{i,j}$. The positive part of the gradient is

$$[\nabla_{W_H} D(V|\tilde{V})]^+_{i,j} = (\phi V^T W_H)_{i,j} + (V\phi^T W_H)_{i,j},$$

with

$$\phi_{i,j} = (\frac{I}{W_H W_H^T V + W_P H_P})_{i,j}.$$

and $I \in \mathbb{R}^{f \times t}; \forall i,j \quad I_{i,j} = 1$.

Similarly, the gradient wrt H_P gives

$$[\nabla_{H_P} D(V|\tilde{V})]^- = W_P^T V$$

and

$$[\nabla_{H_P} D(V|\tilde{V})]^+ = 2W_P^T W_H W_H^T V + W_P^T W_P H_P.$$

8. REFERENCES

[1] M. Aharon, M. Elad, and Alfred A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, pages 4311–4322, 2006.

[2] S. Araki, A. Ozerov, V. Gowreesunker, H. Sawada, F. Theis, G. Nolte, D. Lutter, and N. Duong. The 2010 signal separation evaluation campaign: audio source separation. In *Proc. of LVA/ICA*, pages 114–122, 2010.

[3] J.J. Aucouturier and F. Pachet. Representing musical genre: A state of the art. *Journal of New Music Research*, pages 83–93, 2003.

[4] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. Bello. MedleyDB: A multitrack dataset for annotation-intensive MIR research. In *Proc. of ISMIR*, 2014.

[5] F. Canadas-Quesada, P. Vera-Candeas, N. Ruiz-Reyes, J. Carabias-Orti, and P. Cabanas-Molero. Percussive/harmonic sound separation by non-negative matrix factorization with smoothness/sparseness constraints. *EURASIP Journal on Audio, Speech, and Music Processing*, pages 1–17, 2014.

[6] D. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, pages 51–60, 2007.

[7] S. Ewert and M. Müller. Score-informed source separation for music signals. *Multimodal music processing*, pages 73–94, 2012.

[8] D. Fitzgerald. Harmonic/percussive separation using median filtering. In *Proc. of DAFx*, 2010.

[9] D. Fitzgerald. Upmixing from mono-a source separation approach. In *Proc. of IEEE DSP*, pages 1–7, 2011.

[10] A. Gersho and R.M. Gray. *Vector quantization and signal compression*. Springer Science & Business Media, 2012.

[11] O. Gillet and G. Richard. Enst-drums: an extensive audio-visual database for drum signals processing. In *Proc. of ISMIR*, pages 156–159, 2006.

[12] R. Hennequin, B. David, and R. Badeau. Score informed audio source separation using a parametric

model of non-negative spectrogram. In *Proc. of IEEE ICASSP*, 2011.

[13] J. Hockman, M. Davies, and I. Fujinaga. One in the jungle: Downbeat detection in hardcore, jungle, and drum and bass. In *Proc. of ISMIR*, pages 169–174, 2012.

[14] C. Hsu, D. Wang, J.R. Jang, and K. Hu. A tandem algorithm for singing pitch extraction and voice separation from music accompaniment. *IEEE Transactions on Audio, Speech, and Language Processing.*, pages 1482–1491, 2012.

[15] P. Huang, S.D. Chen, P. Smaragdis, and M. Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *Proc. of IEEE ICASSP*, pages 57–60, 2012.

[16] X. Jaureguiberry, P. Leveau, S. Maller, and J. Burred. Adaptation of source-specific dictionaries in non-negative matrix factorization for source separation. In *Proc. of IEEE ICASSP*, pages 5–8, 2011.

[17] M. Kim, J. Yoo, K. Kang, and S. Choi. Nonnegative matrix partial co-factorization for spectral and temporal drum source separation. *Journal of Selected Topics in Signal Processing*, pages 1192–1204, 2011.

[18] A. Lampropoulos, P. Lampropoulou, and G. Tsihrintzis. Musical genre classification enhanced by improved source separation technique. In *Proc. of ISMIR*, pages 576–581, 2005.

[19] C. Laroche, M. Kowalski, H. Papadopoulous, and G.Richard. Structured projective non negative matrix factorization with drum dictionaries for harmonic/percussive source separation. *Submitted to IEEE Transactions on Acoustics, Speech and Signal Processing*.

[20] C. Laroche, M. Kowalski, H. Papadopoulous, and G.Richard. A structured nonnegative matrix factorization for source separation. In *Proc. of EUSIPCO*, 2015.

[21] D. Lee and S. Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, pages 788–791, 1999.

[22] D. Lee and S. Seung. Algorithms for non-negative matrix factorization. *Proc. of NIPS*, pages 556–562, 2001.

[23] K. Lee and M. Slaney. Acoustic chord transcription and key extraction from audio using key-dependent hmms trained on synthesized audio. *IEEE Transactions on Audio, Speech, and Language Processing*, pages 291–301, 2008.

[24] T. Li, M.Ogihara, and Q. Li. A comparative study on content-based music genre classification. In *Proc. of ACM*, pages 282–289, 2003.

[25] A. Liutkus and R. Badeau. Generalized wiener filtering with fractional power spectrograms. In *Proc. of IEEE ICASSP*, pages 266–270, 2015.

[26] C. McKay and I. Fujinaga. Musical genre classification: Is it worth pursuing and how can it be improved? In *Proc. of ISMIR*, pages 101–106, 2006.

[27] Y. Ni, M. McVicar, R. Santos-Rodriguez, and T. De Bie. Using hyper-genre training to explore genre information for automatic chord estimation. In *Proc. of ISMIR*, pages 109–114, 2012.

[28] N. Ono, K. Miyamoto, J. Le Roux, H. Kameoka, and S. Sagayama. Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram. In *Proc. of EUSIPCO*, 2008.

[29] J. Paulus and T. Virtanen. Drum transcription with non-negative spectrogram factorisation. In *Proc. of EUSIPCO*, pages 1–4, 2005.

[30] H. Rump, S. Miyabe, E. Tsunoo, N. Ono, and S. Sagayama. Autoregressive mfcc models for genre classification improved by harmonic-percussion separation. In *Proc. of ISMIR*, pages 87–92, 2010.

[31] M.N. Schmidt and R.K. Olsson. Single-channel speech separation using sparse non-negative matrix factorization. In *Proc. of INTERSPEECH*, 2006.

[32] P. Smaragdis and JC. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180, 2003.

[33] I. Tošić and P. Frossard. Dictionary learning. *IEEE Transactions on Signal Processing*, pages 27–38, 2011.

[34] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE transactions on Speech and Audio Processing*, pages 293–302, 2002.

[35] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing.*, pages 528–537, 2010.

[36] E. Vincent, R. Gribonval, and C. Févotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, Language Process.*, pages 1462–1469, 2006.

[37] C. Wu and A. Lerch. Drum transcription using partially fixed non-negative matrix factorization. In *Proc. of EUSIPCO*, 2008.

[38] Z. Yuan and E. Oja. Projective nonnegative matrix factorization for image compression and feature extraction. *Image Analysis*, pages 333–342, 2005.

[39] Q. Zhang and B. Li. Discriminative K-SVD for dictionary learning in face recognition. In *Proc. of IEEE CVPR*, pages 2691–2698. IEEE, 2010.

GOOD-SOUNDS.ORG: A FRAMEWORK TO EXPLORE GOODNESS IN INSTRUMENTAL SOUNDS

Giuseppe Bandiera[1] **Oriol Romani Picas**[1] **Hiroshi Tokuda**[2]
Wataru Hariya[2] **Koji Oishi**[2] **Xavier Serra**[1]
[1] Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
[2] Technology Development Dept., KORG Inc., Tokyo, Japan
giuseppe.bandiera@upf.edu, oriol.romani@upf.edu

ABSTRACT

We introduce good-sounds.org, a community driven framework based on freesound.org to explore the concept of goodness in instrumental sounds. Goodness is considered here as the common agreed basic sound quality of an instrument without taking into consideration musical expressiveness. Musicians upload their sounds and vote on existing sounds, and from the collected data the system is able to develop sound goodness measures of relevance for music education applications. The core of the system is a database of sounds, together with audio features extracted from them using MTG's Essentia library and user annotations related to the goodness of the sounds. The web front-end provides useful data visualizations of the sound attributes and tools to facilitate user interaction. To evaluate the framework, we carried out an experiment to rate sound goodness of single notes of nine orchestral instruments. In it, users rated the sounds using an AB vote over a set of sound attributes defined to be of relevance in the characterization of single notes of instrumental sounds. With the obtained votes we built a ranking of the sounds for each attribute and developed a model that rates the goodness for each of the selected sound attributes. Using this approach, we have succeeded in obtaining results comparable to a model that was built from expert generated evaluations.

1. INTRODUCTION

Measuring sound goodness, or quality, in instrumental sounds is difficult due to its intrinsic subjectivity. Nevertheless, it has been shown that there is some consistency among people while discriminating good or bad music performances [1]. Furthermore, recent studies have demonstrated a correlation between the perceived music quality and the musical performance technique [2]. Bearing this in mind, in a previous work [3] we proposed a method to automatically rate goodness by defining a set of sound attributes and by using a set of good/bad labels given by expert musicians. The definition of goodness was treated as a classification problem and an outcome of that work was a mobile application (Cortosia ®) that gives goodness scores in real-time for single notes on a scale from 0 to 100. This score was computed considering the distribution of the features values in the classification step. While developing that system we realized that we could improve the scores, specially their correlation with the perceptual sound goodness, if we could use more training data and include a range of goodness levels given by users rather than the binary good/bad labels that we used. However, the task of labeling sounds this way would have been very time consuming and we would also need more sounds, covering the whole range of sound goodness. To address these issues we are now crowdsourcing the problem. We have developed a website, good-sounds.org, on which users can upload sound content and can tag and rate sounds in various ways.

© Giuseppe Bandiera, Oriol Romani Picas, Hiroshi Tokuda, Wataru Hariya, Koji Oishi, Xavier Serra. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Giuseppe Bandiera, Oriol Romani Picas, Hiroshi Tokuda, Wataru Hariya, Koji Oishi, Xavier Serra. "Good-sounds.org: a framework to explore goodness in instrumental sounds", 17th International Society for Music Information Retrieval Conference, 2016.

2. GOOD-SOUNDS.ORG

Good-sounds.org [1] is an online platform to explore the concept of goodness in instrumental sounds with the help of a community of users. It provides social community features in the web front-end and a framework for sound analysis and modeling in the background. It also includes an API to access the collected data.

2.1 Description

The website has been designed from a user perspective, meant to be modern and to provide a seamless experience. It makes use of state of the art design concepts and community oriented web technologies. The web front-end includes three main sections: (1) a page to list and visualize the uploaded sounds as shown in Figure 1, (2) a page to upload and describe sounds as shown in Figure 2 and (3) a section to gather user ratings and annotations. The visualization page shows a list of all the sounds and it includes filter options to narrow down the results, being able to show things like specific instruments or sounds uploaded a certain date. The upload page allows users to add sounds into the site and also provides a recording tool

[1] *https://good-sounds.org*

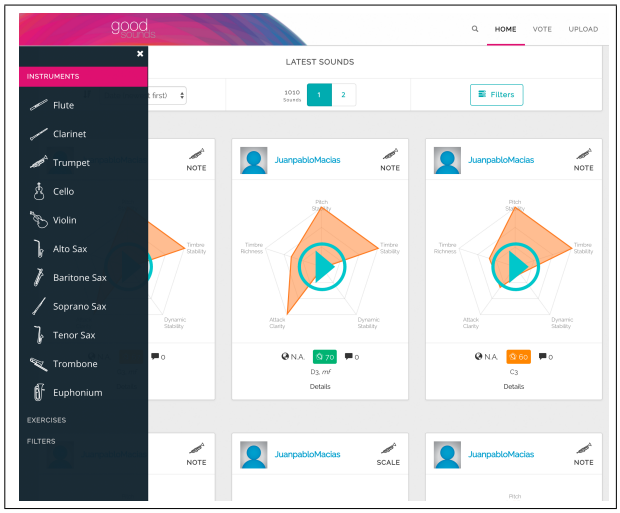


Figure 1. Good-sound.org sound list page.

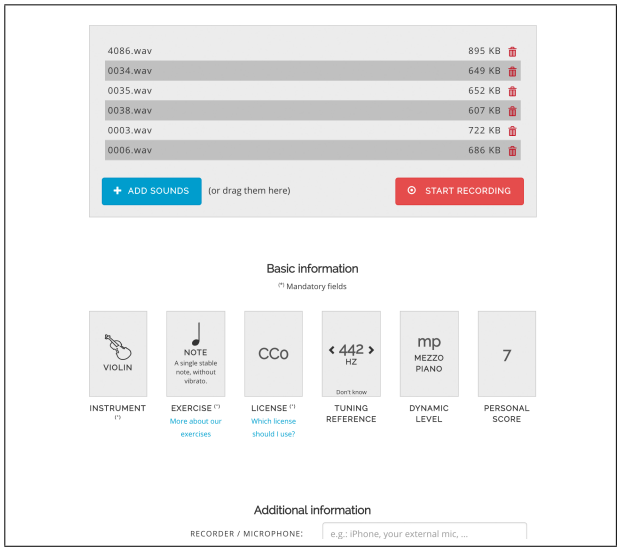


Figure 2. Good-sound.org sound upload page.

built using Web Audio API [2] . The annotation section has been designed for the specific experiment explained in Section 3. The website backend is based on the experience we have obtained in all these years developing and maintaining freesound [2] . It is written in Python using the Django web application framework. The metadata is stored in a PostgreSQL database while the sound files plus other analysis files are stored locally in the server. An API accepts requests from authorized clients to upload sounds (currently through the mobile app Cortosia) and retrieve statistics from the users community. At this time, the website supports 11 instruments, it includes 8470 unique sounds and there are 363 active users.

2.2 Content

The main data stored in good-sounds.org consists of sounds and the metadata accompanying them. When uploading sounds, the users can choose between three different types of Creative Commons licenses for their content: Universal, Attribution or Attribution Non-Commercial. As soon as a sound is uploaded, it is analyzed using the freesound extractor [4], thus obtaining a number of low-level audio features, and the system generates an mp3 version of the file together with images of the waveform and spectrogram. The audio, image and audio feature files are stored in the good-sounds server and the metadata is stored in the PostgreSQL database.

2.2.1 *Segmentation*

One of the critical audio processing steps performed in good-sounds.org is the segmentation of the uploaded sound files to find appropriate note boundaries. Given that the audios come from different and not well controlled sources, they might include all kinds of issues (ex. silence at beginning and end or background noise) that can difficult the subsequent feature extraction steps. Considering that the sounds we are working with are all monophonic pitched instrument sounds, we can base the segmentation mainly on pitch. Our approach extracts pitch using Essentia's [5] implementations of the YinFFT algorithm [8] and the Yin time based algorithm [6]. Then the sound is segmented into notes using pitch contours [7] and signal RMS with Essentias PitchContourSegmentation algorithm. The segmentation data is also stored in the database. This allows us to build client-side data visualizations that effectively reflect the quality of the segmentation algorithm and the user can modify the parameters for this algorithm and rerun it on the fly from the website. The results of this iteration is immediately shown on the same page, for an easy comparison of the results, as it is shown in Figure 3.

Figure 3. Good-sound.org segmentation visualisation page.

2.2.2 *Descriptors*

The feature extraction module is based on the freesound extractor module of Essentia. It computes a subset of its spectral, tonal and temporal descriptors. With it, the audios are first resampled to 44.1kHz sampling rate. The descriptors are then extracted across all the frames using a 2048 window size and 512 hop window size. We then compute statistical measures (mean, median and standard deviation)

[2] *http://www.w3.org/TR/webaudio/*

of the descriptors which are the values stored as JSON files in the server. The list of the descriptors extracted is shown in Table 1.

3. EXPERIMENT

As a test case to evaluate the usefulness of the good-sounds.org framework we setup an experiment to rate the goodness of single notes. The goal of the experiment was to build models from both the uploaded sounds and the community annotations, with which we can then automatically rate the sound goodness. We compared the results of the obtained models with the ones we got in our previous work using expert annotations.

3.1 Dataset

The data used in this experiment comes from several sources. First, we uploaded all the sounds from our previous work to the website, together with the expert annotations. Since the website has been public for a few months, we also had sounds uploaded by users, both directly and through the mobile app (using the API). Then, user annotations on the sounds according to a goodness scale where collected using a voting task. These annotations use a set of sound attributes that affect sound goodness. These attributes were defined in our previous article [3] by consulting with a group of music experts:

- *dynamic stability*: the stability of the loudness.
- *pitch stability*: the stability of the pitch.
- *timbre stability*: the stability of the timbre.
- *timbre richness*: the quality of the timbre.
- *attack clarity*: the quality of the attack.

The sounds from the recording sessions are also uploaded to freesound and thus are openly accessible. We also provide a tool that allow the good-sounds.org users to link their accounts to their freesound ones and upload the sounds there.

3.1.1 Sounds

For this experiment we only used single note sounds. At the time of the experiment there were sounds for 5467 single notes of 9 instruments. We show the list of sounds per instrument in Table 2. The sounds we recorded ourselves from the recording sessions are uncompressed wave files, while the ones uploaded by users to the website are in different audio formats.

3.1.2 Annotations

We distinguish two kinds of annotations: (1) recording annotations and (2) community annotations. The recording annotations are the ones coming from the recording sessions that we did and consists of one tag per sound. This tag says if the sound is a good or a bad example of each sound attribute (e.g. bad-timbre-stability, good-attack-clarity). Those are the annotations used later on for a first evaluation of the models and are only available for the sounds we recorded ourselves. The community annotations are the ones generated from the user votes and used in this work to explore goodness. In order to be able to rate a sound in a goodness scale we need annotations on a wide range of different goodness levels. We originally thought of asking the community to rate sounds in a scale of goodness but we discarded this option because of the following:

- the task can be excessively demanding.
- without a reference sound the criteria of different users can differ extremely.
- with a reference sound we influence the users criteria, thus annotations can be less generalisable.

Instead, we designed a user task that gave as outcome a ranked list of the sounds based on the goodness for each sound attribute. An A/B multi vote task was used for this purpose. Two sounds are presented and the user is asked to decide which sound is better according to one or more of the sound attributes. One vote is stored for each selected attribute. A list of the votes per instruments (considering all sound attributes) is shown in Table 3. In order to prevent random votes in the task we run checks periodically. This checks consists of two sounds; one being a bad example of a sound attribute regarding the expert annotations and the other being a good example. The task is presented to the users the first time they vote and also randomly after some votes. If the user does not vote for the expected sound in the reference task, his next votes are not used. The votes of this user are again taken into account if he succeeds to pass the reference task.

instrument	number of sounds
cello	935
violin	802
clarinet	1360
flute	1434
alto sax	352
baritone sax	292
tenor sax	292
soprano sax	343
trumpet	738

Table 2. Number of sounds in the experiment's dataset.

3.1.3 Rankings

In order to have learning data in a wide range of goodness we built rankings with the community votes for each sound attribute. The position of a sound in the ranking represents its goodness level. To build them we count the number of wins and the number of votes of each sound in the database. Then the sounds are sorted according to two parameters:

- total number of votes: number of participations in the voting task.

spectral	tonal	temporal
spectrum, barkbands, melbands, flatness, crest, rolloff, decrease, hfc, pitch salience, flatness db, skewness, kurtosis, spectral complexity, flatness sfx,	pitch yinfft, pitch yin, pitch confidence,	zerocrossingrate, loudness, centroid,

Table 1. Descriptors extracted by Essentia library present in good-sounds.org.

instrument	number of votes
cello	140
violin	90
clarinet	293
flute	305
alto sax	78
baritone sax	59
tenor sax	14
soprano sax	21
trumpet	230

Table 3. Number of votes in good-sounds for the dataset's experiment.

- ratio between wins and votes: the ratio between the number of wins and the total number of participations in the voting task.

Using these parameters for building the rankings we assure that the sounds in the top are the ones voted more times, as being better than others, and not sounds with few votes but high percentage of wins.

3.2 Learning

The goal of our learning process is to build a model for each instrument that is able to rate each sound attribute in a 0 to 100 score. To do so we want to find a set of features that highly correlate with the rankings extracted in the previous step. Our approach uses a regression model to predict the score. These predictions are then used as samples of the final score function. The final score is then computed as an interpolation of the samples.

3.2.1 Models

We want to find the combination of regression model and set of features that better describes the rankings. For such a purpose we tried different regression algorithms available in scikit-learn [9]. As one of the outputs of the project is a system that rates the goodness of sounds in real-time we want to restrict the number of features in order to maintain a low computational cost. For each one of the algorithms we build a model for each ranking using one, two or three features and we compute the average prediction score of the model across all the options. The prediction score R^2 or Coefficient of Determination is defined as follows:

$$R^2 = (1 - u/v) \tag{1}$$

where

$$u = \sum((y_{true} - y_{pred})^2) \tag{2}$$

Regression model	Avg. score	Score variance
SVR 3 features	-1.208	5.4436
SVR 2 features	-1.2411	5.3895
SVR 1 feature	-1.4254	5.6554

Table 5. Performance of SVR model with different number of features.

and

$$v = \sum((y_{true} - \prod_{i=1}^{n} y_{true})^2) \tag{3}$$

Where y_{true} is the set of ground truth annotations and y_{pred} the set of predictions, having both the same length. The best possible score R^2 is 1.0 and it can be negative. The variance of the prediction score across all the rankings and set of features is also computed. The number of features that give the best score for each ranking is taken into account to compute an average number of features for each regression model. A comparison of the performance of the different models is shown in Table 4.

As we can see in the table, the SVR (Epsilon-Support Vector Regression) model has the best average score across all the rankings and using all possible combination of feature sets (up to 3 features). It also has the lowest score variance so we can expect the model to be robust across the different instruments and sound attributes. However the average number of features is almost two and the computation of two features at each frame of all the sounds in the database can be computationally expensive. For this reason we tested how good the model can be if we force it to use less than three features. We show the results of such a comparison in Table 5.

The results show that the differences between using one or three features are not too big so we decided to use SVR with a single feature in order to maintain a low computational cost for future applications of the system. We then tried all possible combinations of parameters (kernel, degree of polynomially, cost parameter..) to find the best model for each instrument and sound attribute.

3.2.2 Scores

From the model we predict the ranking position of a sound and we map this position into a 0 to 100 score of the sound attribute. The final goodness score is computed as the average score across the five attributes. We compute the sound attribute scores of all the sounds in the database to test the distribution of the scores according to the feature value. For example, a distribution of the score for the timbre stability of flute is shown in Figure 4.

Regression model	Avg. score	Score variance	Average of features
SVR	-1.208	5.4436	1.843
Ridge	-2.644	31.005	2.166
KRR	-1.79	10.798	1.906
Linear regression	-3.503	30.03	1.718
RANSAC	-3.202	17.532	1.478
Theilsen	-4.14	37.135	1.781

Table 4. Performance of the different regression models.

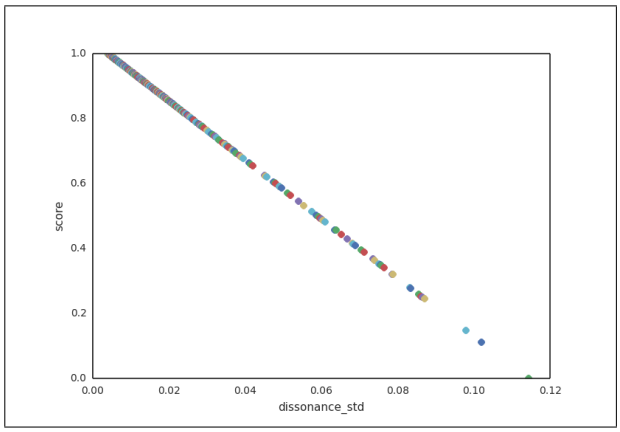


Figure 4. Distribution of scores of flute timbre stability.

The resulting distributions are not balanced. For this reason we push the scores of each sound attribute to fit a Gaussian distribution. This gives us balanced distributions and it also allows us to refine the scores by tweaking the parameters of the gaussian function. A result of this process is shown in Figure 5. The final score is computed interpolating the feature according to these tuned distributions.

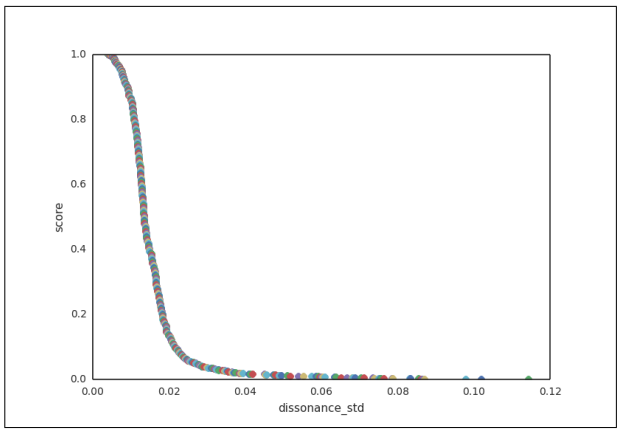


Figure 5. Distribution of scores of flute timbre stability after normalisation.

3.2.3 Models evaluation

In order to evaluate the models we want to check the correlation between the scores and the rankings as we expect the sounds ranked in the first positions to have the highest scores. We evaluate this correlation using Kendall Rank Correlation Coefficient [10], commonly referred as Kendalls tau coefficient. We use the implementation available in the scipy library, that is based on the tau-b version. Its computation, given two rankings x and y of the same size is defined by the following equation:

$$\tau = \frac{(P - Q)}{\sqrt{((P + Q + T) * (P + Q + U))}} \quad (4)$$

where P is the number of concordant pairs, Q the number of discordant pairs, T the number of ties only in x, and U the number of ties only in y. If a tie occurs for the same pair in both x and y, it is not added to either T or U. The values range from -1 to 1, where 1 indicates strong agreement and -1 strong disagreement. We compute τ between the score and the ranking position for all the sounds that are contained in the rankings. The results for each sound attribute and instrument are shown in Table 6.

4. CONCLUSIONS

In this article we presented a web based framework for exploring sound goodness in instrumental sounds using a community of users. The framework provides an easy way to collect sounds and annotations as well as tools to extract and store music descriptors. This allows us to explore the concept of sound goodness in a controlled and flexible environment. Furthermore, the website is useful to the community as a place in which to discuss the issues affecting sound goodness as well as a learning tool to improve their playing techniques. As a way to evaluate the framework we extended our previous work by using annotations from the community collected through a voting task. The models built using this approach provide an automatic rating of goodness for each attribute that tends to match the expert annotations collected in our previous work. The results should improve with more annotations from the community. As future work we want to design new tasks to collect user annotations and build new models according to them.

5. ACKNOWLEDGEMENTS

This research has been partially funded by KORG Inc. The authors would like to thank the entire good-sounds.org community who contributed to the website with sounds and annotations.

Sound attribute	Flute	Violin	Clarinet	Trumpet	Cello	Violin	Alto sax	Baritone sax	Soprano	**Average**
timbre stability	0.37	0.65	0.46	0.38	0.28	0.65	0.33	0.73	0.73	**0.51**
dynamic stability	0.41	0.33	0.24	0.44	0.64	0.33	0.33	0.31	0.31	**0.37**
pitch stability	0.42	0.46	0.22	0.38	0.58	0.46	1	1	0.81	**0.59**
timbre richness	0.04	0.35	0.32	0.11	0.21	0.35	1	0.56	1	**0.43**
attack clarity	0.33	0.59	0.38	0.3	0.18	0.59	0	0.34	0.35	**0.34**

Table 6. Kendall tau coefficient between the scores and the rankings of each sound attribute.

6. REFERENCES

[1] J. Geringer and C. Madsen. "Musicians ratings of good versus bad vocal and string performances," *Journal of Research in Music Education*, vol. 46, pages 522-534, 1998.

[2] Brian E. Russell. "An empirical study of a solo performance assessment model," *International Journal of Music Education*, vol. 33, pages 359-371, 2015.

[3] O. Roman Picas, H. Parra Rodriguez, D. Dabiri, H. Tokuda, W. Hariya, K. Oishi, and X. Serra. "A real-time system for measuring sound goodness in instrumental sounds," *Audio Engineering Society Convention 138.* Audio Engineering Society 2015.

[4] F. Font, G. Roma, and X. Serra. "Freesound technical demo," *Proceedings of the 21st ACM international conference on Multimedia*, 2013.

[5] D. Bogdanov, N. Wach, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. R. Zapata, and X. Serra. "Essentia: An audio analysis library for music information retrieval," *Proceedings of the International Society for Music Information Retrieval Conference,* pages 493-498, 2013.

[6] A. de Cheveign and H. Kawahara. "YIN, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America,* pages 111-1917, 2002.

[7] R. J. McNab, Ll. A. Smith, and I. H. Witten. "Signal processing for melody transcription," *Australasian Computer Science Communications 18*, pages 301-307, 1996.

[8] P. M. Brossier. "Automatic Annotation of Musical Audio for Interactive Applications," *Ph.D. Thesis, Queen Mary University of London, UK*, 2007.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, and J. Vanderplas. "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research 12*, pages 2825-2830, 2011.

[10] Stepanov, Alexei. "On the Kendall Correlation Coefficient," arXiv preprint arXiv:1507.01427, 2015.

IMPROVING PREDICTIONS OF DERIVED VIEWPOINTS IN MULTIPLE VIEWPOINT SYSTEMS

Thomas Hedges
Queen Mary University of London
t.w.hedges@qmul.ac.uk

Geraint Wiggins
Queen Mary University of London
geraint.wiggins@qmul.ac.uk

ABSTRACT

This paper presents and tests a method for improving the predictive power of derived viewpoints in multiple viewpoints systems. Multiple viewpoint systems are a well established method for the statistical modelling of sequential symbolic musical data. A useful class of viewpoints known as derived viewpoints map symbols from a basic event space to a viewpoint-specific domain. Probability estimates are calculated in the derived viewpoint domain before an inverse function maps back to the basic event space to complete the model. Since an element in the derived viewpoint domain can potentially map onto multiple basic elements, probability mass is distributed between the basic elements with a uniform distribution. As an alternative, this paper proposes a distribution weighted by zero-order frequencies of the basic elements to inform this probability mapping. Results show this improves the predictive performance for certain derived viewpoints, allowing them to be selected in viewpoint selection.

1. INTRODUCTION

Multiple viewpoint systems [7] are an established statistical learning approach to modelling multidimensional sequences of symbolic musical data. Music is presented as a series of events comprising of basic attributes (e.g. pitch, duration) modelled by a collection of *viewpoints*. For example, pitch may be modelled by pitch interval, pitch class, or even pitch itself. Statistical structure for each viewpoint is captured with a Markovian approach, usually in the form of a Prediction by Partial Match (PPM) [2] suffix tree. Predictions from different viewpoints modelling the same basic attribute are combined, weighting towards viewpoints with lower uncertainty in terms of Shannon entropy [24]. The system can be viewed as a mixture of experts, or ensemble method machine learning approach to symbolic music, dynamically using specialised models which are able to generalise data in order to find structure.

The current research explores a problem associated with a collection of viewpoints known as *derived viewpoints*. Derived viewpoints apply some function to basic attributes aiming to capture some relational structure between basic attributes (e.g. pitch interval), or to generalise sparse data (e.g. pitch class). During training, elements from the basic attribute domain are mapped onto the derived viewpoint domain with a surjective function. Viewpoint models must be combined over a shared alphabet in order to calculate probability estimates, therefore, an inverse function maps from the derived viewpoint domain to the basic attribute domain. Where a derived element maps onto several basic elements, probability mass from the derived element is distributed uniformly between the basic elements [17]. This can be problematic for derived viewpoints with small domains mapping onto large basic attribute domains as the derived elements could refer to many basic elements. Such viewpoints may generalise sparse data and find useful statistical structure, but this information is lost when mapping back to the basic attribute domain. This is especially prevalent where the zero-order (or unigram) distribution of the basic attribute domain is of low entropy, such that a few elements are very frequent and the rest relatively infrequent.

This paper proposes a method for improving predictions from derived viewpoints. The basic premise behind the method is to use the zero-order distribution of the basic attribute to weight the probabilities from the derived viewpoint when mapping back to the basic attribute. This enables the derived viewpoint to take advantage of the zero order statistics of basic attributes in a way which is not possible if the basic and derived viewpoints are modelled separately. After a review of research using multiple viewpoint systems (Section 2), the system used in the current paper is presented (Section 3), and a detailed description of the proposed method given (Section 4). The method is tested on individual derived viewpoints (Section 5.1) before being applied to various full multiple viewpoint systems, including viewpoint selection (Section 5.2).

2. RELATED RESEARCH

Multiple viewpoint systems have become an important tool for statistical learning of music since their inception over twenty-five years ago [3]. This section reviews their uses and applications to both musical and non-musical domains.

Early multiple viewpoint systems [3, 7, 16] focussed on monophonic melodic music, namely chorale and folksong melodies. The seminal paper [7] uses hand-constructed multiple viewpoint systems with a corpus of 100 Bach chorales. Results show that a system of four viewpoints

© Thomas Hedges, Geraint Wiggins. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Thomas Hedges, Geraint Wiggins. "Improving Predictions of Derived Viewpoints in Multiple Viewpoint Systems", 17th International Society for Music Information Retrieval Conference, 2016.

capturing pitch, sequential pitch interval, scale degree, durational, and metrical information performs best. The system can be used as a generative tool, using a random walk process to generate a chorale in the style of the training corpus. Further work with monophonic melodic music can be seen with the Information Dynamics of Music (IDyOM) model [16], which is developed as a cognitive model of melodic expectation. The PPM* algorithm is refined [20] with a thorough evaluation of smoothing methods, as well as the methods for combining predictions from various individual models, and the method for constructing viewpoint systems [17]. IDyOM is found to closely correlate with experimental data of melodic expectation from human participants, accounting for 78% of variance when predicting notes in English hymns [19], and 83% of variance for British folksongs [21]. Multiple viewpoint systems have also been applied successfully to Northern Indian raags [25], Turkish folk music [23], and Greek folk tunes [6], strengthening their position as a general, domain-independent statistical learning model for music.

Multiple viewpoint systems can be applied to polyphonic musical data, modelling some of the harmonic aspects of music. Musical data with multiple voices is divided into vertical slices [4] representing collections of simultaneous notes, i.e. chords. Relationships between voices can be captured with the use of *linked viewpoints* between voices. This approach has been utilised extensively for the harmonisation of four-part chorales [27, 28]. Harmonic structure can also be modelled directly from chord symbols [5, 10, 22], removing the problems of sparsity and equivalence associated with chord voicing.

Strong probabilistic models of expectation for sequential data can be used for segmentation and chunking. IDyOM is compared to rule-based models for boundary detection in monophonic melodic music in [18], with the statistical model performing comparably rule-based systems. Similar methods have been applied to segmenting natural language at the phoneme and morpheme level [9]. These segmentation studies utilise the fact that certain information theoretic properties, namely information content, can be used to predict boundaries in sequences. The ability for multiple viewpoint systems to model the information theoretic properties of sequences, as well as their general approach to statistical learning, makes them an attractive basis for cognitive architectures capable of general learning, finding higher order structure, and computational creativity [29].

3. A MULTIPLE VIEWPOINT SYSTEM FOR CHORD SEQUENCES

This section presents a brief technical description of the multiple viewpoint system and corpus used in the current research. The corpus consists of 348 chord sequences from jazz standards in lead sheet format from *The Real Book* [11] compiled by [15]. This gives a suitably large corpus of 15,197 chord events, represented as chord symbols (e.g. Dm^7, $Bdim$, $G7$). *The Real Book* is core jazz repertoire comprising of a range of composers and styles, indicating it is a good candidate for studying tonal jazz harmony. The viewpoint pool is derived from similar multiple viewpoint systems dealing with chord symbol sequences [5, 10].

3.1 Harmonic Viewpoints

Three basic attributes, `Root`, `ChordType`, and `PosInBar`, are used to represent chord labels. `Root` is the functional root of the chord as a pitch class assuming enharmonic equivalence. `ChordType` represents the quality of the chord (e.g. major, minor seventh) and are simplified to a set of 13 (*7, M, m7, m, 6, m6, halfdim, dim, aug, sus, alt, no3rd, NC*) for practical reasons. [1] NC represents the special case where no harmonic instruments are instructed to play in the score. `PosInBar` represents the metrical position in the current bar measured in quavers. Since, by definition, a chord must be stated at the start of each bar, this is a sufficient basic attribute to represent any durational or temporal information in the chord sequence.

The following viewpoints are derived from `Root`. `RootInt` is the root interval in semitones modulo-12 between two adjacent chords, returning the symbol -1 if either is NC. `MeeusInt` categorises root movement (`RootInt`) using root progression theories [14]. The symbol 1 represents dominant root progressions (`RootInt` = 1,2,5,8,9), -1 for subdominant progressions (`RootInt` = 3,4,7,10,11), 0 for no root movement (`RootInt` = 0), -2 for a diminished fifth (`RootInt` = 6), and -3 when either root is NC. Since tonal harmony progresses predominantly in perfect fifths, the `ChromaDist` viewpoint simply represents the minimum number of perfect fifths required to get from one root to the next, or the smallest distance around a cycle of fifths, with -1 representing the NC case. All of these viewpoints return the undefined symbol, $\perp$, for the first event of a piece when the previous event does not exist. `RootIntFiP`, `MeeusIntFiP`, and `ChromaDistFiP`, apply `RootInt`, `MeeusInt` and `ChromaDist` to the current event and the first event of the piece instead of the previous event. Finally, a *threaded viewpoint* (see [7]), `RootInt` $\ominus$ `FiB`, measures `RootInt` between chords on the first beats of successive bars.

Three viewpoints are derived from `ChordType`, allowing chord types to be categorised in a number of ways. `MajType` assigns a 1 to all chords where the third is major, a 2 to all chords where the third is minor and a 0 to all chords without a third. `7Type` assigns a 1 to all chords with a minor 7th, and a 0 to all other chords, (except a *NC* which is given a -1 symbol.) `FunctionType` assigns all chords with a major third and minor seventh a 0 (dominant chords), all other chords with a major third a 1 (major tonics), all chords with a minor third and minor seventh a 2 (pre-dominant), all other minor chords a 3 (minor tonic), and *NC* a -1. Table 1 summarises all of the harmonic viewpoints presented in this section over a sample chord sequence.

[1] See [10] for a detailed explanation of chord type simplification.

	Bm^7	D^7	NC	GM^7
Root	11	2	-1	7
ChordType	min	7	NC	maj
PosInBar	0	0	2	0
RootInt	$\perp$	3	-1	-1
MeeusInt	$\perp$	-1	-3	-3
ChromaDist	$\perp$	3	-1	-1
RootIntFiP	$\perp$	3	-1	8
MeeusIntFiP	$\perp$	-1	-3	1
ChromaDistFiP	$\perp$	3	-1	4
RootInt $\ominus$ FiB	$\perp$	3	$\perp$	5
MajType	0	1	-1	1
7Type	1	1	-1	0
FunctionType	2	0	-1	1

Table 1. Sample chord sequence with basic and derived viewpoints.

3.2 System Description

A fully detailed model description is beyond the scope of this paper, however, broadly the system follows the IDyOM model [16], branching from the publicly available LISP implementation [1]. The system estimates probabilities of sequences of events in a basic event space ξ with viewpoints, τ, operating over sequences formed from elements of a viewpoint alphabet $[\tau]$. Formally, a viewpoint modelling a type τ comprises of a partial function $\Psi_\tau : \xi^* \rightharpoonup [\tau]$, a *type set* $\langle\tau\rangle$ specifying the basic types the viewpoint is capable of predicting, and a PPM* model trained from sequences in $[\tau]$. In order to make predictions over the basic event space ξ, symbols are converted back from $[\tau]$ with the inverse function $\Psi' : \xi^* \times [\tau] \rightarrow 2^{[\tau_b]}$ where τ_b is the basic type associated by τ. This many-to-one mapping means that a single derived sequence can represent multiple basic event sequences.

Long-term (LTM) and short-term (STM) models [7] are used to capture both the general trends of the style modelled and the internal statistical structure of the piece being processed. An LTM consists of the full training set, whilst the STM is built incrementally from the current piece and is discarded after it has been processed. Predictions from all viewpoints within the LTM/STM are combined first, before combining the LTM and STM predictions. Prediction combination is achieved with a weighted geometric mean [17], favouring the least uncertain models according to their Shannon entropy. [2] Various smoothing methods are employed, allowing novel symbols to be predicted and predictions from different length contexts to be combined in a meaningful way without assuming a fixed order bound [20].

Multiple viewpoint systems are assessed quantitatively with methods from information theory [13]. The main performance measure is mean information content $\bar{h}$, representing the number of bits required on average to represent each symbol in the sequence of length J (1).

[2] For reference, all model combinations in this paper are achieved with an LTM-STM bias of 7 and a viewpoint bias of 2 see [17] for details.

$$\bar{h}(e_1^J) = -\frac{1}{J}\sum_{i=1}^{J} log_2\, p(e_i \mid e_{i-n+1}^{i-1}) \quad (1)$$

4. USING ZERO-ORDER STATISTICS TO WEIGHT Ψ'

The focus of this paper is to improve predictions from derived viewpoints by weighting probabilities after the inverse mapping function Ψ' has been applied. Firstly, it is useful to show in detail cases where certain derived viewpoints would be poor predictors for a basic attribute.

Where a derived viewpoint maps an element onto a large number of basic elements, a certain amount of information is lost by dividing the probability mass uniformly. Suppose a prediction from MajType returns a high probability for a major chord, mapping onto a *'7'*, *'M7'*, *'6'*, *'alt'* or *'aug'* ChordType. *'7'* and *'M7'* chords are very common, whilst *'alt'* and *'aug'* chords are comparatively rare. Since MajType must distribute probability mass equally to all five of these basic elements, a considerable amount of information is lost and it remains a poor predictor of ChordType. The predictive strength of these kinds of viewpoints are to generalise data which will become sparse, specifically in sequence prediction when matching contexts in the PPM* model. This strength is likely to be reduced by the uniform distribution of probability mass and could make these viewpoints poor predictors; returning high mean information content estimates and remaining unselected in viewpoint selection.

A general approach to counter this loss of information is to weight probabilities with the zero-order (unigram) frequencies when distributing probability mass from a derived element to the relevant basic elements. For reference, (2) shows a probability estimate of a basic element, $p(t_{\tau_b})$, calculated by uniformly distributing the probability mass of a derived element, $p(t_\tau)$, following [17]. B represents the set of basic elements that are mapped onto from the derived element t_τ. The proposed alternative, shown in (3), uses probabilities from the zero-order model $p_0(t_{\tau_b})$ to weight the distribution of probability mass from t_τ to t_{τ_b}. As with PPM* predictions, probability mass must be reserved for unseen symbols in the basic element alphabet, so a smoothing method and -1^{th} order distribution is utilised. Using an established smoothing framework [20], (4) shows an interpolated smoothing method with escape method C, an order bound of 0 and with no update exclusion. $c(t_{\tau_b})$ is the number of times the symbol t_{τ_b} occurs the training set, J is the length of the training set, $[\tau_b]$ is the alphabet of the basic viewpoint, and $[\tau_b]^s$ the observed alphabet of the basic viewpoint.

$$p(t_{\tau_b}) = \frac{p(t_\tau)}{|B|} \quad (2)$$

$$p_w(t_{\tau_b}) = p(t_\tau)\frac{p_0(t_{\tau_b})}{\sum_{i\in B} p_0(i)} \quad (3)$$

$$p_0(t_{\tau_b}) = \frac{c(t_{\tau_b})}{J + |[\tau_b]^s|} + ... \\ \frac{|[\tau_b]^s|}{J + |[\tau_b]^s|} \cdot \frac{1}{|[\tau_b]| + 1 - |[\tau_b]^s|} \quad (4)$$

A demonstration of this process is shown in Figure 1. `FunctionType` is used to predict the next `ChordType` symbol with an LTM model given the context *Am7, D7, Bm7, Bbm7*. The top chart shows a strong expectation of a pre-dominant chord which could map onto a *m7*, *halfdim*, or *dim* `ChordType`. With an unweighted Ψ' (2) from `FunctionType` to `ChordType`, these three basic elements are all given equal probability (middle chart). However, since *m7* is far more common than *halfdim* and *dim*, a more accurate probability distribution could be one weighted (3) by the zero-order frequencies (bottom chart), assigning a high probability to *m7*. This approach allows the powerful generalisation of derived viewpoint models to be combined efficiently with more specific predictions from the basic viewpoint.

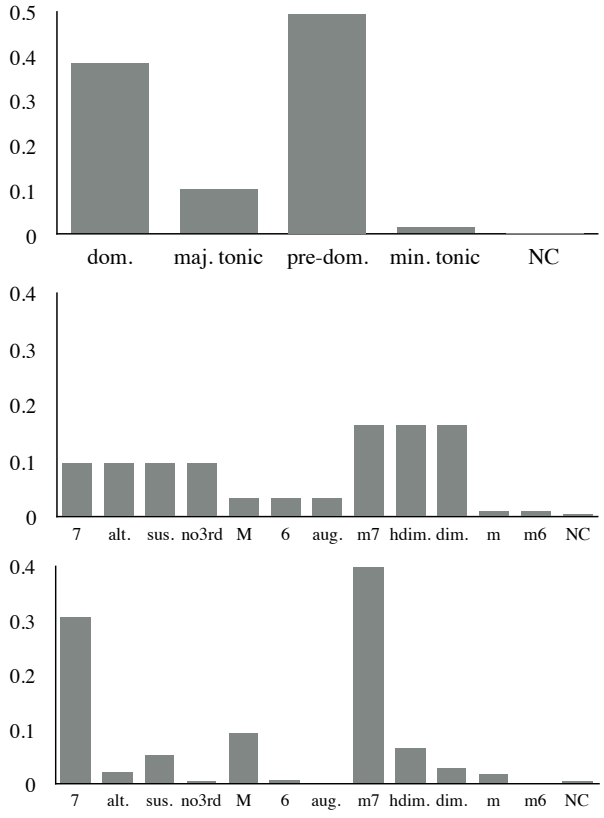


Figure 1. Top: probability distribution of `FunctionType` following the context *Am7, D7, Bm7, Bbm7*. Middle and bottom: probability distributions for `ChordType` predicted by `FunctionType` with an unweighted (middle) and zero-order weighted Ψ' (bottom).

5. TESTING THE IMPACT OF WEIGHTING Ψ'

To investigate the effect of weighting Ψ'_τ with a zero order model, the mean information content, $\bar{h}$ (1), is used as a performance metric to compare predictions with the weighted and unweighted inverse mapping function. In all cases, $\bar{h}$ is calculated with a 10-fold cross-validation of the corpus. The effect of the weighting on individual derived viewpoints is observed first (Section 5.1) before comparing the impact on full multiple viewpoint systems (Section 5.2). The STM is an unbounded interpolated smoothing model with escape method D using update exclusion, and the LTM an unbounded interpolated smoothing model with escape method C without update exclusion [20]. These parameters have been found to be optimal for the current corpus [10].

For the individual viewpoints, it is expected that derived viewpoints which abstract heavily from their basic viewpoint will benefit most from weighting Ψ'. Typically, these are viewpoints derived from `ChordType`, for example, `MajType` reduces the alphabet of `ChordType` from 13 down to 3. By contrast, it is expected that the impact of weighting Ψ' will be far smaller for derived viewpoints with a close to one-to-one mapping between alphabets (e.g. `RootInt`), if significant at all. When constructing a full multiple viewpoint system it is hoped that weighting Ψ' will help more derived viewpoints to be selected over basic viewpoints. Not only should this give a lower mean information content, but also produce a more compact viewpoint model. Successful derived viewpoints should abstract information away from basic viewpoints onto smaller alphabets without a loss in performance.

5.1 Individual Viewpoints Results

Six derived viewpoints for predicting `Root` and `ChordType` are chosen for testing, as well as the basic viewpoints themselves for reference. Table 2 shows the mean information content calculated using both weighted and unweighted Ψ' functions. Effect size measured by Cohen's $d = \frac{\bar{h}_1 - \bar{h}_2}{\sigma_{pooled}}$ across all pieces $(n = 348)$ is used to quantify the relative performance for each viewpoint. A one-sided paired *t*-test across pieces assesses statistical significance between the means at the $p < .001$ level, marked with a *.

Strikingly, the derived viewpoints predicting `ChordType` benefit most from the weighting method, all with effect sizes greater than 1.7 and an absolute improvement of around 0.9 bit/symbol. By contrast, the impact of the weighting on the viewpoints derived from `Root` is small and inconsistent, with effect sizes of around 0.1 or less. Indeed, weighting Ψ' has a marginally negative impact on `RootInt`, although only by 0.016 bits/symbol. It is likely that this is because in the majority of cases `RootInt` has a one-to-one mapping with `Root`, except for the *NC* case where a `RootInt` symbol of -1 maps onto the full alphabet of `Root`. It is interesting to note that none of the individual derived viewpoints are able to predict their basic viewpoint better than the

Derived Viewpoint	Unweighted Ψ'	Weighted Ψ'	d
ChordType	1.807	1.807	.000
MajType	3.270	2.315	1.977*
7Type	3.249	2.371	1.766*
FunctionType	3.060	2.080	1.731*
Root	2.259	2.259	.000
RootInt	2.297	2.313	-.030
MeeusInt	3.152	3.076	.129*
ChromaDist	2.688	2.681	.009

Table 2. Predicting ChordType (top) and Root (bottom) with weighted and unweighted Ψ'. Performance difference is measured by Cohen's $d = \frac{\bar{h}_1 - \bar{h}_2}{\sigma_{pooled}}$. * marks differences which are statistically significant at the $p <.001$ level according to a one-sided paired t-test.

basic viewpoint itself, even with a weighted Ψ'. At this point their impact on full multiple viewpoint systems is unknown and must be tested with a viewpoint selection algorithm.

5.2 Viewpoint Selection Results

A viewpoint selection algorithm is a search algorithm to find the locally optimal multiple viewpoint system given a set of candidate viewpoints. Following [17], the current research uses a forward stepwise algorithm which, starting from the empty set of viewpoints, alternately attempting to add and then delete viewpoints from the current system, greedily selecting the best system according to $\bar{h}$ at each iteration. For this study a stopping criteria is imposed such that the new viewpoint system must improve $\bar{h}$ by at least an effect size of $d > .005$, or more than 0.5% of a standard deviation.

Predicting the Root and ChordType together, given the metrical position in the bar (PosInBar), is chosen as a cognitively tangible task for the multiple viewpoint system to perform. In order to predict the two basic attributes simultaneously they are considered as the merged attribute Root$\otimes$ChordType. Merged attributes are simply a cross product of basic attributes, equivalent to *linked viewpoints* [7], and have been found to be an effective method for predicting multiple highly correlated basic attributes [10]. An unbounded interpolated smoothing model with escape method C for both STM and LTM is found to be optimal for predicting merged attributes in the current corpus [10], with update exclusion used in the STM only. Using all of the basic and derived viewpoints specified in Section 3.1 and allowing linked viewpoints consisting of up to two constituent viewpoints, or three if one is PosInBar, a pool of 64 candidate viewpoints for selection is formed.

The unweighted Ψ' system goes through five iterations of viewpoint addition (without deletion) before termination returning $\bar{h} = 3.037$ (Figure 2). By contrast, the weighted Ψ' system terminates after seven viewpoint additions with a lower $\bar{h}$ of 3.012 (Figure 3). The difference between these results is found to be statistically significant with a paired one-sided *t*-test at the .001 level ($df = 347$ $t = 5.422$ $p < .001$). However, more importantly, the effect size is found to be small, $d = .026$, owing to the absolute different of .025 bits/symbols between the means. Since the termination criteria is somewhat arbitrary (an appropriate value for d is hand-selected), the unweighted system was allowed to continue up to seven iterations to match the weighted system. This returns $\bar{h} = 3.025$, which is still found to be significantly outperformed by the weighted model ($df = 347$ $t = 3.725$ $p < .001$, effect size $d = .017$).

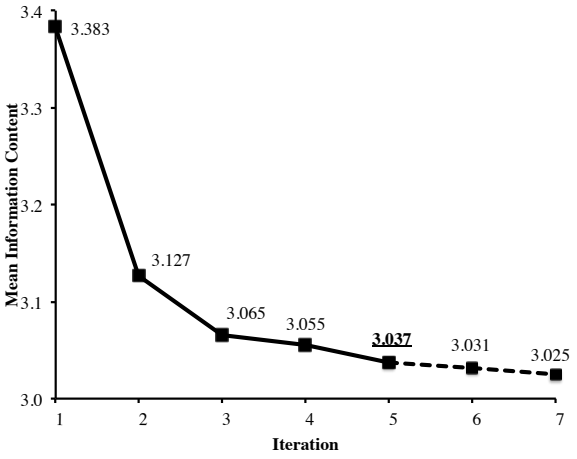
In the context of the current study the viewpoints chosen from both viewpoint selection runs is highly relevant. The unweighted Ψ' selects only basic viewpoints and viewpoints derived from Root. No viewpoints derived from ChordType are selected, nor MeeusInt or ChromaDist. This is to be expected given the findings in Section 5.1, where derived viewpoints with an unweighted Ψ' are found to be poor predictors of ChordType. By contrast, during viewpoint selection with a weighted Ψ', linked viewpoints containing FunctionType are added on the third and sixth iterations and MeeusInt on the fourth iteration. This means that not only does the weighted Ψ' model perform slightly better in terms of $\bar{h}$, but is also more compact since the average viewpoint alphabet size of the seven linked viewpoints selected is 124.4, as opposed to 169 for the unweighted Ψ' model. [3]

6. CONCLUSIONS AND DISCUSSION

This paper has presented a new method for improving predictions from derived viewpoints by weighting Ψ' (the function which maps from the derived to basic alphabet of a viewpoint) with the zero-order frequencies of the basic attribute. Results show that such a weighting significantly improves the performance of derived viewpoints which abstract heavily away from their basic viewpoint, notably MajType, 7Type, and FunctionType. On the other hand, viewpoints derived from Root, such as RootInt, MeeusInt, and ChromaDist, see only marginal improvements or slight decreases in performance. It has been shown that weighting Ψ' allows more derived viewpoints to be chosen in viewpoint selection. This produces a model which returns a slightly lower mean information content than its unweighted counterpart. This model is also slightly more computationally efficient owing to the smaller alphabet sizes of the selected viewpoints. In practical terms, this creates a model that has a closer fit to the training data whilst taking slightly less time to run for any of the tasks outlined in Section 2 (computational modelling of expectation, segmentation, and automatic music generation).

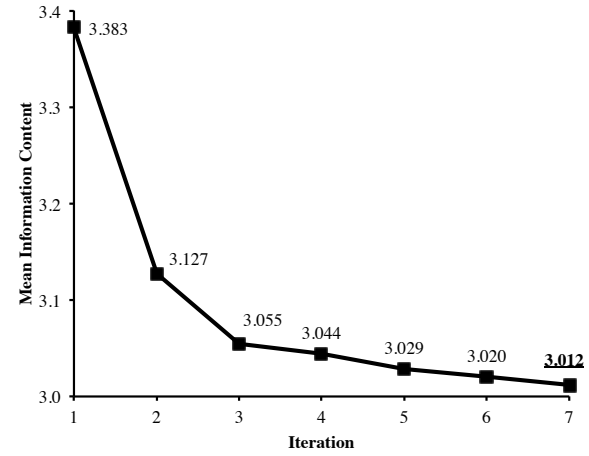
This paper studied weighting only by zero-order frequency. Useful future research might explore alternative weighting schemes beyond the zero-order frequencies, such as first-order Markov, or even more aggressive, exponential weighting schemes. Furthermore applying the

[3] Note that PosInBar is a given attribute and so contributes an alphabet size of only 1 during the prediction phase.



$1 +$ `Root` $\otimes$ `ChordType` $\otimes$ `PosInBar`
$2 +$ `RootInt` $\otimes$ `ChordType`
$3 +$ `RootIntFiP` $\otimes$ `ChordType` $\otimes$ `PosInBar`
$4 +$ `Root` $\otimes$ `ChordType`
$5 +$ `RootInt` $\otimes$ `ChordType` $\otimes$ `PosInBar`
$(6 +$ `RootIntFiP` $\otimes$ `ChordType`$)$
$(7 +$ `RootInt` $\ominus$ `FiB` $\otimes$ `ChordType`$)$

Figure 2. Viewpoint selection for multiple viewpoint models using unweighted Ψ'. Viewpoints added at each iteration are shown below the graph. Bracketed viewpoints and the dotted line indicate viewpoints added after termination.



$1 +$ `Root` $\otimes$ `ChordType` $\otimes$ `PosInBar`
$2 +$ `RootInt` $\otimes$ `ChordType`
$3 +$ `RootIntFiP` $\otimes$ `FunctionType` $\otimes$ `PosInBar`
$4 +$ `MeeusInt` $\otimes$ `ChordType` $\otimes$ `PosInBar`
$5 +$ `RootIntFiP` $\otimes$ `ChordType`
$6 +$ `RootInt` $\otimes$ `FunctionType` $\otimes$ `PosInBar`
$7 +$ `Root` $\otimes$ `ChordType`

Figure 3. Viewpoint selection for multiple viewpoint models using weighted Ψ'. Viewpoints added at each iteration are shown below the graph.

weighting schemes to a range of domains, genres, and corpora beyond jazz harmony is necessary to prove the methods presented in this paper can be universally applied.

The weighting of Ψ' for derived viewpoints appears to be successful as it combines a more general, abstracted model capable of finding statistical regularities with the more fine-grained model of the basic viewpoint. It could be argued that this is already achieved by multiple viewpoint systems in that they combine predictions from multiple models at various levels of abstraction in an information-theoretically informed manner. However, if the effect of weighting Ψ' with a zero-order model was entirely subsumed by viewpoint combination then almost identical viewpoints would be chosen during the viewpoint selection process, which is not the case (Section 5.2). As the results stand, the weighted Ψ' model selects more derived viewpoints, forming a more compact model and performs slightly better in terms of mean information content.

The compactness of multiple viewpoint systems is relevant both to computational complexity and their relationship with cognitive representations. Searching a suffix tree for the PPM* algorithm with the current implementation using Ukkonen's algorithm [26] is achieved in linear time (to the size of the training data J), but must be done $|[\tau]|$ times to return a complete prediction set over the viewpoint alphabet $[\tau]$, giving a time complexity of $O(J|[\tau]|)$. Selecting viewpoints with a smaller alphabet size has, therefore, a substantial impact on the time complexity for the system. As a model for human cognition [19], selecting viewpoints with smaller alphabets without a loss of performance is equivalent to building levels of abstraction when learning cognitive representations [29].

Additionally, the weighted Ψ' model constructs more convincing viewpoint systems from a musicological perspective. Chord function is an important aspect of jazz music [12] and tonal harmony in general, where common cadences progress in *pre-dominant, dominant, tonic,* patterns. Therefore, the fact that `ChordType` is selected over `MajType` and `7Type` suggests that chord function as signified by the third and seventh of the chord together is more important than the quality of the third (modelled by `MajType`) or seventh (modelled by `7type`) separately. Similarly, the selection of `MeeusInt` in the model suggests that functional theories for root progressions may be useful descriptors of tonal harmony. On the other hand, `ChromaDist`, which considers rising and falling progressions by a perfect fifth equivalent, is not selected. This supports the notion that harmonic progressions in tonal harmony are goal-oriented and strongly directional [8].

7. ACKNOWLEDGEMENTS

The authors would like to thank Marcus Pearce for the use of the IDyOM software. This work is supported by the Media and Arts Technology programme, EPSRC Doctoral Training Centre EP/G03723X/1.

8. REFERENCES

[1] `https://code.soundsoftware.ac.uk/projects/idyom-project`. Accessed: 23-03-2016.

[2] J. Cleary and I. Witten. Data compression using adaptive coding and partial string matching. *Communications, IEEE Transactions on*, 32(4):396–402, 1984.

[3] D. Conklin. *Prediction and Entropy of Music*. PhD thesis, Department of Computer Science, University of Calgary, 1990.

[4] D. Conklin. Representation and discovery of vertical patterns in music. In *IMCAI*, pages 32–42, Edinburgh, Scotland, 2002. Springer.

[5] D. Conklin. Discovery of distinctive patterns in music. *Intelligent Data Analysis*, 14(5):547–554, 2010.

[6] D. Conklin and C. Anagnostopoulou. Comparative Pattern Analysis of Cretan Folk Songs. In *3rd International Workshop on Machine Learning and Music*, pages 33–36, Florence, Italy, 2010.

[7] D. Conklin and I. Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.

[8] C. Dahlhaus. *Studies on the Origin of Harmonic Tonality*. Princeton University Press, Princetown, NJ, 1990.

[9] S. Griffiths, M. Purver, and G. Wiggins. From phoneme to morpheme: A computational model. In *6th Conference on Quantitative Investigations in Theoretical Linguistics*, Tübingen, Germany, 2015.

[10] T. Hedges and G. Wiggins. The prediction of merged attributes with multiple viewpoint systems. *Journal of New Music Research*, accepted.

[11] H. Leonard. *The Real Book: Volume I, II, III, IV and V*. Hal Leonard, Winoa, MN, 2012.

[12] M. Levine. *The Jazz Theory Book*. Sher Music Co., Petaluma, CA, 1995.

[13] D. Mackay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2003.

[14] N. Meeus. Toward a post-schoenbergian grammar of tonal and pre-tonal harmonic progressions. *Music Theory Online*, 6(1):1–8, 2000.

[15] F. Pachet, J. Suzda, and D. Martín. A comprehensive online database of machine-readable leadsheets for jazz standards. In *14th International Society for Music Information Retrieval Conference*, pages 275–280, Curitiba, Brazil, 2013.

[16] M. Pearce. *The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Composition*. PhD thesis, City University, London, 2005.

[17] M. Pearce, D. Conklin, and G. Wiggins. Methods for combining statistical models of music. In *CMMR'04: Proceedings of the Second International Conference on Computer Music Modeling and Retrieval*, pages 295–312. Springer-Verlag, 2005.

[18] M. Pearce, D. Mullensiefen, and G. Wiggins. The role of expectation and probabilistic learning in auditory boundary perception: A model comparison. *Perception*, 39(10):1365–1389, 2010.

[19] M. Pearce, M. Ruiz, S. Kapasi, G. Wiggins, and J. Bhattacharya. Unsupervised statistical learning underpins computational, behavioural, and neural manifestations of musical expectation. *NeuroImage*, 50(1):302–313, 2010.

[20] M. Pearce and G. Wiggins. Improved methods for statistical modelling of monophonic music. *Journal of New Music Research*, 33(4):367–385, 2004.

[21] M. Pearce and G. Wiggins. Expectation in melody: the influence of context and learning. *Music Perception: An Interdisciplinary Journal*, 23(5):377–405, 2006.

[22] M. Rohrmeier and T. Graepel. Comparing feature-based models of harmony. In *9th International Symposium on Computer Music Modeling and Retrieval (CMMR 2012)*, pages 357–370, London, UK, 2012.

[23] S. Sertan and P. Chordia. Modeling Melodic Improvisation in Turkish Folk Music Using Variable-Length Markov Models. In *12th International Society for Music Information Retrieval Conference*, pages 269–274, Miami, FL, 2011.

[24] C. Shannon. A Mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[25] A. Srinivasamurthy and P. Chordia. Multiple viewpoint modeling of north Indian classical vocal compositions. In *International Symposium on Computer Music Modeling and Retrieval*, pages 344–356, London, 2012.

[26] E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, 14(3):249–260, 1995.

[27] R. Whorley. *The Construction and Evaluation of Statistical Models of Melody and Harmony*. PhD thesis, Goldsmiths, University of London, London, 2013.

[28] R. Whorley, G. Wiggins, C. Rhodes, and M. Pearce. Multiple viewpoint systems: time complexity and the construction of domains for complex musical viewpoints in the harmonization problem. *Journal of New Music Research*, 42(3):237–266, 2013.

[29] G. Wiggins and J. Forth. IDyOT: A computational theory of creativity as everyday reasoning from learned information. In *Computational Creativity Research: Towards Creative Machines*, pages 127–148. Atlantis Press, 2015.

KNOWN-ARTIST LIVE SONG ID: A HASHPRINT APPROACH

TJ Tsai[1] **Thomas Prätzlich**[2] **Meinard Müller**[2]
[1]University of California Berkeley, Berkeley, CA
[2]International Audio Laboratories Erlangen, Erlangen, Germany
tjtsai@berkeley.edu, thomas.praetzlich,meinard.mueller@audiolabs-erlangen.de

ABSTRACT

The goal of live song identification is to recognize a song based on a short, noisy cell phone recording of a live performance. We propose a system for known-artist live song identification and provide empirical evidence of its feasibility. The proposed system represents audio as a sequence of hashprints, which are binary fingerprints that are derived from applying a set of spectro-temporal filters to a spectrogram representation. The spectro-temporal filters can be learned in an unsupervised manner on a small amount of data, and can thus tailor its representation to each artist. Matching is performed using a cross-correlation approach with downsampling and rescoring. We evaluate our approach on the Gracenote live song identification benchmark data set, and compare our results to five other baseline systems. Compared to the previous state-of-the-art, the proposed system improves the mean reciprocal rank from .68 to .79, while simultaneously reducing the average runtime per query from 10 seconds down to 0.9 seconds.

1. INTRODUCTION

This paper tackles the problem of song identification based on short cell phone recordings of live performances. This problem is a hybrid of exact-match audio identification and cover song detection. Similar to the exact-match audio identification problem, we would like to identify a song based on a short, possibly noisy query. The query may only be a few seconds long, and might be corrupted by additive noise sources as well as convolutive noise based on the acoustics of the environment. Because song identification is a real-time application, the amount of latency that the user is willing to tolerate is very low. Similar to the cover song detection problem, we would like to identify different performances of the same song. These performances may have variations in timing, tempo, key, instrumentation, and arrangement. In this sense, the live song identification problem is doubly challenging in that it inherits the challenges and difficulties of both worlds: it is given a short, noisy query and expected to handle performance variations and to operate in (near) real-time.

© TJ Tsai, Thomas Prätzlich, Meinard Müller. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** TJ Tsai, Thomas Prätzlich, Meinard Müller. "Known-Artist Live Song ID: A Hashprint Approach", 17th International Society for Music Information Retrieval Conference, 2016.

To make this problem feasible, we must reduce the searchable set to a tractable size. One way to accomplish this is shown by the system architecture in Figure 1. When a query is submitted, the GPS coordinates of the cell phone and the timestamp information are used to associate the query with a concert, which enables the system to infer who the musical artist is. Once the artist has been inferred, the problem is reduced to a known-artist search: we assume the artist is known, and we would like to identify which song is being played. The known-artist search is more tractable because it constrains the set of possible songs to the musical artist's studio recordings. In this work, we will focus our attention on the known-artist search.

One important assumption in Figure 1 is that the musical artist or group is popular enough that its concert schedule (dates and locations) can be stored in a database. So, for example, this system architecture would *not* work for an amateur musician performing at a local restaurant. It *would* work for popular artists whose concert schedule is available online.

Exact-match audio identification and cover song detection have both been explored fairly extensively (e.g. [25] [1] [22] [20] [19] [7]). There are several successful commercial applications for exact-match music identification, such as Shazam and SoundHound. Both tasks have benefited from standardized evaluations like the TRECVid content-based copy detection task [13] and the MIREX cover song retrieval task [4]. There have also been a number of works on identifying related musical passages based on query fragments [8] [10] [2], but most of these works assume a fragment length that is too long for a real-time application (10 to 30 seconds). Additionally, these works mostly focus on classical music, where performed works are typically indicated on a printed program and where the audience is generally very quiet (unlike at a rock concert).

In contrast, live song identification based on short cell phone queries is relatively new and unexplored. One major challenge for this task, as with many other tasks, is collecting a suitable data set. Rafii et al. [15] collect a set of cell phone recordings of live concerts for 10 different bands, and they propose a method for song identification based on a binarized representation of the constant Q transform. In this work, we propose an approach based on a binarized representation of audio called hashprints coupled with an efficient, flexible method for matching hashprint sequences, and we explore the performance of such an ap-

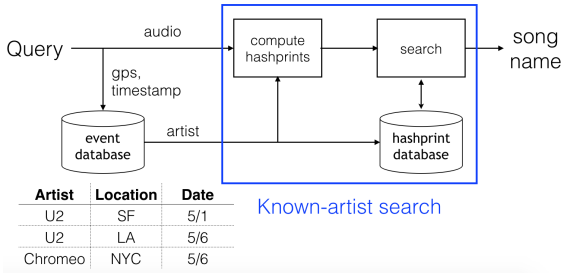


Figure 1. System architecture of the live song identification system. Using GPS and timestamp information, queries are associated with a concert in order to infer the artist.

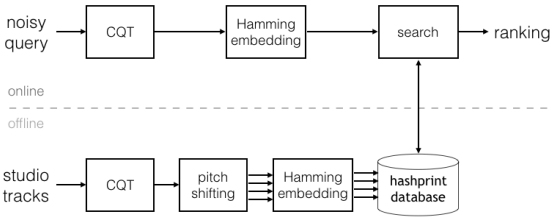


Figure 2. Block diagram for a known-artist search. Multiple pitch-shifted versions of the original studio tracks are considered to handle the possibility that the live performance is performed in a different key.

proach on the live song identification task.

This paper is organized as follows. Section 2 describes the proposed system. Section 3 describes the evaluation of the system. Section 4 presents some additional analyses of interest. Section 5 concludes the work.

2. SYSTEM DESCRIPTION

Figure 2 shows a block diagram of the proposed known-artist search system. There are four main system components, each of which is described below.

2.1 Constant Q Transform

The first main system component is computing a constant Q transform (CQT). The CQT computes a time-frequency representation of audio using a set of logarithmically spaced filters with constant Q-factor. [1] This representation is advantageous for one very important reason: the spacing and width of the filters are designed to match the pitches on the Western musical scale, so the representation is especially suitable for considering key transpositions. In our experiments, we used the CQT implementation described by Schörkhuber and Klapuri [18]. Similar to the work by Rafii et al. [15], we consider 24 subbands per octave between C3 (130.81 Hz) and C8 (4186.01 Hz). To mimic the nonlinear compression of the human auditory system, we compute the log of the subbands' local energies. At the end of this processing, we have 121 log-energy values every 12.4 ms.

2.2 Hamming Embedding

The second main system component is computing a Hamming (binary) embedding. Using a Hamming representation has two main benefits. First, it enables us to store fingerprints very efficiently in memory. In our implementation, we represent each audio frame in a 64-dimensional Hamming space, which allows us to store each hashprint in memory as a single 64-bit integer. Second, it enables us to compute Hamming distances between fingerprints very efficiently. We can compute the Hamming distance between two hashprints by performing a single logical xor operator on two 64-bit integers, and then counting the number of bits in the result. This computation offers significant savings compared to computing the Euclidean distance between two vectors of floating point numbers. These computational savings will be important in reducing the latency of the system.

Our Hamming embedding grows out of two basic principles: compactness and robustness. Compactness means that the binary representation is efficient. This means that each bit should be balanced (i.e. 0 half the time and 1 half the time) and that the bits should be uncorrelated. Note that any imbalance in a bit or any correlation among bits will result in an inefficient representation. Robustness means that each bit should be robust to noise. In the context of thresholding a random variable, robustness means maximizing the variance of the random variable's probability distribution. To see this, note that if the random variable takes a value that is close to the threshold, a little bit of noise may cause the random variable to fall on the wrong side of the threshold, resulting in an incorrect bit. We can minimize the probability of this occurring by maximizing the variance of the underlying distribution. [2]

The Hamming embedding is determined by applying a set of 64 spectro-temporal filters at each frame, and then encoding whether each spectro-temporal feature is increasing or decreasing in time. The spectro-temporal filters are learned in an unsupervised manner by solving the sequence of optimization problems described below. These filters are selected to maximize feature variance, which maximizes the robustness of the individual bits. Consider the CQT log-energy values for a single audio frame along with its context frames, resulting in a $\mathbb{R}^{121w}$ vector, where w specifies the number of context frames. We can stack a bunch of these vectors into a large $\mathbb{R}^{M\times 121w}$ matrix A, where M corresponds (approximately) to the total number of audio frames in a collection of the artist's studio tracks. Let $S \in \mathbb{R}^{121w\times 121w}$ be the covariance matrix of A, and let $x_i \in \mathbb{R}^{121w}$ denote the coefficients of the i^{th} spectro-temporal filter. Then, for $i = 1, \ldots, 64$, we solve the following sequence of optimization problems:

[1] The Q-factor refers to the ratio between the filter's center frequency and bandwidth, so a constant Q-factor means that each filter's bandwidth is proportional to its center frequency.

[2] Since the random variable is a linear combination of many CQT values, the distribution will generally be roughly bell-shaped due to the central limit theorem.

$$\begin{aligned} \text{maximize} \quad & x_i^T S x_i \\ \text{subject to} \quad & \|x_i\|_2^2 = 1 \\ & x_i^T x_j = 0,\ j = 1, \ldots, i-1. \end{aligned} \tag{1}$$

The objective function is simply the variance of the features resulting from filter x_i. So, this formulation maximizes the variance (i.e. robustness) while ensuring that the filters are uncorrelated (i.e. compactness). The above formulation is exactly the eigenvector problem, for which very efficient off-the-shelf solutions exist.

Each bit in the hashprint representation encodes whether the corresponding spectro-temporal feature is increasing or decreasing in time. We first compute delta spectro-temporal features at a separation of approximately one second, and then we threshold the delta features at zero. The separation of one second was determined empirically, and the threshold at zero ensures that the bits are balanced. Note that if we were to threshold the spectro-temporal features directly, our Hamming representation would not be invariant to volume changes (i.e. scaling the audio by a constant factor would change the Hamming representation). Because we threshold on delta features, each bit captures whether the corresponding spectro-temporal feature is increasing or decreasing, which is a volume-invariant quantity.

2.3 Search

The third main system component is the search mechanism: given a query hashprint sequence, find the best matching reference sequence in the database. In this work, we explore the performance of two different search strategies. These two systems will be referred to as hashprint1 and hashprint2 (abbreviated as hprint1 and hprint2 in Figure 3). In both approaches, we compute hashprints every 62 ms and using $w = 20$ context frames. These parameters were determined empirically.

The first search strategy (hashprint1) is a subsequence dynamic time warping (DTW) approach based on a Hamming distance cost matrix. The subsequence DTW is a modification of the traditional DTW approach which allows one sequence (the query) to begin anywhere in the other recording (the reference) with no penalty. One explanation of this technique can be found in [11]. We allow $\{(1,1),(1,2),(2,1)\}$ transitions, which allows live versions to differ in tempo from studio versions by a factor up to two. We perform subsequence DTW of the query with all sequences in the database, and then use the alignment score (normalized by path length) to rank the studio tracks.

The second search strategy (hashprint2) is a cross-correlation approach with downsampling and rescoring. First, the query and reference hashprint sequences are downsampled by a factor of B. For example, when $B = 2$ every other hashprint is discarded. Next, for each reference sequence in the database, we determine the frame offset that maximizes bit agreement between the downsampled query sequence and the downsampled reference sequence. The bit agreement at this offset is used as a match score for the reference sequence. After sorting all of the sequences in the database by their downsampled match score, we identify the top 10 candidate sequences. We then rescore these top 10 candidate sequences using the full hashprint sequence (i.e. without downsampling). Finally, we resort the top 10 candidate sequences based on their refined match score. The resulting ranking is the final output of the system. The advantage of the second search strategy is computational efficiency: we first do a rough scoring of the sequences, and only do a more fine-grained scoring on the top few candidate sequences.

2.4 Pitch Shifting

The fourth main system component is pitch shifting. A band might perform a live version of a song in a slightly different key than the studio version, or the live version may have tuning differences. To ensure robustness to these variations, we considered pitch shifts up to four quarter tones above and below the original studio version. So, the database contains nine hashprint sequences for each studio track. When performing a search, we use the maximum alignment score from the nine pitch-shifted versions as the aggregate score for a studio track. We then rank the studio tracks according to their aggregate scores.

2.5 Relation to Previous Work

It is instructive to interpret the above approach in light of previous work. Using multiple context frames in the manner described above is often referred to as shingling [2] or time delay embedding [20], a technique often used in music identification and cover song detection tasks. It allows for greater discrimination on a single feature vector than could be achieved based only on a single frame. The technique of thresholding on projections of maximum variance is called spectral hashing [26] in the hashing literature. It can be thought of as a variant of locality sensitive hashing [3], where the projections are done in a data-dependent way instead of projecting onto random directions. So, we can summarize our approach as applying spectral hashing to a shingle representation, along with a modification to ensure invariance to volume changes (i.e. thresholding on delta features). This approach was first proposed in an exact-match fingerprinting application using reverse-indexing techniques [23]. Here, instead of using the Hamming embedding to perform a table lookup, we instead use the Hamming distance between hashprints as a metric of similarity in a non-exact match scenario.

There are, of course, many other ways to derive a Hamming embedding. The previous work by Rafii et al. [15] performs the Hamming embedding by comparing each CQT energy value to the median value of a surrounding region in time-frequency. Many recent works have explored Hamming embeddings learned through deep neural network architectures [17] [12], including a recent work by Raffel and Ellis [14] proposing such an approach for matching MIDI and audio files. One advantage of our proposed method is that it learns the audio fingerprint rep-

Artist Name	Genre	# Tracks
Big K.R.I.T.	hip hop	71
Chromeo	electro-funk	44
Death Cab for Cutie	indie rock	87
Foo Fighters	hard rock	86
Kanye West	hip hop	92
Maroon 5	pop rock	66
One Direction	pop boy band	60
Taylor Swift	country, pop	71
T.I.	hip hop	154
Tom Petty	rock, blues rock	193

Table 1. Overview of the Gracenote live song identification data. The database contains full tracks taken from artists' studio albums. The queries consist of 1000 6-second cell phone recordings of live performances (100 queries per artist).

resentation in an unsupervised manner. This is particularly helpful for our scenario of interest, since collecting noisy cell phone queries and annotating ground truth is very time-consuming and labor-intensive. Our proposed method also has the benefit of requiring relatively little data to learn a reasonable representation. This can be helpful if, for example, the artist of interest only has tens of studio tracks. In such cases, a deep auto-encoder [9] may not have sufficient training data to converge to a good representation. So, our method straddles two different extremes: it is adaptive to the data (unlike the fixed representation proposed in [15]), but it works well with small amounts of data (unlike representations based on deep neural networks).

3. EVALUATION

We will describe the evaluation of the proposed system in three parts: the data, the evaluation metric, and the results.

3.1 Data

We use the Gracenote live song identification data set. This is a proprietary data set that is used for internal benchmarking of live song identification systems at Gracenote. The data comes from 10 bands spanning a range of genres, including rock, pop, country, and rap. There are two parts to the data set: the database and the queries. The database consists of full tracks taken from the artists' studio albums. Table 1 shows an overview of the database, including a brief description of each band and the number of studio tracks. Note that the number of tracks per artist ranges from 44 (for newer groups like Chromeo) up to 193 (for very established musicians like Tom Petty). The queries consist of 1000 short cell phone recordings of live performances, and were generated in the following fashion. For each band, 10 live audio tracks were extracted from Youtube videos, each from a different song. The videos were all recorded from smartphones during actual live performances. For each cell phone recording, the audio was

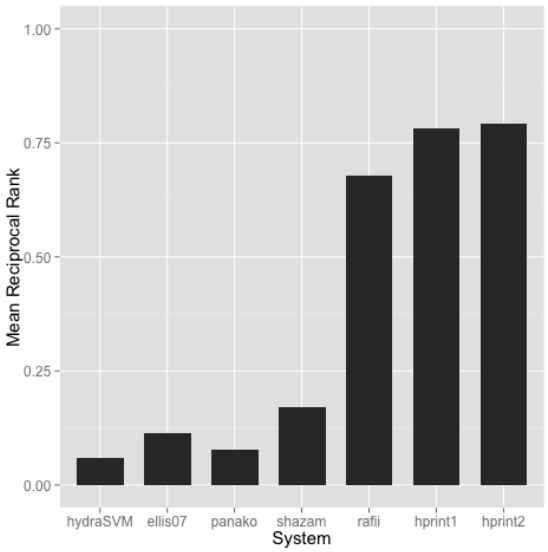


Figure 3. Mean reciprocal rank for five baseline systems and the two proposed systems (hprint1, hprint2).

cropped to exclude any non-music material at the beginning or end (e.g. applause, introducing the song, etc). Finally, ten 6-second segments evenly spaced throughout the cropped recording were extracted. Thus, there are 100 6-second queries for each band, totaling 1000 queries.

3.2 Evaluation Metric

We use mean reciprocal rank (MRR) as our evaluation metric [24]. This measure is defined by the equation

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{R_i}$$

where N is the number of queries and R_i specifies the rank of the correct answer in the i^{th} query. When a song has two or more studio versions, we define R_i to be the best rank among the multiple studio versions. The MRR is a succinct way to measure rankings when there is an objective correct answer. Note that when a system performs perfectly — it returns the correct answer as the first item every time — it will have an MRR of 1. A system that performs very poorly will have an MRR close to 0. Higher MRR is better.

3.3 Results

Figure 3 compares the performance of the proposed hashprint1 and hashprint2 systems with five different baselines. The first two baselines (HydraSVM [16] and Ellis07 [6]) are open-source cover song detection systems. The next two baselines (Panako [21] and Shazam [25]) are open-source audio fingerprinting systems. [3] The fifth baseline is the previously proposed live song identification system by Rafii et al. [15]. In order to allow for a more fair comparison, we also ran this baseline system with four quarter tone pitch shifts above and below the original studio

[3] For the Shazam baseline, we used the implementation by Ellis [5].

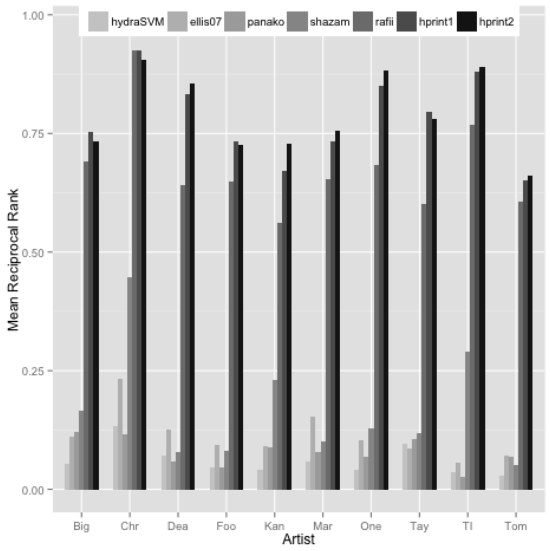


Figure 4. Breakdown of results by artist. The first three letters of the artist's name is shown at bottom.

recording. The two rightmost bars in Figure 3 show the performance of the hashprint1 and hashprint2 systems, respectively. Figure 4 shows the same results broken down by artist.

There are four things to notice in Figures 3 and 4. First, cover song and fingerprinting approaches perform poorly. The first four baseline systems suggest that existing cover song detection and existing audio fingerprinting approaches may not be suitable solutions to the live song identification problem. Audio fingerprinting approaches typically assume that the underlying source signal is identical, and may not be able to cope with the variations found in live performances. On the other hand, cover song detection systems typically assume that an entire clean studio track is available, and may not cope well with short, noisy queries. Second, the proposed systems improve upon the previous state-of-the-art. Comparing the three rightmost systems, we see that the two proposed systems improve the MRR from .68 (rafii) up to .78 (hashprint1) and .79 (hashprint2). Given the reciprocal nature of the evaluation metric, this amounts to a major improvement in performance. Third, the more computationally efficient version of the proposed system (hashprint2) has the best performance. In system design, we often sacrifice accuracy for efficiency. But in this case, we observe no degradation in system performance while reducing computational cost. The reason for this, as we will see in Section 4, is because the extra degrees of freedom in the DTW matching are not necessary. We will also investigate the runtime performance of these systems in the next section. Fourth, performance varies by artist. We see a wide variation in MRR from artist to artist, but all three live song identification systems generally agree on which artists are 'hard' and which are 'easy'. One major factor determining this difficulty level is how much variation there is between an artist's studio recording and live performance. The other major factor, of course, is how many studio tracks are in the database. Note that the best performance (Chromeo) and worst performance (Tom Petty) correlate with how many studio tracks the artist had.

Matching	Downsample	MRR	Runtime (s)
DTW	-	**.78**	**29.3**
xcorr	1	.81	3.43
xcorr	2	.80	1.26
xcorr	**3**	**.79**	**.90**
xcorr	4	.77	.76
xcorr	5	.73	.69

Table 2. Effect of downsampling on a cross-correlation matching approach. The third and fourth columns show system performance and average runtime required to process each 6-second query. The top row shows the performance of a DTW matching approach for comparison. The first and fourth rows correspond to the hashprint1 and hashprint2 systems shown in Figure 3.

4. ANALYSIS

In this section, we investigate two different questions of interest about the proposed systems.

4.1 Runtime

The first question of interest to us is "What is the runtime of the proposed systems?" Since live song identification is a real-time application, the amount of latency is a very important consideration. Table 2 shows the average runtime of a cross-correlation approach across a range of downsampling rates. This is the average amount of time required to process each 6-second query. [4] The runtime for a subsequence DTW approach is also shown for reference. The first and fourth rows (highlighted in bold) correspond to the hashprint1 and hashprint2 systems shown in Figure 3.

There are three things to notice about Table 2. First, cross-correlation is unilaterally better than DTW. When we compare the first two rows of Table 2, we see that switching from DTW to cross-correlation drastically reduces the runtime (from 29.3s to 3.43s) while simultaneously improving the performance (from .78MRR to .81MRR). These results are an indication that the extra degrees of freedom in the DTW matching are not beneficial or necessary. Across a short 6-second query, it appears that we can simply assume a 1-to-1 tempo correspondence and allow the context frames in each hashprint to absorb slight mismatches in timing. Of course, this conclusion only generalizes to the extent that these 10 artists are representative of other live song identification scenarios.

Second, downsampling trades off accuracy for efficiency. When we compare the bottom five rows of Table 2, we see a tradeoff between MRR and average runtime: as downsampling rate increases, we sacrifice performance for efficiency. For a downsampling rate of 3 (the hashprint2

[4] Note that the runtime scales linearly with the size of the database. So, for example, the runtime for Tom Petty will be longer than for Chromeo.

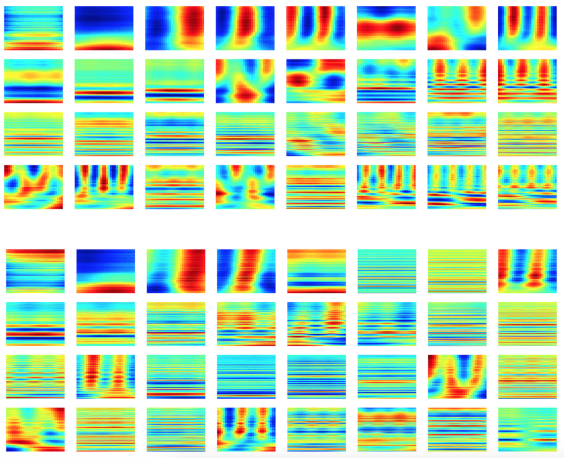


Figure 5. Learned filters for Big K.R.I.T. (top four rows) and Taylor Swift (bottom four rows). The filters are ordered first from left to right, then from top to bottom. Each filter spans .372 sec and covers a frequency range from C3 to C8.

system), we can reduce the average runtime to under a second, while only sacrificing a little on accuracy (MRR falls from .81 to .79). Note that the previously proposed system by Rafii et al. [15] has a self-reported runtime of 10 seconds per query, so the hashprint2 system may offer substantial improvement in runtime efficiency. [5]

Third, there is a floor to the runtime. Note that using a downsampling rate higher than 3 only benefits the average runtime marginally. This is because there is a fixed cost (about .5 seconds) for computing the CQT. The downsampling can only improve the time spent searching the database, but the time required to compute the query hashprints is a fixed cost. In a commercial application, however, the CQT could be computed in a streaming manner, so that the effective latency experienced by the user is determined by the search time. Such an optimization, however, is beyond the scope of this work.

4.2 Filters

The second question of interest to us is "What do the learned filters look like?" This can provide intuition about what type of information the hashprint is capturing. Figure 5 shows the top 32 learned filters for Big K.R.I.T. (top four rows) and Taylor Swift (bottom four rows). The filters are arranged first from left to right, and then from top to bottom. Each filter spans .372 sec (horizontal axis) and covers a frequency range from C3 to C8 (vertical axis).

There are three things to notice about the filters in Figure 5. First, they contain both temporal and spectral modulations. Some of the filters primarily capture modulations in time, such as filters 3, 4, 5, and 8 in the first row. Some filters primarily capture modulations in frequency, such as the filters in row 3 that contain many horizontal bands. Other filters capture modulations in both time and frequency, such as filters 15 and 16 (in row 2), which seem to capture temporal modulations in the higher frequencies and spectral modulations in the lower frequencies. The important thing to notice is that both types of modulations are important. If our hashprint representation only considered the CQT energy values for a single context frame, we would hinder the representational power of the hashprints.

Second, the filters capture both broad and fine spectral structures. Many of the filters capture pitch-like quantities based on fine spectral structure, which appear as thin horizontal bands. But other filters capture very broad spectral structure (such as filter 6, row 1) or treat broad ranges of frequencies differently (such as filters 15 and 16, previously mentioned). Whereas many other feature representations often focus on only fine spectral detail or only broad spectral structure, the hashprint seems to be capturing both types of information.

Third, the filters are artist-specific. When we compare the filters for Big K.R.I.T. and the filters for Taylor Swift, we can see that the hashprint representation adapts to the characteristics of the artist's music. The first four filters of both artists seem to be very similar, but thereafter the filters begin to reflect the unique characteristics of each artist. For example, more of the filters for Big K.R.I.T. seem to emphasize temporal modulations, perhaps an indication that rap tends to be more rhythmic and percussion-focused. In contrast, the filters for Taylor Swift seem to have more emphasis on pitch-related information, which may indicate music that is more based on harmony.

5. CONCLUSION

We have proposed a system for a known-artist live song identification task based on short, noisy cell phone recordings. Our system represents audio as a sequence of hashprints, which is a Hamming embedding based on a set of spectro-temporal filters. These spectro-temporal filters can be learned in an unsupervised manner to adapt the hashprint representation for each artist. Matching is performed using a cross-correlation approach with downsampling and rescoring. Based on experiments with the Gracenote live song identification benchmark, the proposed system improves the mean reciprocal rank of the previous state-of-the-art from .68 to .79, while simultaneously reducing the average runtime per query from 10 seconds down to 0.9 seconds. Future work will focus on characterizing the effect of various system parameters such as number of context frames, Hamming dimension, and database size.

6. ACKNOWLEDGMENTS

We would like to thank Zafar Rafii and Markus Cremer at Gracenote for generously providing the data set, and Brian Pardo for helpful discussions. Thomas Prätzlich has been supported by the German Research Foundation (DFG MU 2686/7-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institut für Integrierte Schaltungen IIS.

[5] Since we re-implemented this baseline system without optimizing for runtime efficiency, we rely on the self-reported runtime in [15].

7. REFERENCES

[1] S. Baluja and M. Covell. Waveprint: Efficient wavelet-based audio fingerprinting. *Pattern Recognition*, 41(11):3467–3480, May 2008.

[2] M. Casey, C. Rhodes, and M. Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(5):1015–1028, 2008.

[3] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational Geometry*, pages 253–262, 2004.

[4] J. Downie, M. Bay, A. Ehmann, and M. Jones. Audio cover song identification: MIREX 2006-2007 results and analyses. In *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pages 468–474, 2008.

[5] D. Ellis. Robust landmark-based audio fingerprinting. Available at `http://labrosa.ee.columbia.edu/matlab/fingerprint/`, 2009.

[6] D. Ellis and G. Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1429–1432, 2007.

[7] D. Ellis and B. Thierry. Large-scale cover song recognition using the 2d fourier transform magnitude. In *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pages 241–246, 2012.

[8] P. Grosche and M. Müller. Toward characteristic audio shingles for efficient cross-version music retrieval. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 473–476, 2012.

[9] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[10] F. Kurth and M. Müller. Efficient index-based audio matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):382–395, 2008.

[11] M. Müller. *Fundamentals of Music Processing*. Springer, 2015.

[12] M. Norouzi, D. Blei, and R. Salakhutdinov. Hamming distance metric learning. In *Advances in neural information processing systems*, pages 1061–1069, 2012.

[13] P. Over, G. Awad, J. Fiscus, B. Antonishek, M. Michel, A.F. Smeaton, W. Kraaij, and G. Quénot. TRECVID 2011 - An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. In *TRECVID 2011 - TREC Video Retrieval Evaluation Online*, Gaithersburg, Maryland, USA, December 2011.

[14] C. Raffel and D. Ellis. Large-scale content-based matching of midi and audio files. In *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pages 234–240, 2015.

[15] Z. Rafii, B. Coover, and J. Han. An audio fingerprinting system for live version identification using image processing techniques. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 644–648, 2014.

[16] S. Ravuri and D. Ellis. Cover song detection: from high scores to general classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 65–68, 2010.

[17] R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

[18] C. Schörkhuber and A. Klapuri. Constant-q transform toolbox for music processing. In *Sound and Music Computing Conference*, pages 3–64, 2010.

[19] J. Serra, E. Gómez, and P. Herrera. Audio cover song identification and similarity: background, approaches, evaluation, and beyond. In *Advances in Music Information Retrieval*, pages 307–332. Springer, 2010.

[20] J. Serra, X. Serra, and R. Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11(9):093017, 2009.

[21] J. Six and M. Leman. Panako: a scalable acoustic fingerprinting system handling time-scale and pitch modification. In *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2014.

[22] R. Sonnleitner and G. Widmer. Quad-based audio fingerprinting robust to time and frequency scaling. In *Proceedings of the International Conference on Digital Audio Effects*, 2014.

[23] T. Tsai and A. Stolcke. Robust and efficient multiple alignment of unsynchronized meeting recordings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):833–845, 2016.

[24] E.M. Voorhees. The TREC-8 question answering track report. In *Proceedings of the 8th Text Retrieval Conference*, pages 77–82, 1999.

[25] A. Wang. An industrial-strength audio search algorithm. In *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pages 7–13, 2003.

[26] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems 21 (NIPS'09)*, pages 1753–1760, 2009.

LEARNING TEMPORAL FEATURES USING A DEEP NEURAL NETWORK AND ITS APPLICATION TO MUSIC GENRE CLASSIFICATION

Il-Young Jeong and Kyogu Lee
Music and Audio Research Group
Graduate School of Convergence Science and Technology, Seoul National University, Korea
{finejuly, kglee}@snu.ac.kr

ABSTRACT

In this paper, we describe a framework for temporal feature learning from audio with a deep neural network, and apply it to music genre classification. To this end, we revisit the conventional spectral feature learning framework, and reformulate it in the cepstral modulation spectrum domain, which has been successfully used in many speech and music-related applications for temporal feature extraction. Experimental results using the GTZAN dataset show that the temporal features learned from the proposed method are able to obtain classification accuracy comparable to that of the learned spectral features.

1. INTRODUCTION

Extracting features from audio that are relevant to the task at hand is a very important step in many music information retrieval (MIR) applications, and the choice of features has a huge impact on the performance. For the past decades, numerous features have been introduced and successfully applied to many different kinds of MIR systems. These audio features can be broadly categorized into two groups: 1) spectral and 2) temporal features.

Spectral features (SFs) represent the spectral characteristics of music in a relatively short period of time. In a musical sense, it can be said to reveal the timbre or tonal characteristics of music. Some of popular SFs include: spectral centroid, spectral spread, spectral flux, spectral flatness measure, mel-frequency cepstral coefficients (MFCCs) and chroma. On the other hand, temporal features (TFs) describe the relatively long-term dynamics of a music signal over time such as temporal transition or rhythmic characteristics. These include zero-crossing rate (ZCR), temporal envelope, tempo histogram, and so on. The two groups are not mutually exclusive, however, and many MIR applications use a combination of many different features.

The abovementioned features - be it spectral or temporal - have one thing in common: they are all 'hand-crafted' features, which are highly based on the domain knowledge or signal processing techniques. With the rapid advances in the field of machine learning and deep learning in particular, however, more recent works have become less dependent of using the standard audio features but instead try to 'learn' optimal features [4]. These approaches usually take no preprocessing step [1] or least, such as a magnitude spectrum [2,12] or mel-scale filter banks [1,10], but just let the machine learn the optimal features for a given task. Although a number of feature learning approaches have been proposed so far for many MIR-related applications, most of them have focused on learning SFs for a short-time signal [2, 12]. In case of TFs, on the other hand, few studies tried to apply deep learning models but it was limited to training the classification model from the high-level features [11, 14].

In this paper, we endeavor to learn TFs using a deep neural network (DNN) from a low-level representation. By reversing the conventional SF learning and temporal aggregation, we aim to learn TFs for a narrow spectral band and summarize them by using spectral aggregation. Furthermore, we parallelize SF and TF learning frameworks, and combine the two resulting features to use as a front end to a genre classification system. We expect this approach to provide a performance gain because each learned feature conveys different types of information present in a musical audio.

2. CONVENTIONAL FRAMEWORK FOR SPECTRAL FEATURE LEARNING

In this section, we briefly revisit how SFs are extracted using a DNN in a typical classification framework [12]. Figure 1 (a) shows the block diagram of its overall framework, which is similar to the proposed method for temporal feature learning except the input representation and feature aggregation. Let s_i be a single channel waveform of i-th music data with a label y_i. Here, the label can be various high-level descriptor, including genre, mood, artist, chord, and tag. A magnitude spectrogram of s_i, X_i, is computed using short-time Fourier transform (STFT) defined by

© Il-Young Jeong and Kyogu Lee. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Il-Young Jeong and Kyogu Lee. "learning temporal features using a deep neural network and its application to music genre classification", 17th International Society for Music Information Retrieval Conference, 2016.

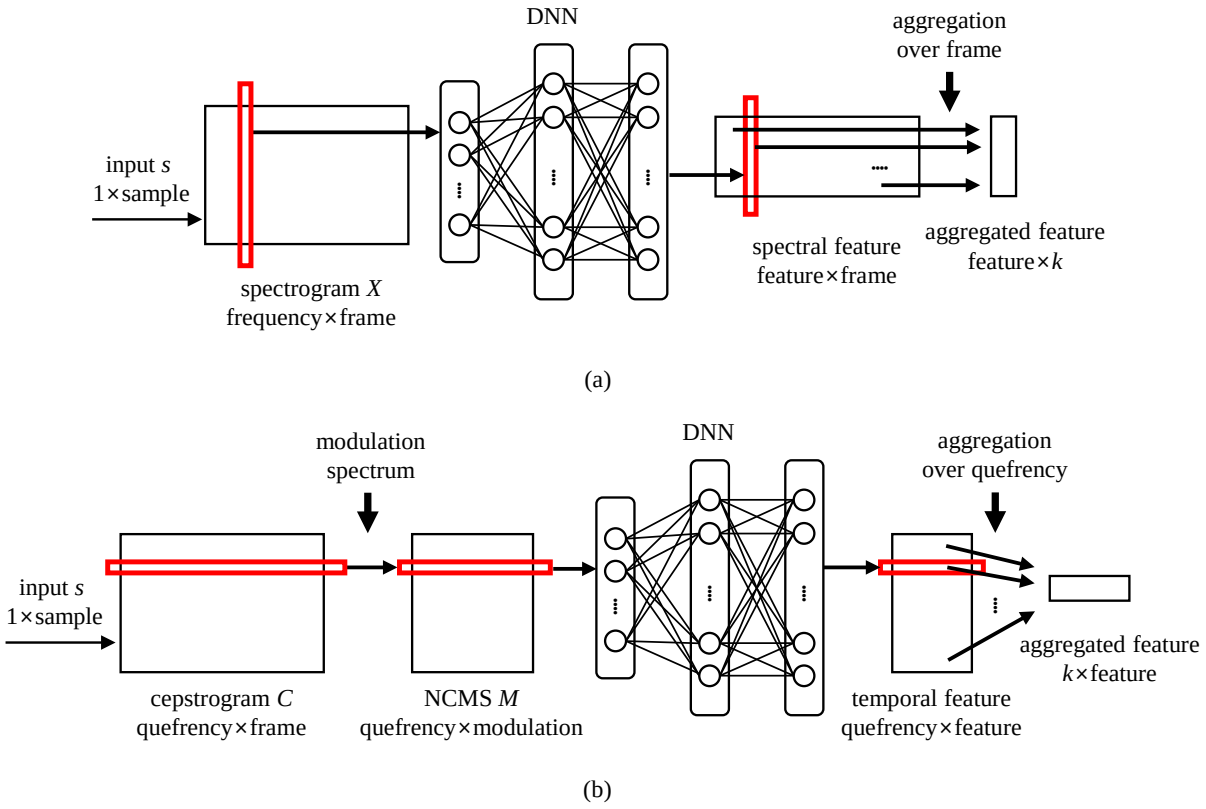


Figure 1. Overall frameworks for (a) conventional spectral feature learning and (b) proposed temporal feature learning. Some details (*e.g.* normalization) are omitted. k denotes the number of aggregation methods. All figures in the paper are best viewed in color.

$$X_i(f,t) = \left| \sum_{n=0}^{N-1} s_i\left(\Lambda t + n\right) w\left(n\right) \exp\left(-j\frac{2\pi f n}{N}\right) \right|, \tag{1}$$

where f and t denote a index of frequency bin and time frame, and N and Λ indicate a size and a shift of window function w, respectively. $|\cdot|$ denotes the absolute operator. A different time-frequency representation such as mel-spectrogram is also widely used [1, 10].

In order to remove the bias and reduce the variance, X is normalized that all frequency bins have zero-mean and unit variance across all the frames in training data as follows:

$$\bar{X}_i(f,t) = \frac{X_i(f,t) - \mu_X(f)}{\sigma_X(f)}, \tag{2}$$

where $\mu_X(f)$ and $\sigma_X(f)$ denote mean and standard deviation of the magnitude spectrogram of training data in f-th frequency bin, respectively. Sometimes amplitude compression or PCA whitening is added to a preprocessing step [10].

The training scheme is to learn a DNN model so that each normalized spectrum $\bar{x}_{i,t} = [\bar{X}_i(0,t), \ldots, \bar{X}_i(N/2,t)]$ ($N/2$ instead of N due to its symmetry) belongs to the target class of the original input y_i. In other words, it can be considered as a frame-wise classification model. After training the DNN, the activations of the hidden layers are used as features.

Because many high-level musical descriptors cannot be defined within a very short segment of time, the frame-wise features usually go through a feature aggregation step before classification. The aggregation is done within the specific time range, typically 3-6s, and depending on the applications various aggregation methods exist, including mean, variance, maximum, minimum, or moments [3]. The dimension of the final feature depends on the number of aggregation methods.

To summarize, the above spectral feature extraction framework for musical applications has three steps: 1) preprocessing (STFT, normalization), 2) feature learning (DNN), and 3) temporal aggregation (*e.g.*, average and variance over frames). In the next section, we propose how each step can be modified to extract the temporal features.

3. PROPOSED FRAMEWORK

In this section, we present the proposed method for temporal feature learning using the normalized cepstral modulation spectrum (normalized CMS or NCMS) and DNN. Overall procedure is illustrated in Figure 1 (b).

3.1 Normalized cepstral modulation spectrum

We first transform a music signal to the quefrency- normalized version of CMS [8,9] because a cepstrogram is shown to be a more robust representation to capture the dynamics of the overall timbre than a spectrogram. Although there are some variations of CMS such as mel-cepstrum modulation spectrum [15], we expect that CMS is able to minimize the information loss in the procedure. To compute the NCMS, the magnitude spectrogram in Eq. (1) is first transformed into a cepstrogram domain, which is harmonic decomposition of a logarithmic magnitude spectrum using inverse discrete Fourier transform (DFT). A cepstrogram is computed from a magnitude spectrogram X as follows:

$$C_i(q,t) = \frac{1}{N}\sum_{f=0}^{N-1} \ln\left(X_i(f,t)+\varepsilon\right)\exp\left(j\frac{2\pi qf}{N}\right), \quad (3)$$

where q is a quefrency index, and ε is a small constant to regularize a log operation. In this work, we empirically set ε to be 10^{-4}.

Similar to spectrogram normalization shown in Eq. (2), cepstrogram is normalized so as to have zero-mean and unit variance across quefencies:

$$\bar{C}_i(q,t) = \frac{C_i(q,t)-\mu_C(q)}{\sigma_C(q)}, \quad (4)$$

where $\mu_C(q)$ and $\sigma_C(q)$ denote mean and standard deviation of q-th quefrency bin in a cepstrogram of training data, respectively.

To analyze the temporal dynamics from the data, the shift invariance has to be considered since the extracted TFs are expected to be robust against its absolute location in time or phase. Some approaches were proposed for this purpose, such as l_2-pooling [5], but we chose a modulation spectrum because it is simpler to compute. In addition, modulation spectral characteristics can be analyzed over a few seconds instead of a whole signal, and thus are suitable for efficiently analyzing the local characteristics. The modulation spectrum of normalized cepstrogram is obtained as follows:

$$M_i(q,v,u) = \left|\sum_{t=u\Phi}^{u\Phi+T-1} \bar{C}_i(q,t)\exp\left(-j\frac{2\pi vt}{T}\right)\right|, \quad (5)$$

where v denotes the index of modulation frequency bin and u is the index of the sliding window that is T frames long with a Φ frames shift.

Finally, before being used as an input to a DNN, M is normalized for each modulation frequency v to have zero-mean and unit variance as in Eq. (2) as follows:

$$\bar{M}_i(q,v,u) = \frac{M_i(q,v,u)-\mu_M(v)}{\sigma_M(v)}, \quad (6)$$

where $\mu_M(v)$ and $\sigma_M(v)$ denote mean and standard deviation of v-th modulation frequency over the training data.

3.2 Temporal feature learning using deep neural network

The next step for temporal feature learning is the same as that of the spectral feature learning. The only difference is that an input vector of the DNN is now a normalized cepstral modulation spectrum $\bar{m}_{i,q,u} = [\bar{M}_i(q,0,u),\ldots,\bar{M}_i(q,T/2,u)], 0 \le q \le N/2$ which we expect better describes the long-term temporal properties over time for each quefrency.

3.3 Feature aggregation and combination

The output of a DNN in the previous section is a quefrency-wise feature, and therefore we need to aggregate it to be more appropriate as a front end to a classifier. We use the same aggregation method - *i.e.*, mean and variance - as we do in SF aggregation but only across quefrencies this time.

We believe that SFs described in Section 2 and TFs explained above represent the musical characteristics from different perspectives that can complement each other. By setting the time window size for temporal aggregation in SF to be same as that for modulation analysis in TF, say 5s, we can combine the two features and construct a complementary feature set.

In the following section, we test the effectiveness of the proposed approach and present the results obtained using a benchmark music dataset.

4. EXPERIMENTS

4.1 Data preparation

To evaluate the proposed TFs and compare it with conventional SFs, we conducted genre classification task with the GTZAN database, which consists of 1,000 30-second long music clips with the sampling rate of 22,050 Hz [16]. Each clip is annotated with one of 10 genres and for each genre there are 100 clips. Even though some drawbacks and limits were indicated [13], it is still one of the most widely used datasets for music genre classification.

We examined the two different partitioning methods. First, we randomly divided the data into three groups: 50% for training, 25% for validation, and 25% for testing, maintaining the balance among genres. We performed the experiment four times to present the averaged results. This random partitioning gaurantees that the equal number of music clips is distributed among the different genres. However, random partitioning of the GTZAN dataset may lead to the numerical evaluation results that cannot be trusted because many clips in the GTZAN dataset are from the same artists. Therefore, we also tried the 'fault-filtered' partitioning, which manually divides the dataset into 443/ 197/290 to avoid the repetition of artist across training, validation, and testing sets [6].

4.2 Parameter setting

Parameters in the proposed framework are basically inspired from the conventional work [12]. For STFT, we

used Hanning window of N=1024 samples with half overlap of Λ=512. For NCMS, the number of frames and shift to analyze the temporal dynamics were set to be T=214 and Φ=107, respectively, which is the closest to 5s and 2.5s, respectively. The number of input units for DNN is thus 513 and 108, respectively, due to its symmetry. DNN is designed to have 3 hidden layers and each layer has 50 units for both spectral and temporal model. In other words, the network has a size of 513-50-50-50-10 for SF and 108-50-50-50-10 for TF. Rectified linear unit (ReLU) that is defined as $f(x) = max(0, x)$ was applied for the nonlinearity in every hidden layer, and the softmax function was used for the output layer. We did not use dropout or regularization terms since it did not help to improve the accuracy in our work, which is similar as previous work [12].

DNN was trained using mini-batch gradient descent with 0.01 step size and 100 batch size for both conventional and proposed algorithm. Optimization procedure was done after 200 epoches. By means of early-stopping strategy, the model which scores the lowest cross-entropy for the validation data is decided to be a final model with 10 patience.

In the aggregation stage, the outputs in the last hidden layer were aggregated using average and variance. In case of SF, the number of frames and shift for aggregation are set to be 214 and 107, respectively, which are the same as the temporal modulation window for TF. Although conventional studies also tried more complex model with various settings [2, 12], such as increasing the number of hidden units and aggregating with all the hidden layer, in this work we did not consider this kind of model settings since it is out of our scope. As shown in Figure 3, the proposed model with a simple setting already exceed the classification accuracy of the conventional approach with more complex model.

4.3 Genre classification

We performed genre classification using random forest (RF) with 500 trees as a classifier. Each music clip of 30s was first divided into a number of 5s-long short segments with 2.5s overlap. We then performed classification on each 5s-long segment, and used majority voting to classify the whole music clip. It is noted that both training and validation data were used to train RF since it does not require additional data for validation. The entire classification process, including training and testing, is illustrated in Figure 2.

Detailed results for each genre with the two partitioning methods are shown in Figure 3 and Figure 4. In case of random partitioning, overall accuracy of 72.6% was obtained using TFs, and 78.2% using SFs, respectively. The accuracy improved up to 85.0% when the two features are jointly used. Moreover, the combined features achieved the highest F-scores for all the genres except classical. Theses results suggest that the each type of feature contains information that helps improve genre classification.

With fault-filtered partitioning, the accuracy decreased in general, which is consistent with the results presented in [6]. Contrary to random partioning, however, the proposed TFs show much higher accuracy of 65.9% compared to 48.3% of SFs. Considering that the main difference between random and fault-filtered partitioning is artist repetition across train, validation and test sets, a possible explanation for this is that SFs are a better representation that captures similarity between the songs by the same artists. From the combined features, we obtained 59.7% accuracy which is lower than TFs alone. We believe that this unexpected performance degradation is due to the fact that the results were obtained from one trial with a fixed partition, which may have caused a bias. From an additional experiment where the classifier was trained using the training and testing sets and tested on the validation set, we obtained 50.3%, 57.4%, and 63.5% accuracies from SFs, TFs, and combined features, respectively.

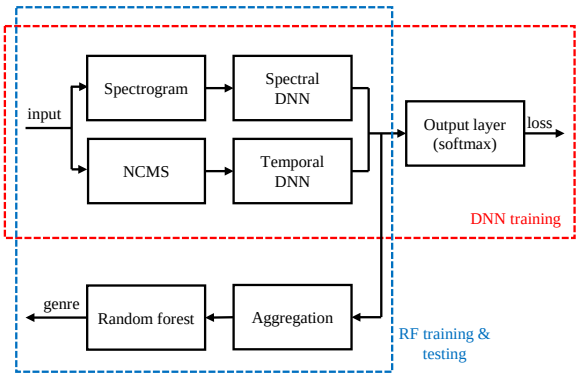


Figure 2. Overall framework for genre classification using conventional spectral features [12] and proposed temporal features.

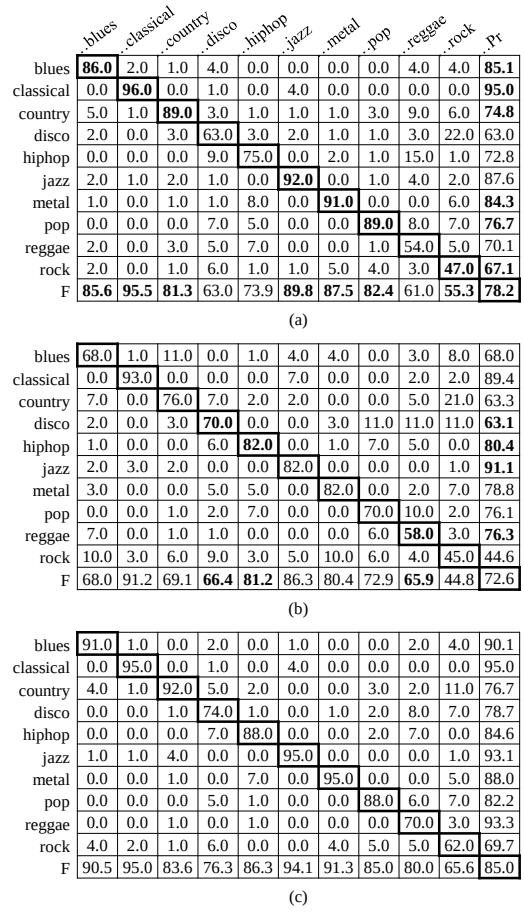
4.4 Feature visualization

To visually inspect the performance of different features, we visualized the features from test data using a 2-dimensional projection with t-sne [7]. Figure 5 and Figure 6 show the scatter plots of three different features, using random and fault-filtered partitioning, respectively. Although the classification accuracies are higher with random partitioning, it is not clearly represented in the figures. This may suggest that the higher performance with random partitioning is because of artist repetition, as explained in Section 4.3.

4.5 Discussion

Although the experimental results presented in the previous section are not sufficient to draw a firm conclusion, we can find some insights from our study worthy of further discussions. First, musical audio is an intrinsically time-varying signal, and understanding temporal dynamics is critical to better represent music. This has been done in various ways but we have demonstrated that using a more appropriate representation from the start helps achieve better performance.

The suitable domain for the analysis of temporal characteristics also leaves a room for more in-depth discus-



	..blues	..classical	..country	..disco	..hiphop	..jazz	..metal	..pop	..reggae	..rock	..Pr
blues	**86.0**	2.0	1.0	4.0	0.0	0.0	0.0	0.0	4.0	4.0	**85.1**
classical	0.0	**96.0**	0.0	1.0	0.0	4.0	0.0	0.0	0.0	0.0	**95.0**
country	5.0	1.0	**89.0**	3.0	1.0	1.0	1.0	3.0	9.0	6.0	**74.8**
disco	2.0	0.0	3.0	63.0	3.0	2.0	1.0	1.0	3.0	22.0	63.0
hiphop	0.0	0.0	0.0	9.0	75.0	0.0	2.0	1.0	15.0	1.0	72.8
jazz	2.0	1.0	2.0	1.0	0.0	**92.0**	0.0	1.0	4.0	2.0	87.6
metal	1.0	0.0	1.0	1.0	8.0	0.0	**91.0**	0.0	0.0	6.0	**84.3**
pop	0.0	0.0	0.0	7.0	5.0	0.0	0.0	**89.0**	8.0	7.0	**76.7**
reggae	2.0	0.0	3.0	5.0	7.0	0.0	0.0	1.0	54.0	5.0	70.1
rock	2.0	0.0	1.0	6.0	1.0	1.0	5.0	4.0	3.0	**47.0**	**67.1**
F	**85.6**	**95.5**	**81.3**	63.0	73.9	**89.8**	**87.5**	**82.4**	61.0	**55.3**	**78.2**

(a)

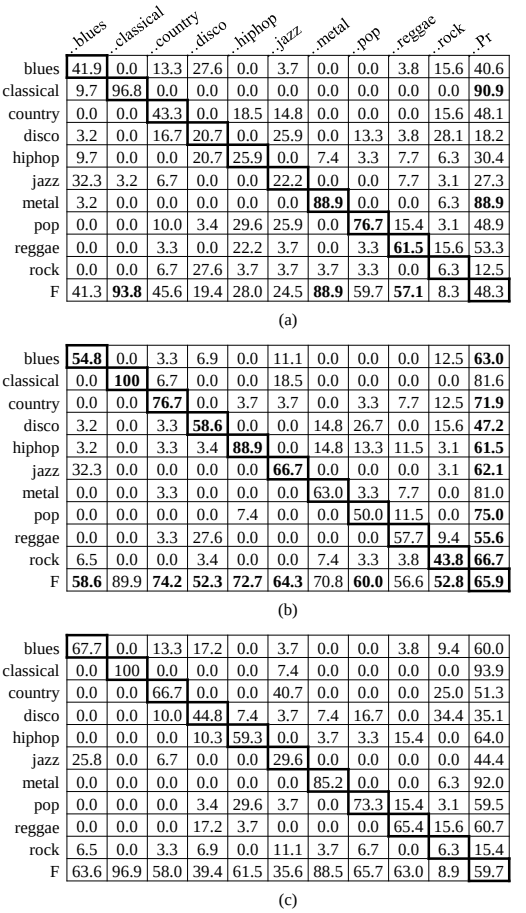
blues	68.0	1.0	11.0	0.0	1.0	4.0	4.0	0.0	3.0	8.0	68.0
classical	0.0	93.0	0.0	0.0	0.0	7.0	0.0	0.0	2.0	2.0	89.4
country	7.0	0.0	76.0	7.0	2.0	2.0	0.0	0.0	5.0	21.0	63.3
disco	2.0	0.0	3.0	**70.0**	0.0	0.0	3.0	11.0	11.0	11.0	**63.1**
hiphop	1.0	0.0	0.0	6.0	**82.0**	0.0	1.0	7.0	5.0	0.0	**80.4**
jazz	2.0	3.0	2.0	0.0	0.0	82.0	0.0	0.0	0.0	1.0	**91.1**
metal	3.0	0.0	0.0	5.0	5.0	0.0	82.0	0.0	2.0	7.0	78.8
pop	0.0	0.0	1.0	2.0	7.0	0.0	0.0	70.0	10.0	2.0	76.1
reggae	7.0	0.0	1.0	1.0	0.0	0.0	0.0	6.0	**58.0**	3.0	**76.3**
rock	10.0	3.0	6.0	9.0	3.0	5.0	10.0	6.0	4.0	45.0	44.6
F	68.0	91.2	69.1	**66.4**	**81.2**	86.3	80.4	72.9	**65.9**	44.8	72.6

(b)

blues	91.0	1.0	0.0	2.0	0.0	1.0	0.0	0.0	2.0	4.0	90.1
classical	0.0	95.0	0.0	1.0	0.0	4.0	0.0	0.0	0.0	0.0	95.0
country	4.0	1.0	92.0	5.0	2.0	0.0	0.0	3.0	2.0	11.0	76.7
disco	0.0	0.0	1.0	74.0	1.0	0.0	1.0	2.0	8.0	7.0	78.7
hiphop	0.0	0.0	0.0	7.0	88.0	0.0	0.0	2.0	7.0	0.0	84.6
jazz	1.0	1.0	4.0	0.0	0.0	95.0	0.0	0.0	0.0	1.0	93.1
metal	0.0	0.0	1.0	0.0	7.0	0.0	95.0	0.0	0.0	5.0	88.0
pop	0.0	0.0	0.0	5.0	1.0	0.0	0.0	88.0	6.0	7.0	82.2
reggae	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	70.0	3.0	93.3
rock	4.0	2.0	1.0	6.0	0.0	0.0	4.0	5.0	5.0	62.0	69.7
F	90.5	95.0	83.6	76.3	86.3	94.1	91.3	85.0	80.0	65.6	85.0

(c)

Figure 3. Figure of merit (FoM, $\times 100$) with random partitioning for (a) the conventional spectral features, (b) the proposed temporal features, and (c) the combined features. Each row and column represents the predicted and true genres respectively. The elements in the matrix denote the recall (diagonal), precision (last column), F-score (last row), confusions (off-diagonal), and overall accuracy (the last element of diagonal). The higher values of recall, precision, and F-score between (a) and (b) are emphasized in bold.



	..blues	..classical	..country	..disco	..hiphop	..jazz	..metal	..pop	..reggae	..rock	..Pr
blues	41.9	0.0	13.3	27.6	0.0	3.7	0.0	0.0	3.8	15.6	40.6
classical	9.7	96.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**90.9**
country	0.0	0.0	43.3	0.0	18.5	14.8	0.0	0.0	0.0	15.6	48.1
disco	3.2	0.0	16.7	20.7	0.0	25.9	0.0	13.3	3.8	28.1	18.2
hiphop	9.7	0.0	0.0	20.7	25.9	0.0	7.4	3.3	7.7	6.3	30.4
jazz	32.3	3.2	6.7	0.0	0.0	22.2	0.0	0.0	7.7	3.1	27.3
metal	3.2	0.0	0.0	0.0	0.0	0.0	**88.9**	0.0	0.0	6.3	**88.9**
pop	0.0	0.0	10.0	3.4	29.6	25.9	0.0	**76.7**	15.4	3.1	48.9
reggae	0.0	0.0	3.3	0.0	22.2	3.7	0.0	3.3	**61.5**	15.6	53.3
rock	0.0	0.0	6.7	27.6	3.7	3.7	3.7	3.3	0.0	6.3	12.5
F	41.3	**93.8**	45.6	19.4	28.0	24.5	**88.9**	59.7	**57.1**	8.3	48.3

(a)

blues	**54.8**	0.0	3.3	6.9	0.0	11.1	0.0	0.0	0.0	12.5	**63.0**
classical	0.0	**100**	6.7	0.0	0.0	18.5	0.0	0.0	0.0	0.0	81.6
country	0.0	0.0	**76.7**	0.0	3.7	3.7	0.0	3.3	7.7	12.5	**71.9**
disco	3.2	0.0	3.3	**58.6**	0.0	0.0	14.8	26.7	0.0	15.6	**47.2**
hiphop	3.2	0.0	3.3	3.4	**88.9**	0.0	14.8	13.3	11.5	3.1	**61.5**
jazz	32.3	0.0	0.0	0.0	0.0	**66.7**	0.0	0.0	0.0	3.1	**62.1**
metal	0.0	0.0	3.3	0.0	0.0	0.0	63.0	3.3	7.7	0.0	81.0
pop	0.0	0.0	0.0	0.0	7.4	0.0	0.0	50.0	11.5	0.0	**75.0**
reggae	0.0	0.0	3.3	27.6	0.0	0.0	0.0	0.0	57.7	9.4	**55.6**
rock	6.5	0.0	0.0	3.4	0.0	0.0	7.4	3.3	3.8	**43.8**	**66.7**
F	**58.6**	89.9	**74.2**	**52.3**	**72.7**	**64.3**	70.8	**60.0**	56.6	**52.8**	**65.9**

(b)

blues	67.7	0.0	13.3	17.2	0.0	3.7	0.0	0.0	3.8	9.4	60.0
classical	0.0	100	0.0	0.0	0.0	7.4	0.0	0.0	0.0	0.0	93.9
country	0.0	0.0	66.7	0.0	0.0	40.7	0.0	0.0	0.0	25.0	51.3
disco	0.0	0.0	10.0	44.8	7.4	3.7	7.4	16.7	0.0	34.4	35.1
hiphop	0.0	0.0	0.0	10.3	59.3	0.0	3.7	3.3	15.4	0.0	64.0
jazz	25.8	0.0	6.7	0.0	0.0	29.6	0.0	0.0	0.0	0.0	44.4
metal	0.0	0.0	0.0	0.0	0.0	0.0	85.2	0.0	0.0	6.3	92.0
pop	0.0	0.0	0.0	3.4	29.6	3.7	0.0	73.3	15.4	3.1	59.5
reggae	0.0	0.0	0.0	17.2	3.7	0.0	0.0	0.0	65.4	15.6	60.7
rock	6.5	0.0	3.3	6.9	0.0	11.1	3.7	6.7	0.0	6.3	15.4
F	63.6	96.9	58.0	39.4	61.5	35.6	88.5	65.7	63.0	8.9	59.7

(c)

Figure 4. Figure of merit (FoM, $\times 100$) with fault-filtered partitioning. Details are the same as Figure 3.

sion. While NCMS shows good performance in our experiments, it is probable that there exists a representation that can better describe temporal properties in music. One possible way would be analyzing temporal dynamics of SFs learned from DNN. It might be able to minimize the feature extraction step, and the process should be simpler by concatenating the spectral/temporal DNNs in series.

5. CONCLUSION

In this paper, we presented a novel feature learning framework using a deep neural network. In particular, while most studies have been trying to learn the spectral features from a short music segment, we focused on learning the features that represent the long-term temporal characteristics, which are expected to convey different information from that in the conventional spectral features. To this end, we used a normalized cepstral modulation spectrum as an input to DNN, and introduced a feature aggregation method over quefrencies. Experiments with genre classification show that the proposed temporal features yielded performance comparable to or better than that of the spectral features, depending on the partitioning methods of the dataset. We plan to apply the proposed method to various MIR-related tasks, including mood classification or instrument identification where spectral features are predominantly used. We also intend to develop a single framework in which both spectral and temporal features are jointly learned.

6. ACKNOWLEDGEMENTS

This research was supported by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2016-H8501-16-1016) supervised by the IITP (Institute for Information & communications Technology Promotion).

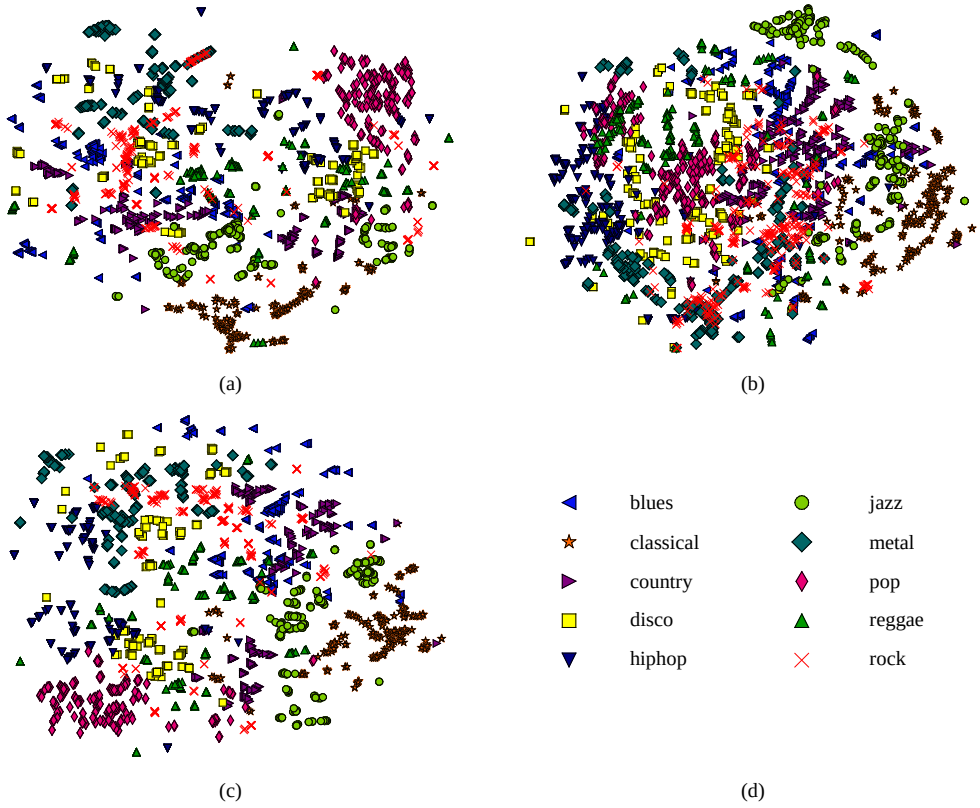


Figure 5. 2-dimensional scatter plots using t-sne [7] with random partitioning for (a) the conventional spectral features, (b) the proposed temporal features, and (c) the combined features. Each marker represents a 5s excerpt of a music signal whose genre is labeled as in (d).

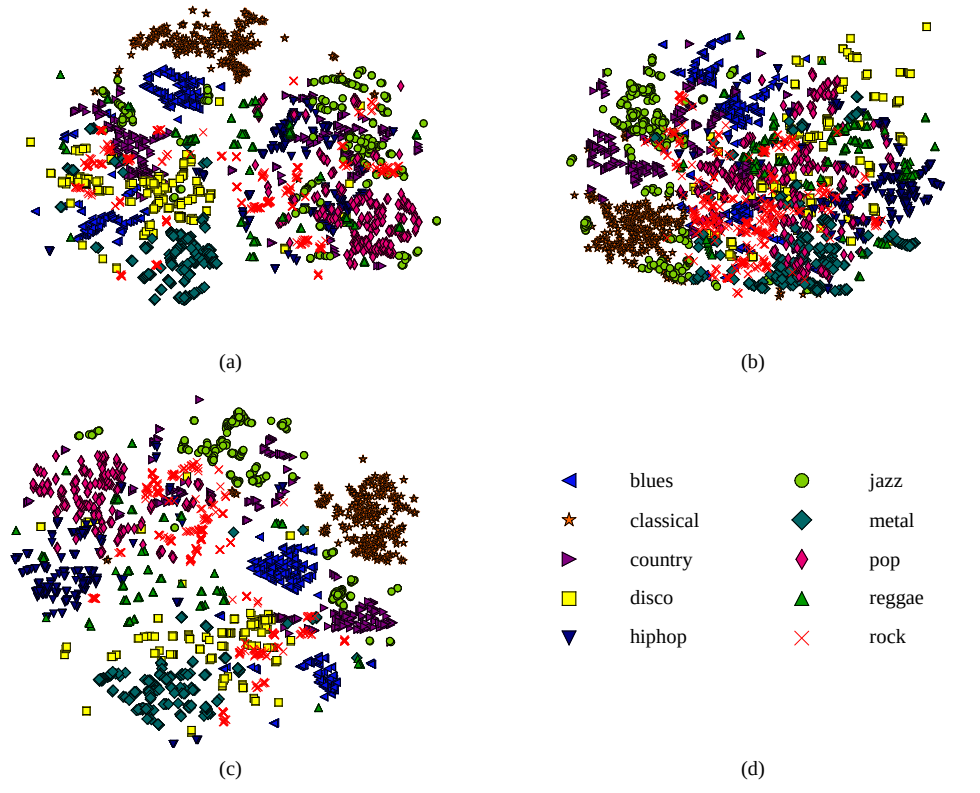


Figure 6. 2-dimensional scatter plots using t-sne [7] with fault-filtered partitioning. Details are the same as Figure 5.

7. REFERENCES

[1] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*. Florence, Italy, 2014.

[2] Philippe Hamel and Douglas Eck. Learning features from music audio with deep belief networks. In *ISMIR*, pages 339–344. Utrecht, Netherlands, 2010.

[3] Philippe Hamel, Simon Lemieux, Yoshua Bengio, and Douglas Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *ISMIR*, pages 729–734, 2011.

[4] Eric J. Humphrey, Juan P. Bello, and Yann LeCun. Feature learning and deep architectures: New directions for music informatics. *Journal of Intelligent Information Systems*, 41(3):461–481, 2013.

[5] Aapo Hyvärinen and Patrik Hoyer. Emergence of phase-and shift-invariant features by decomposition of natural images into independent feature subspaces. *Neural computation*, 12(7):1705–1720, 2000.

[6] Corey Kereliuk, Bob L. Sturm, and Jan Larsen. Deep learning and music adversaries. *IEEE Transactions on Multimedia*, 17(11):2059–2071, 2015.

[7] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[8] Rainer Martin and Anil Nagathil. Cepstral modulation ratio regression (CMRARE) parameters for audio signal analysis and classification. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009.

[9] Anil Nagathil, Timo Gerkmann, and Rainer Martin. Musical genre classification based on a highly-resolved cepstral modulation spectrum. In *Proceedings of the European Signal Processing Conference*, 2010.

[10] Juhan Nam, Jorge Herrera, Malcolm Slaney, and Julius O Smith. Learning sparse feature representations for music annotation and retrieval. In *ISMIR*, pages 565–570, 2012.

[11] Aggelos Pikrakis. A deep learning approach to rhythm modeling with applications. In *Proceedings of the International Workshop on Machine Learning and Music*, 2013.

[12] Siddharth Sigtia and Simon Dixon. Improved music feature learning with deep neural networks. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*. Florence, Italy, 2014.

[13] Bob L. Sturm. The state of the art ten years after a state of the art: Future research in music information retrieval. *Journal of New Music Research*, 43(2):147–172, 2014.

[14] Bob L. Sturm, Corey Kereliuk, and Jan Larsen. ¿El caballo viejo? latin genre recognition with deep learning and spectral periodicity. *Mathematics and Computation in Music*, pages 335–346, 2013.

[15] Vivek Tyagi, Iain McCowan, Hemant Misra, and Hervé Bourlard. Mel-cepstrum modulation spectrum (MCMS) features for robust asr. In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 399–404, 2003.

[16] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

METER DETECTION IN SYMBOLIC MUSIC USING INNER METRIC ANALYSIS

W. Bas de Haas
Utrecht University
W.B.deHaas@uu.nl

Anja Volk
Utrecht University
A.Volk@uu.nl

ABSTRACT

In this paper we present PRIMA: a new model tailored to symbolic music that detects the meter and the first downbeat position of a piece. Given onset data, the metrical structure of a piece is interpreted using the Inner Metric Analysis (IMA) model. IMA identifies the strong and weak metrical positions in a piece by performing a periodicity analysis, resulting in a weight profile for the entire piece. Next, we reduce IMA to a feature vector and model the detection of the meter and its first downbeat position probabilistically. In order to solve the meter detection problem effectively, we explore various feature selection and parameter optimisation strategies, including Genetic, Maximum Likelihood, and Expectation-Maximisation algorithms. PRIMA is evaluated on two datasets of MIDI files: a corpus of ragtime pieces, and a newly assembled pop dataset. We show that PRIMA outperforms autocorrelation-based meter detection as implemented in the MIDItoolbox on these datasets.

1. INTRODUCTION

When we listen to a piece of music we organise the stream of auditory events seemingly without any effort. Not only can we detect the beat days after we are born [31], as infants we are able to develop the ability to distinguish between a triple meter and duple meter [18]. The processing of metrical structure seems to be a fundamental human skill that helps us to understand music, synchronize our body movement to the music, and eventually contributes to our musical enjoyment. We believe that a system so crucial to human auditory processing must be able to offer great merit to Music Information Retrieval (MIR) as well. But what exactly constitutes meter, and how can models of metrical organisation contribute to typical MIR problems? With the presentation of the PRIMA [1] model we aim to shed some light on these matters in this paper.

The automatic detection of meter is an interesting and challenging problem. Metrical structure has a large influence on the harmonic, melodic and rhythmic structure of a piece, and can be very helpful in many practical situations. For instance, in [30] a statistical exploration of common syncopation patterns in a large corpus of symbolic ragtime pieces is presented. For correct analysis of syncopation patterns knowledge of the meter is essential. However, many corpora lack reliable meter annotations, making automatic meter detection a prerequisite for rhythmic pattern analysis. Similarly, chord recognition algorithms have been shown to improve when metrical information is taken into account, e.g. [3]. Finally, also melodic similarity estimation benefits from (automatically derived) metrical information. Humans appear to be more tolerant to note transformations placed on weak metrical positions [11].

In this paper we present PRIMA: a new model for detecting the meter and the first downbeat in a sequence of onsets. Where most other approaches reduce the problem to a binary duple / triple meter detection, PRIMA estimates all time signatures that are available in the training set and also detects the first downbeat position. PRIMA's architecture is outlined as follows: the model employs Inner Metric Analysis [28, IMA] to determine the strong and weak metrical positions in an onset sequence. The IMA is folded into one-bar profiles, which are subsequently optimised. This metrical analysis feature serves as input to a probabilistic model which eventually determines the meter. Finally, two feature optimisation strategies are discussed and evaluated.

PRIMA is trained and tested on two datasets of MIDI files: the RAG collection [30] and the newly collected FMpop collection. The main motivation for choosing the RAG collection for evaluation is that there is a clear need for meter and first downbeat detection for facilitating corpus-based studies on this dataset. Since Ragtime is a genre that is defined by syncopated rhythms [30], information on meter and the location of the first downbeat is crucial for corpus-based rhythm analyses. In order to assess the flexibility of PRIMA, we also train and evaluate the model on a new dataset of pop music: the FMpop collection. All data has been produced by music enthusiasts and is separated into a test and a training set. Both datasets are too big to manually check all meter annotations. Therefore, we assume that in the training set the majority of the meters are correctly annotated. In the test set, the meter and first downbeat positions are manually corrected, and this confirms the intuition that the majority of the meters is correct, but annotation errors do occur.

Taks description: We define the meter detection task

[1] Probabilistic Reduction of Inner Metric Analysis

© W. Bas de Haas, Anja Volk. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** W. Bas de Haas, Anja Volk. "Meter Detection in Symbolic Music Using Inner Metric Analysis", 17th International Society for Music Information Retrieval Conference, 2016.

as follows: given a series of onsets, automatically detect the time signature and the position of the first beat of the bar. This first beat position is viewed as the offset of the meter measured from the starting point of an analysed segment, and we will refer to this offset as the *rotation* of the meter. [2] After all, a metrical hierarchy recurs every bar, and if the meter is stable, the first beat of the bar can easily be modelled by rotating the metrical grid. In this paper we limit our investigation to the $\frac{2}{2}$, $\frac{2}{4}$, $\frac{4}{4}$, $\frac{3}{4}$, $\frac{6}{8}$, $\frac{12}{8}$ meters that occur at least in 40 pieces of the dataset (five different meters in the RAG, four in the FMpop Collection). Naturally, additional meters can be added easily. In the case of duple / triple classification $\frac{2}{2}$, $\frac{2}{4}$, and $\frac{4}{4}$ are considered duple meters and $\frac{3}{4}$, $\frac{6}{8}$, and $\frac{12}{8}$ are considered triple meters. Within this study we assume that the meter does not change throughout an analysed segment, and we consider only MIDI data.

Contribution: The contribution of this paper is threefold. First, we present a new probabilistic model for automatically detecting the meter and first downbeat position in a piece. PRIMA is conceptually simple, based on a solid metrical model, flexible, and easy to train on style specific data, Second, we present a new MIDI dataset containing 7585 pop songs. Furthermore, for small subsets of this new FMpop Collection and a collection of ragtime pieces, we also present new ground-truth annotations of the meter and rotation. Finally, we show that all variants of PRIMA outperform the autocorrelation-based meter detection implemented in the MIDItoolbox [5].

2. RELATED WORK

The organisation of musical rhythm and meter has been studied for decades, and it is commonly agreed upon that this organisation is best represented hierarchically [13]. Within a metrical hierarchy strong metrical positions can be distinguished from weaker positions, where strong positions positively correlate with the number of notes, the duration of the notes, the number of equally spaced notes, and the stress of the notes [16]. A few (computational) models have been proposed that formalise the induction of metrical hierarchies, most notable are the models of Steedman [20], Longuet-Higgins & Lee [14] Temperley [21], and Volk [28]. However, surprisingly little of this work has been applied to the automatic detection of the meter (as in the time signature) of a piece of music, especially in the domain of symbolic music.

Most of the work in meter detection focusses on the audio domain and not on symbolic music. Although large individual differences exists, in the audio domain the meter detection systems follow a general architecture that consists of a feature extraction front-end and a model that accounts for periodicities in the onset or feature data. In the front-end typically features are used that are associated with *onset detection* such as spectral difference, or flux, and energy spectrum are used [1]. Or, in the symbolic case, one simply assumes that onset data is available [9, 22], like we do in this paper.

After feature extraction the periodicity of the onset data is analysed, which is typically done using auto-correlation [2, 23], a (beat) similarity matrix [6, 8], or hidden Markov models [17, 12]. Next, the most likely meter has to be derived from the periodicity analysis. Sometimes statistical machine learning techniques, such as Gaussian Mixture Models, Neural Networks, or Support Vector Machines [9], are applied to this task, but this is less common in the symbolic domain. The free parameters of these models are automatically trained on data that has meter annotations. Frequently the meter detection problem is simplified to classifying whether a piece uses a *duple* or *triple* meter [9, 23], but some authors aim at detecting more fine-grained time signatures [19, 24] and can even detect odd meters in culturally diverse music [10]. Albeit focussed on the audio domain, for a relatively recent overview of the field we refer to [24].

2.1 Inner Metric Analysis

Similar to most of the meter detection systems outlined in the previous section PRIMA relies on periodicity analysis. However, an important difference is that it uses the Inner Metric Analysis [28, IMA] instead of the frequently used autocorrelation. IMA describes the *inner* metric structure of a piece of music generated by the actual onsets opposed to the *outer* metric structure which is associated with an abstract grid annotated by a time signature in a score, and which we try to detect automatically with PRIMA.

What distinguishes IMA from other metrical models, such as Temperley's Grouper [21], is that IMA is flexible with respect to the number of metric hierarchies induced. It can therefore be applied both to music with a strong sense of meter, e.g. pop music, and to music with less pronounced or ambiguous meters. IMA has been evaluated in listening experiments [25], and on diverse corpora of music, such as classical pieces [26], rags [28], latin american dances [4] and on 20th century compositions [29].

IMA is performed by assigning a *metric weight* or a *spectral weight* to each onset of the piece. The general idea is to search for all chains of equally spaced onsets within a piece and then to assign a weight to each onset. This chain of equally spaced onsets underlying IMA is called a *local meter* and is defined as follows. Let On denote the set of all onsets of notes in a given piece. We define every subset $m \subset On$ of equally spaced onsets to be a local meter if it contains at least three onsets and is not a subset of any other subset of equally spaced onsets. Each local meter can be identified by three parameters: the starting position of the first onset s, the period denoting the distance between consecutive onsets d, and the number of repetitions k of the period (which equals the size of the set minus one).

The metric weight of an onset o is calculated as the weighted sum of the length k_m of all local meters m that coincide at this onset ($o \in m$), weighted by parameter p that regulates the influence of the length of the local meters on the metric weight. Let $M(\ell)$ be the set of all local

[2] We chose the new term *rotation* for the offset of the meter because the musical terms generally used to describe this phenomenon, like *anacrusis*, *upbeat figure*, or *pickup*, are sometimes interpreted differently.

meters of the piece of length at least ℓ, then the metric weight of an onset, $o \in On$, is defined as follows:

$$W_{\ell,p}(o) = \sum_{\{m \in M(\ell): o \in m\}} k_m^p. \quad (1)$$

The spectral weight is calculated in a similar fashion, but for the spectral weight each local meter is extended throughout the entire piece. The idea behind this is that the metrical structure induced by the onsets stretches beyond the region in which onsets occurs. The extension of a local meter m is defined as $ext(m_{s,d,k}) = \{s + id, \forall i\}$ where i is an integer number. For all discrete metrical positions t, regardless whether it contains an onset or not, the spectral weight is defined as follows:

$$SW_{\ell,p}(t) = \sum_{\{m \in M(\ell): t \in ext(m)\}} k_m^p. \quad (2)$$

In this paper we have used the standard parameters $p = 2$, and $\ell = 2$. Hence, we consider all local meters that exist in a piece. A more elaborate explanation of the IMA including examples can be found in [28].

3. IMA BASED METER DETECTION

In this section we will outline the PRIMA model in a bottom-up fashion. We start with the input MIDI data, and describe how we transform this into onset data, perform IMA and finally optimise a feature based on IMA. Next, we explain how this feature is used in a probabilistic model to detect the meter and rotation of sequence of onsets, and we elaborate on two different training strategies.

3.1 Quantisation and Preprocessing

Before we can perform IMA, we have to preprocess the MIDI files to obtain a quantised sequence of onsets. The following preprocessing steps are taken:

To be able to find periodicities, the onset data should be quantised properly. Within Western tonal music duple as well as triple subdivisions of the beat occur commonly. Hence, we use a metrical grid consisting of 12 equally spaced onset positions per quarter note. With this we can quantise both straight and *swung* eight notes. Here, *swing* refers to the characteristic long-short rhythmical pattern that is particularly common in Jazz, but is found throughout popular music.

In the quantisation process we use the length of a quarter note as annotated in the MIDI file. This MIDI *time division* specifies the number of MIDI ticks per quarter note and controls the resolution of the MIDI data. Because the MIDI time division is constant, strong tempo deviations in the MIDI data might distort the quantisation process and the following analyses. To estimate the quality of the alignment of the MIDI data to the metrical grid, we collect the quantisation deviation for every onset, and the average quantisation deviation divided by the MIDI time division gives a good estimate of the quantisation error. To make sure that the analysed files can be quantised reasonably well, we discard all MIDI files with an average quantisation error higher than 2 percent.

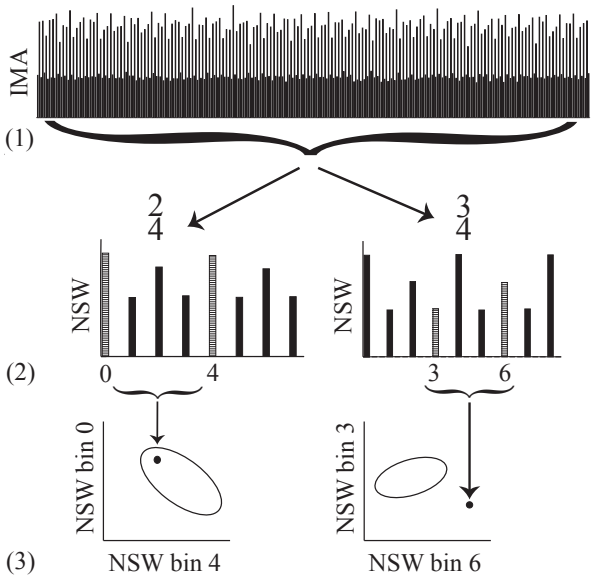


Figure 1. The construction of NSW profiles for a piece in $\frac{2}{4}$: (1) displays IMA, (2) displays the NSW profiles derived from IMA for a $\frac{2}{4}$ and a $\frac{3}{4}$ meter, and (3) shows how two bins are selected from each profile and used to estimate the probability of that meter. The ellipse represents the Gaussian distribution fitted to selected bins of the NSW profiles in the training phase. Note that the $\frac{3}{4}$ NSW profile does not resemble a typical $\frac{3}{4}$ and receives a low probability. Also, the selected bins may differ per optimisation strategy.

After quantising the MIDI data, we collect all onset data from all voices and remove duplicate onsets. Next, the MIDI data is segmented at all positions where a meter change is annotated in the MIDI file. Segments that are empty or shorter than 4 bars are excluded from further analysis. Also, MIDI files that do not contain meter annotations at all are ignored in the training phase.

3.2 Normalised spectral weight profiles

We use the spectral weights of IMA to construct a feature for detecting the meter in a piece. More specifically, this feature will model the conditional probability of a certain meter given a sequence of onsets. As we will explain in more detail in the next section, the distribution of these features will be modelled with a Gaussian distribution. We call this feature a *Normalised Spectral Weight* (NSW) profile, and discern three stages in constructing them: (1) perform IMA, (2) folding the IMA in one-bar profiles and normalising the weights profiles, and (3) selecting the most relevant bins for modelling the Gaussian distribution. These three stages are displayed schematically in Figure 1, and are detailed below.

IMA marks the metrical importance of every quantised onset position in a piece. Because of the large numbers of spectral weights and the large differences per piece, IMA cannot be used to detect the meter directly. How-

ever, we can fold the analysis into one-bar profiles to get a more concise metrical representation for every candidate meter. These one-bar profiles are created by summing the spectral weights per quantised position within a bar. Consequently, the shape of these profiles is determined by the meter (beats per bar), the length of piece (longer pieces yield higher spectral weights), and the number of quantisation bins.

We normalise spectral weights in the one-bar profiles by applying Formula 3:

$$\text{normalise}(w) = \log(\frac{w}{n^{\beta}} + \alpha) \tag{3}$$

Here, w is the summed spectral weight of a particular quantised beat position and n is the number of bars used to create the profile. We use a parameter β to control the effect of the length of the piece in the normalisation. Furthermore, because many quantised beat positions might have a summed metrical weight of 0, and this will cause problems when we fit Gaussian models to these profiles, we use Laplace smoothing [15, p. 260] and add a constant factor α to all weights. Finally, because statistical analysis of large amounts of profiles showed that differences in weights are distributed normally on a logarithmic scale, we apply the natural logarithm in Eq. 3. For the results in this report we have used $\beta = 2$ and $\alpha = 1$. We call these profiles *Normalised Spectral Weight* (NSW) Profiles.

The raw NSW profiles cannot yet be conveniently used as a feature for meter detection: the dimensionality of the NSW profiles is relatively high, and the dimensionality differs per meter. Also, not every metrical position within a bar is equally important. For instance, the first beat of the bar will have a high spectral weight, while the metrical position of the second eighth note will generally have a much lower spectral weight. Hence, we select profile bins that contain the information most relevant for meter detection.

The selection of the relevant profile bins is a special case of feature dimensionality reduction where the feature bins are highly dependent on each other. In this section we introduce two selection methods that will be experimentally verified in Section 4.3. A first intuition is to select the n profile bins that have the highest weights on average for a given dataset. A brief analysis showed that these bins roughly correspond to the first n principal components. However, a preliminary evaluation shows that NSW profiles containing only these bins perform relatively poorly. Hence, in order to learn more about what are the distinctive bins in the profiles, we use a Genetic Algorithm (GA) to explore the space of possible bin selections. [3] When we analyse these bin selections, we notice that the GA selects bins for a meter that contain weights that are maximally different to other meters.

[3] Note that these n profile bins may differ between meters, and n does not have to be the same for all meters, as long as the bin selection of the examined profile is exactly the same as the selection used in the template profile for that meter. For the implementation of the GA we use the Haskell library `https://github.com/boegel/GA`, using a population size of 100 candidates, a crossover rate of 0.7, a mutation rate of 0.2, and Eq. 5 as fitness function.

Training a GA on large amounts of data takes a lot of time, even if the NSW profiles are pre-calculated. Since we have a clear intuition about how the GA selects profile bins, we might be able to mimic this behaviour without exploring the complete space of possible bin selections. Recall when we classify a single piece, we calculate multiple NSW profiles for a single IMA: one for each meter. If we select the same bins in each profile for matching, i.e. every first beat of a bar, the chances are considerable that this selection will match multiple meters well. Hence, we select the n bins of which the NSW profiles of the ground-truth meter are maximally different from the NSW profiles of other meters. In this calculation we define maximally different as having a maximal absolute difference in spectral weight, and n does not differ between meters. We call this method the *Maximally Different Bin* (MDB) Selection.

3.3 A probabilistic meter classification model

To restate our initial goal: we want to determine the meter and its rotation given a sequence of note onsets. Ignoring rotation, a good starting point is to match NSW profiles with template profiles of specific meters. However, although the spectral weights of IMA reflect human intuitions about the musical meter [27], it is rather difficult to design such templates profiles by hand. Moreover, these template profiles might be style specific. Hence, we propose a model that learns these templates from a dataset.

Another style dependent factor that influences meter detection is the distribution of meters in a dataset. For instance, in Ragtime $\frac{2}{2}$ and $\frac{2}{4}$ occur frequently, while pop music is predominantly notated in a $\frac{4}{4}$ meter. Just matching NSW profiles with meter templates will not take this into account. When we combine simple profile matching with a weighting based on a meter distribution (prior), this conceptually equals a Naive Bayes classifier [15]. Therefore, probabilistically modelling meter detection is a natural choice.

If we ignore the rotations for sake of simplicity, we can express the probability of a meter given a set of note onsets with Equation 4:

$$P(\text{meter}|\text{onsets}) \propto P(\text{onsets}|\text{meter}) \cdot P(\text{meter}) \tag{4}$$

Here, $P(\text{onset}|\text{meter})$ reflects the probability of an onset sequence given a certain meter, and $\propto$ denotes "is proportional to". Naturally, certain meters occur more regularly in a dataset than others which is modelled by $P(\text{meter})$. The conditional probability $P(\text{onset}|\text{meter})$ can be estimated using NSW profiles. Given a piece and a specific meter we create an NSW profile that can be used as multidimensional feature. Given a large dataset that provides us with sequences of onsets and meters, we can model the distribution of the NSW profiles as Gaussian distributions. For every meter in the dataset we estimate the mean and covariance matrix of a single Gaussian distribution with the expectation-maximization algorithm [7]. The prior probability of a certain meter, $P(\text{meter})$, can be estimated with maximum likelihood estimation, which equals the number

of times a certain meter occurs in a dataset divided by the total number of meters in the dataset.

Adding the estimation of the rotation makes the problem slightly more complicated. A natural way of incorporating rotation is to add it as a single random variable that is dependent on the meter. This makes sense because it is likely that the kind of rotation depends on the kind of meter: an anacrusis in a $\frac{4}{4}$ meter is likely to differ from an anacrusis in a $\frac{3}{4}$ meter. Hence, we can transform Eq. 4 into the following equation:

$$P(x, r|y) \propto P(y|x, r) \cdot P(r|x) \cdot P(x) \tag{5}$$

Similar to Eq. 4, we estimate the meter x given an onset pattern y, but now we also add the rotation r. The term $P(y|x, r)$ can again be modelled with NSW profiles, but now the profiles should also be rotated according to the rotation r. The term $P(x)$ reflects the probability of a meter and can be estimated with maximum likelihood estimation.

We do not consider all possible rotations. For a $\frac{4}{4}$ meter there are $4 \cdot 12 = 48$ possible rotations, many of which are not likely to occur in practise. The rotations are modelled as a fraction of a bar, making the rotation meter independent. Furthermore, we rotate clock-wise, e.g. $\frac{1}{4}$ represents an anacrusis of one quarter note in a $\frac{4}{4}$ meter. The space of possible rotations can be further reduced by only considering two kinds of rotations: rotations for duple and triple meters. After all, given the very similar metrical structure of $\frac{2}{4}$ and $\frac{4}{4}$, we expect that the rotations will be similar as well (but on another absolute metrical level, e.g. eighth instead of quarter notes). For duple meters we explore eight, and for triple meters we explore six different rotations.

Unfortunately, estimating the prior probability of the rotation given a certain meter, i.e. $P(r|x)$, is not trivial because we rely on MIDI data in which the rotation is not annotated. Hence, we need another way of estimating this prior probability of the rotation. We estimate the rotation by calculating all rotations of the NSW profiles and pick the rotation that maximises probability of the annotated ground-truth meter. Having an estimation of the best fitting rotation per piece, we can perform maximum likelihood estimation by counting different rotations for each meter in order to obtain the rotation probabilities.

3.4 Training

We train and evaluate PRIMA on two datasets (see Sec. 4.1 and 4.2). These datasets consist of MIDI files created by music enthusiasts that might have all sorts of musical backgrounds. Hence, it is safe to assume that the meter annotations in these MIDI files might sometimes be incorrect. A likely scenario is, for instance, that MIDI creation software adds a $\frac{4}{4}$ meter starting at the first note onset by default, while the piece in question starts with an upbeat and is best notated in $\frac{3}{4}$. Nevertheless, we assume that the *majority* of the meters is annotated correctly, and that incorrect meters will only marginally effect the training of PRIMA.

In this paper we evaluate two different ways of training PRIMA. We use Maximally Different Bin (MDB) selection in the feature training phase, or alternatively, we use a GA to select the most salient NSW profile bins. After the bin selection, we use Maximum Likelihood estimation to learn the priors and rotation, as described in the previous section, and Expectation-Maximisation for fitting the Gaussian distributions.

4. EVALUATION

To assess the quality of the meter and rotations calculated by PRIMA, we randomly separate our datasets into testing and training sets. The test sets are manually corrected and assured to have a correct meter and rotation. The next two sections will detail the data used to train and evaluate PRIMA. The manual inspection of the meters and rotations confirms the intuition that most of the meters are correct, but the data does contain meter and rotation errors.

4.1 RAG collection

The RAG collection that has been introduced in [30] currently consists of 11545 MIDI files of ragtime pieces that are collected from the Internet by a community of Ragtime enthusiasts. The collection is accompanied by an elaborate compendium [4] that stores additional information about individual ragtime compositions, like year, title, composer, publisher, etc. The MIDI files in the RAG collection describe many pieces from the ragtime era (approx. $1890 \sim 1920$), but also modern ragtime compositions. The dataset is separated randomly in a test set of 200 pieces and a training set of 11345 pieces. After the preprocessing detailed in Sec. 3.1, 74 and 4600 pieces are considered suitable for respectively testing and training. For one piece we had to correct the meter and for another piece the rotation.

4.2 FMpop collection

The RAG corpus only contains pieces in the ragtime style. In order to study how well PRIMA predicts the meter and rotation of regular pop music, we collected 7585 MIDI files from the website Free-Midi.org. [5] This collection comprises MIDI files describing pop music from the 1950 onwards, including various recent hit songs, and we call this collection the FMpop collection. For evaluation we randomly select a test set of 200 pieces and we use the remainder for training. In the training and test sets, 3122 and 89 pieces successfully pass the preprocessing stage, respectively. Most of the pieces that drop out have a quantisation error greater than 2 percent. For three pieces we had to correct the meter, and for four pieces the rotation.

4.3 Experiments

We perform experiments on both the RAG and the FMpop collections in which we evaluate the detection performance by comparing the proportion of correctly classified meters, rotations, and the combinations of the two. In these experiments we probe three different training variants of PRIMA: (1) a variant where we use Maximally Different Bin (MDB)

[4] see `http://ragtimecompendium.tripod.com/` for more information

[5] `http://www.free-midi.org/`

RAG Collection

(Training) model	Meter	Rotation	Both
Duple / Triple meters			
MIDItoolbox	.76	–	–
MDB selection (2 bins)	.97	.88	.86
MDB selection (3 bins)	.97	.97	.95
GA optimized	.97	.99	**.96**
Meters: $\frac{2}{2}$, $\frac{2}{4}$, $\frac{3}{4}$, $\frac{4}{4}$, $\frac{6}{8}$			
MDB selection (2 bins)	.85	.92	.80
MDB selection (3 bins)	.80	.92	.76
GA optimised	.84	.93	**.82**

Table 1. The proportion of correctly detected meter and rotation in the RAG collection. The first section shows the duple / triple meter classification, the second section shows the proportions for the five most used time signatures.

FMpop Collection

(Training) model	Meter	Rotation	Both
Duple / Triple meters			
MIDItoolbox	.74	–	–
MDB selection (2 bins)	.94	.90	**.85**
MDB selection (3 bins)	.90	.93	.84
GA optimized	.94	.88	.83
Meters: $\frac{3}{4}$, $\frac{4}{4}$, $\frac{6}{8}$, $\frac{12}{8}$			
MDB selection (2 bins)	.94	.81	.79
MDB selection (3 bins)	.94	.81	.78
GA optimised	.94	.91	**.87**

Table 2. The correctly detected proportion for on the FMpop collection for duple / triple meter classification and for the four most used time signatures.

selection in which we select the two most salient bins and (2) a variant in which we select the three most salient bins. Finally, (3) we also use a Genetic algorithm to select the bins and estimate the rotation priors.

To place the performance of PRIMA into context, we compare the results to the meter detection model implemented in the MIDItoolbox [5]. This model only predicts whether a meter is duple or triple and does not predict the time signature. Therefore, we can compare the MIDItoolbox meter finding to PRIMA only in the duple / triple case. To ensure we use the exact same input data, we have written our own NMAT export script that transforms the MIDI as preprocessed by PRIMA into a matrix that can be parsed by the MIDItoolbox. All source code and data reported in this study is available on request.

4.4 Results

We evaluate the performance PRIMA and its different training strategies on duple / triple meter detection and the detection of five different time signatures. In Table 1 the proportions of correctly detected meters in the RAG collection are displayed. In the duple / triple meter detection experiments all variants of PRIMA outperform the MIDItoolbox meter detection. We tested the statistical significance of all individual differences between MIDItoolbox meter detection and PRIMA using McNemar's χ^2 test, and all differences are significant ($p < 0.001$). In the classification of five different time signatures the performance drops considerably. However, rags are mostly notated in $\frac{2}{4}$, $\frac{4}{4}$, and $\frac{2}{2}$ meters, and even experienced musicians have difficulty determining what is the correct meter. Still PRIMA achieves a 96 percent correct estimation for meter and rotation in the duple / triple experiment and 82 percent correct estimation on the full time signature detection.

In Table 2 the proportions of correctly classified meters in the FMpop Collection are displayed. Also on onsets extracted from popular music, PRIMA outperforms the MIDItoolbox meter finding. Again, we tested the statistical significance of the differences between all PRIMA variants using McNemar's χ^2 test, and all differences are statistically significant ($p < 0.002$ for GA and MDB selection (2 bins), and $p < 0.017$ for MDB selection (3 bins)). Overall, PRIMA's performance on the FMpop Collection is lower than on the RAG Collection for the duple / triple detection, but higher for time signature detection. Respectively, 85 and 87 percent correct classification is achieved for both meter and rotation. Generally, the GA seems to yield the best results.

5. DISCUSSION AND CONCLUSION

We presented a new model for detecting the meter and first downbeat position of a piece of music. We showed that IMA is valuable in the context of meter and first downbeat detection. PRIMA is flexible, can be easily trained on new data, and is conceptually simple. We have shown that PRIMA performs well on the FMpop and RAG Collections and outperforms the MIDItoolbox meter finding model. However, while PRIMA can be trained on data of specific styles, the parameters of the MIDItoolbox meter detection model are fixed. Hence, the performance of the MIDItoolbox should be seen as a baseline system that places PRIMA's results into context.

In this study we applied PRIMA to MIDI data only because we believe that corpus based analyses on collections like the RAG collection can really benefit from meter finding. Nevertheless, PRIMA's IMA based feature and probabilistic model are generic and can be easily applied to onset sequences extracted from audio data. Hence, it would be interesting to investigate how PRIMA model performs on audio data, and compare it to the state-of-the-art in audio meter detection. We strongly believe that also in the audio domain meter detection can benefit from IMA. We are confident that IMA has the potential to aid in solving many MIR tasks in both the audio and the symbolic domain.

6. ACKNOWLEDGMENTS

A. Volk and W.B. de Haas are supported by the Netherlands Organization for Scientific Research, through the NWO-VIDI-grant 276-35-001 to Anja Volk.

7. REFERENCES

[1] J.P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M.B. Sandler. A tutorial on onset detection in music signals. *IEEE Trans. ASLP*, 13(5):1035–1047, 2005.

[2] J.C. Brown. Determination of the meter of musical scores by autocorrelation. *JASA*, 94(4):1953–1957, 1993.

[3] R. Chen, W. Shen, A. Srinivasamurthy, and P. Chordia. Chord recognition using duration-explicit hidden markov models. In *Proc. ISMIR*, pages 445–450, 2012.

[4] E. Chew, A. Volk, and C.-Y. Lee. Dance music classification using inner metric analysis. In *The next wave in computing, optimization, and decision technologies*, pages 355–370. Springer, 2005.

[5] T. Eerola and P. Toiviainen. *MIDI Toolbox: MATLAB Tools for Music Research*. University of Jyväskylä, Jyväskylä, Finland, 2004.

[6] J. Foote and S. Uchihashi. The beat spectrum: a new approach to rhythm analysis. *Proc. IEEE ICME*, 0:224–228, 2001.

[7] C. Fraley and A. E. Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458):611–631, 2002.

[8] M. Gainza. Automatic musical meter detection. In *Proc. ICASSP*, pages 329–332, April 2009.

[9] F. Gouyon and P. Herrera. Determination of the meter of musical audio signals: Seeking recurrences in beat segment descriptors. In *Proc. AES*. Audio Engineering Society, 2003.

[10] A. Holzapfel, F. Krebs, and A. Srinivasamurthy. Tracking the "odd": Meter inference in a culturally diverse music corpus. In *Proc. ISMIR*, pages 425–430, 2014.

[11] P. van Kranenburg, A. Volk, F. Wiering, and R. C. Veltkamp. Musical models for folk-song alignment. In *Proc. ISMIR*, pages 507–512, 2009.

[12] F. Krebs, S. Böck, and G. Widmer. Rhythmic pattern modeling for beat and downbeat tracking in musical audio. In *Proc. ISMIR*, pages 227–232, 2013.

[13] F. Lerdahl and R. Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, 1996.

[14] H. C. Longuet-Higgins and C. S. Lee. The rhythmic interpretation of monophonic music. *Music Perception*, 1(4):424–441, 1984.

[15] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[16] C. Palmer and C. L. Krumhansl. Mental representations for musical meter. *JEPHPP*, 16(4):728, 1990.

[17] G. Peeters and H. Papadopoulos. Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation. *IEEE Trans. ASLP*, 19(6):1754–1769, Aug 2011.

[18] J. Phillips-Silver and L. J. Trainor. Feeling the beat: Movement influences infant rhythm perception. *Science*, 308(5727):1430, 2005.

[19] A. Pikrakis, I. Antonopoulos, and S. Theodoridis. Music meter and tempo tracking from raw polyphonic audio. In *Proc. ISMIR*, pages 192–197, 2004.

[20] M. J. Steedman. The perception of musical rhythm and metre. *Perception*, 6(5):555–569, 1977.

[21] D. Temperley. *The Cognition of Basic Musical Structures*. Cambridge, MA, MIT Press, 2001.

[22] P. Toiviainen and T. Eerola. The role of accent periodicities in meter induction: A classification study. In *Proc. ICMPC*, pages 422–425, 2004.

[23] P. Toiviainen and T. Eerola. Autocorrelation in meter induction: The role of accent structure. *JASA*, 119(2):1164–1170, 2006.

[24] M. Varewyck, J.-P. Martens, and M. Leman. Musical meter classification with beat synchronous acoustic features, DFT-based metrical features and support vector machines. *JNMR*, 42(3):267–282, 2013.

[25] A. Volk. The empirical evaluation of a mathematical model for inner metric analysis. In *Proc. ESCOM*, pages 467–470, 2003.

[26] A. Volk. Metric investigations in Brahms symphonies. In G. Mazzola, T. Noll, and E. Lluis-Puebla, editors, *Perspectives in Mathematical and Computational Music Theory*, pages 300–329. epOs Music, Osnabrück, 2004.

[27] A. Volk. Persistence and change: Local and global components of metre induction using inner metric analysis. *J. Math. Music*, 2(2):99–115, 2008.

[28] A. Volk. The study of syncopation using inner metric analysis: Linking theoretical and experimental analysis of metre in music. *J. New Music Res.*, 37(4):259–273, 2008.

[29] A. Volk. Applying inner metric analysis to 20th century compositions. In *MCM 2007*, CCIS 37, pages 204–210. Springer, 2009.

[30] A. Volk and W. B. de Haas. A corpus-based study on ragtime syncopation. In *Proc. ISMIR*, pages 163–168, 2013.

[31] I. Winkler, G. P. Háden, O. Ladinig, I. Sziller, and H. Honing. Newborn infants detect the beat in music. *PNAS*, 106(7):2468–2471, 2009.

MINIMAX VITERBI ALGORITHM FOR HMM-BASED GUITAR FINGERING DECISION

Gen Hori
Asia University / RIKEN
hori@brain.riken.jp

Shigeki Sagayama
Meiji University
sagayama@meiji.ac.jp

ABSTRACT

Previous works on automatic fingering decision for string instruments have been mainly based on path optimization by minimizing the difficulty of a whole phrase that is typically defined as the sum of the difficulties of moves required for playing the phrase. However, from a practical viewpoint of beginner players, it is more important to minimize the maximum difficulty of a move required for playing the phrase, that is, to make the most difficult move easier. To this end, we introduce a variant of the Viterbi algorithm (termed the "minimax Viterbi algorithm") that finds the path of the hidden states that maximizes the minimum transition probability (not the product of the transition probabilities) and apply it to HMM-based guitar fingering decision. We compare the resulting fingerings by the conventional Viterbi algorithm and our proposed minimax Viterbi algorithm to show the appropriateness of our new method.

1. INTRODUCTION

Most string instruments have overlaps in pitch ranges of their strings. As a consequence, such string instruments have more than one way to play even a single note (except the highest and the lowest notes that are covered only by a single string) and thus numerous ways to play a whole song. That is why the fingering decision for a given song is not always an easy task for string players and therefore automatic fingering decision has been attempted by many researchers. Previous works on automatic fingering decision have been mainly based on path optimization by minimizing the difficulty level of a whole phrase that is typically defined as the sum or the product of the difficulty levels defined for each move. (The product of difficulty levels easily reduces to the sum of the logarithm of the difficulty levels and therefore the sum and the product do not make any essential difference.) However, whether a string player can play a passage using a specific fingering depends almost only on whether the most difficult move included in the fingering is playable. Especially, from a practical viewpoint of beginner players, it is most important to minimize the maximum difficulty level of a move included in a fingering, that is, to make the most difficult move easier.

The purpose of this paper is to introduce a variant of the Viterbi algorithm [12] termed the "minimax Viterbi algorithm" that finds the sequence of the hidden states that maximizes the minimum transition probability on the sequence (not the product of all the transition probabilities on the sequence) and apply it to HMM-based guitar fingering decision. We employ a hidden Markov model (HMM) whose hidden states are left hand forms of guitarists and output symbols are musical notes, and perform fingering decision by solving a decoding problem of HMM using our proposed minimax Viterbi algorithm for finding the sequence of hidden states with the maximum minimum transition probability. Because the transition probabilities are set to large for easy moves and small for difficult ones, resulting fingerings "make the most difficult move easier" as previously discussed in this section. To distinguish the original Viterbi algorithm and our variant, we refer to the former as the "conventional Viterbi algorithm" and to the latter as the "minimax Viterbi algorithm" throughout the paper.

As for automatic fingering decision, several attempts have been made in the last two decades. Sayegh [10] first formulated fingering decision of string instruments as a problem of path optimization. Radicioni et al. [8] extended Sayegh [10]'s approach by introducing segmentation of musical phrase. Radisavljevic and Driessen [9] introduced a gradient descent search for the coefficients of the cost function for path optimization. Tuohy and Potter [11] first applied the genetic algorithm (GA) to guitar fingering decision and arrangements. As for applications of HMM to fingering decision, Hori et al. [4] applied input-output HMM [2] to guitar fingering decision and arrangement, Nagata et al. [5] applied HMM to violin fingering decision, and Nakamura et al. [6] applied merged-output HMM to piano fingering decision. Comparing to those previous works, the present work is new in that it introduces "minimax paradigm" to automatic fingering decision.

The rest of the paper is organized as follows. Section 2 recalls the conventional Viterbi algorithm and introduces our proposed minimax Viterbi algorithm. Section 3 introduces a framework of HMM-based fingering decision for monophonic guitar phrases. Section 4 applies the minimax Viterbi algorithm to fingering decision for monophonic guitar phrases and evaluates the results. Section 5 concludes the paper and discusses related future works.

© Gen Hori, Shigeki Sagayama. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Gen Hori, Shigeki Sagayama. "Minimax Viterbi algorithm for HMM-based Guitar fingering decision", 17th International Society for Music Information Retrieval Conference, 2016.

2. MINIMAX VITERBI ALGORITHM

We start by introducing our newly proposed "minimax Viterbi algorithm" on which we build our fingering decision method in the following section. First of all, we recall the definition of HMM [1] and the procedure of the conventional Viterbi algorithm for finding the sequence of hidden states that gives the maximum likelihood. Next, we modify the algorithm to our new one for finding the sequence of hidden states that gives the maximum minimum transition probability.

2.1 Hidden Markov model (HMM)

Suppose that we have two finite sets of hidden states Q and output symbols O,

$$Q = \{q_1, q_2, \ldots, q_N\},$$
$$O = \{o_1, o_2, \ldots, o_K\},$$

and two sequences of random variables $\boldsymbol{X}$ of hidden states and $\boldsymbol{Y}$ of output symbols,

$$\boldsymbol{X} = (X_1, X_2, \ldots, X_T),$$
$$\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_T),$$

then a hidden Markov model M is defined by a triplet

$$M = (A, B, \pi)$$

where A is an $N \times N$ matrix of the transition probabilities,

$$A = (a_{ij}),\ a_{ij} \equiv a(q_i, q_j) \equiv P(X_t = q_j | X_{t-1} = q_i),$$

B an $N \times K$ matrix of the output probabilities,

$$B = (b_{ik}),\ b_{ik} \equiv b(q_i, o_k) \equiv P(Y_t = o_k | X_t = q_i),$$

and Π an N-dimensional vector of the initial distribution of hidden states,

$$\Pi = (\pi_i),\ \pi_i \equiv \pi(q_i) \equiv P(X_1 = q_i).$$

2.2 Conventional Viterbi algorithm

When we observe a sequence of output symbols [2]

$$\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$$

from a hidden Markov model M, we are interested in the sequence of hidden states

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_T)$$

that generates the observed sequence of output symbols $\boldsymbol{y}$ with the maximum likelihood,

$$\begin{aligned}\hat{\boldsymbol{x}}_{ML} &= \arg\max_{\boldsymbol{x}} P(\boldsymbol{y}, \boldsymbol{x} | M) \\ &= \arg\max_{\boldsymbol{x}} P(\boldsymbol{x}|M) P(\boldsymbol{y}|\boldsymbol{x}, M) \\ &= \arg\max_{\boldsymbol{x}} \prod_{t=1}^{T} (a(x_{t-1}, x_t) b(x_t, y_t)) \qquad (1)\end{aligned}$$

where we write $a(x_0, x_1) = \pi(x_1)$ for convenience. The problem of finding the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ is called "decoding problem." Although an exhaustive search requires iterations over the N^T possible sequences, we can solve the problem efficiently using the Viterbi algorithm [12] based on dynamic programming (DP), which uses two $N \times T$ tables $\Delta = (\delta_{it})$ of maximum likelihood and $\Psi = (\psi_{it})$ of back pointers and the following four steps.

Initialization initializes the first columns of the two tables Δ and Ψ using the following formulae for $i = 1, 2, \ldots, N$,

$$\delta_{i1} = \pi_i\ b(q_i, y_1),$$
$$\psi_{i1} = 0.$$

Recursion fills out the rest columns of Δ and Ψ using the following recursive formulae for $j = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$–1,

$$\delta_{j,t+1} = \max_i(\delta_{it} a_{ij})\ b(q_j, y_{t+1}),$$
$$\psi_{j,t+1} = \arg\max_i(\delta_{it} a_{ij}).$$

Termination finds the index of the last hidden state of the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ using the last column of the table Δ,

$$i_T = \arg\max_i \delta_{iT}.$$

Backtracking tracks the indices of the hidden states of the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ from the last to the first using the back pointers of Ψ for $t = T, T$–$1, \ldots, 2$,

$$i_{t-1} = \psi_{i_t, t}$$

from which $\hat{\boldsymbol{x}}_{ML}$ is obtained as

$$x_t = q_{i_t} \quad (t = 1, 2, \ldots, T).$$

2.3 Modification for minimax Viterbi algorithm

Next, we consider the problem of finding the sequence of hidden states $\boldsymbol{x}$ with the maximum minimum transition probability [3],

$$\hat{\boldsymbol{x}}_{MM} = \arg\max_{\boldsymbol{x}} \min_{1 \le t \le T} (a(x_{t-1}, x_t) b(x_t, y_t)), \qquad (2)$$

which we call "minimax decoding problem [4]." A naive approach to the problem is an exhaustive search, that is, to enumerate all the sequences of the N hidden states and the length T, calculate the minimum transition probability for all the sequences, and find the one with the maximum value, which involves iterations over N^T sequences and is not for an actual implementation. Instead, we introduce a

[1] See [7] for more tutorial on HMM and its applications.

[2] According to the conventional notation of the probability theory, we denote random variables by uppercase letters and corresponding realizations by lowercase letters.

[3] Because the output probabilities are 0 or 1 in our application of HMM to guitar fingering decision, the sequence $\hat{\boldsymbol{x}}_{MM}$ eventually becomes the one with the maximum minimum transition probability, although its definition (2) depends on the output probabilities as well.

[4] Although the antonym "maximin" is appropriate for probability (which is the reciprocal of difficulty), we still use "minimax" for our proposed algorithm because it is appropriate for difficulty and conveys our concept of "make the most difficult move easier."

variant of the conventional Viterbi algorithm that can solve the problem efficiently. We modify the second step of the conventional Viterbi algorithm by replacing the term $\delta_{it}a_{ij}$ with $\min(\delta_{it}, a_{ij})$ where

$$\min(\delta_{it}, a_{ij}) = \begin{cases} \delta_{it} & (\delta_{it} \leq a_{ij}) \\ a_{ij} & (a_{ij} < \delta_{it}) \end{cases}.$$

The modified second step is as follows.

Recursion for minimax Viterbi algorithm fills out the two tables Δ and Ψ using the following recursive formulae for $j = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$–1,

$$\delta_{j,t+1} = \max_i(\min(\delta_{it}, a_{ij}))\ b_j(y_{t+1}),$$
$$\psi_{j,t+1} = \arg\max_i(\min(\delta_{it}, a_{ij})).$$

We modify only the second step and leave other steps unchanged. The modified second step works as the original one but now the element δ_{it} keeps the value of the maximum minimum transition probability of the subsequence of hidden states for the first t observations. The term $\min(\delta_{it}, a_{ij})$ updates the value of the minimum transition probability as the term $\delta_{it}a_{ij}$ in the conventional Viterbi algorithm does the likelihood [5].

3. FINGERING DECISION BASED ON HMM

We implement automatic fingering decision based on an HMM whose hidden states are left hand forms and output symbols are musical notes played by the left hand forms. In this formulation, fingering decision is cast as a decoding problem of HMM where a fingering is obtained as a sequence of hidden states. Because each hidden state has a unique output symbol, the output probability for the unique symbol is always 1. To compare the results of the conventional Viterbi algorithm and the minimax Viterbi algorithm clearly, we concentrate on fingering decisions for monophonic guitar phrases in the present study although HMM-based fingering decision is able to deal with polyphonic songs as well.

3.1 HMM for monophonic fingering decision

To play a single note with a guitar, a guitarist depresses a string on a fret with a finger of the left hand and picks the same string with the right hand. Therefore a form q_i for playing a single note can be expressed in a triplet

$$q_i = (s_i, f_i, h_i)$$

where $s_i = 1, \ldots, 6$ is a string number (from the highest to the lowest), $f_i = 0, 1, \ldots$ is a fret number, and $h_i = 1, \ldots, 4$ is a finger number of the player's left hand (1,2,3 and 4 are the index, middle, ring and pinky fingers). The fret number $f_i = 0$ means an open string for which the finger number h_i does not make sense. For a classical guitar with six strings and 19 frets, the total number of forms is $6 \times (19 \times 4 + 1) = 462$ [6]. For the standard tuning (E_4-B_3-G_3-D_3-A_2-E_2), the MIDI note numbers of the open strings are

$$n_1 = 64, n_2 = 59, n_3 = 55, n_4 = 50, n_5 = 45, n_6 = 40$$

from which the MIDI note number of the note played by the form q_i is calculated as

$$note(q_i) = n_{s_i} + f_i.$$

3.2 Transition and output probabilities

In standard applications of HMM, model parameters such as the transition probabilities and the output probabilities are estimated from training data using the Baum-Welch algorithm [1]. However, for our application of fingering decision, it is difficult to prepare enough training data, that is, machine-readable guitar scores attached with tablatures. For this reason, we design those parameters as explained in the following instead of estimation from training data.

The difficulty levels of moves are implemented in the transition probabilities between hidden states; a small value of the transition probability means the corresponding move is difficult and a large value easy. As for the movement of the left hand along the neck, the transition probability should be monotone decreasing with respect to the movement distance with the transition. Furthermore, the distribution of the movement distance is sparse and concentrates on the center because the left hand of a guitarist usually stays at a fixed position for several notes and then leaps a few frets to a new position. To approximate such a sparse distribution concentrated on the center, we employ the Laplace distribution (Figure 1),

$$f(x) = \frac{1}{2\phi}\exp\left(-\frac{|x-\mu|}{\phi}\right). \quad (3)$$

It is known that a one dimensional Markov process with increments according to the Laplace distribution is approximated by a piecewise constant function [3] that is similar to the movement of the left hand along the neck. The mean and the variance of the Laplace distribution (3) are μ and $2\phi^2$ respectively. We set μ to zero and ϕ to the time interval between the onsets of the two notes at both ends of the transition so that a long interval makes the transition probability larger, which reflects that a long interval makes the move easier. For simplicity, we assume that the four fingers of the left hand (the index, middle, ring and pinky fingers) are always put on consecutive frets. This lets us calculate the *index finger position* (the fret number the index finger is put on) of form q_i as follows,

$$ifp(q_i) = f_i - h_i + 1.$$

[5] Note that $\min(\delta_{it}, a_{ij})$ does not compare the probability of some subsequence and some transition probability but it does two transition probabilities here.

[6] The actual number of forms is less than this because the 19th fret is most often split by the sound hole and not usable for third and fourth strings, the players hardly place their index fingers on the 19th fret or pinky fingers on the first fret, and so on.

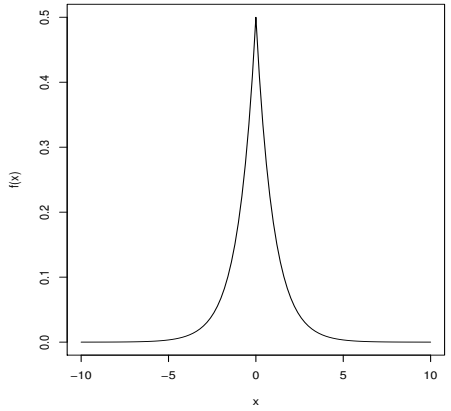


Figure 1. The probability density function of the Laplace distribution for $\mu = \phi = 1$, which is sparse and concentrates on the center.

Using the index finger position, we set the transition probability as

$$\begin{aligned} a_{ij}(d_t) &= P(X_t = q_j | X_{t-1} = q_i, d_t) \\ &\sim \frac{1}{2d_t} \exp(-\frac{|ifp(q_i) - ifp(q_j)|}{d_t}) \\ &\quad \times \frac{1}{1 + |s_i - s_j|} \times p_H(h_j) \end{aligned} \tag{4}$$

where d_t in the first term is set to the time interval between the onsets of the $(t-1)$-th note and the t-th note. The second term corresponds to the difficulty of changing between strings where we employ a function $1/(1+|x|)$ which is less sparse than the Laplace distribution (3). The third term $p_H(h_j)$ corresponds to the difficulty level of the destination form defined by the finger number h_j. In the simulation in the following section, we set $p_H(1) = 0.35$, $p_H(2) = 0.3$, $p_H(3) = 0.25$ and $p_H(4) = 0.1$ which means the form using the index finger is easiest and the pinky finger the most difficult. The difficulty levels of the forms are expressed in the transition probabilities (not in the output probabilities) in such a way that the transition probability is small when the destination form of the transition is difficult.

As for the output probability, because all the hidden states have unique output symbols in our HMM for fingering decision, it is 1 if the given output symbol o_k is the one that the hidden state q_i outputs and 0 if o_k is not,

$$\begin{aligned} b_{ik} &= P(Y_t = o_k | X_t = q_i) \\ &\sim \begin{cases} 1 & (\text{if } o_k = note(q_i)) \\ 0 & (\text{if } o_k \neq note(q_i)) \end{cases} . \end{aligned}$$

4. EVALUATION

To evaluate our proposed method, we compared the results of fingering decision using the conventional Viterbi algorithm and the minimax Viterbi algorithm. Figures2-4 show

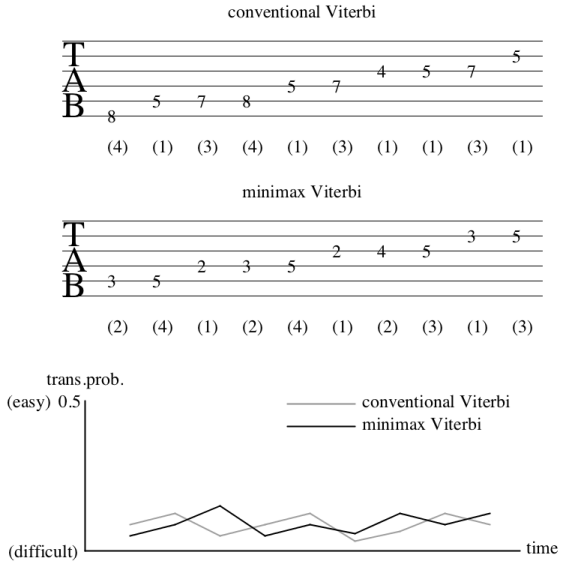


Figure 2. The results of fingering decision for the C major scale starting from C3. Comparing the two tablatures, the one obtained by the minimax Viterbi algorithm is more natural and one that actual guitarists would choose. As for the minimum transition probability, the line chart shows that the minimax Viterbi algorithm gives a larger one.

the results for three example monophonic phrases. In each figure, the top and the middle tablatures show the two fingerings obtained by the conventional Viterbi algorithm and the minimax Viterbi algorithm. The numbers on the tablatures show the fret numbers and the numbers in parenthesis below the tablatures show the finger numbers where 1,2,3 and 4 are the index, middle, ring and pinky fingers. The bottom line chart shows the time evolution of the transition probability of the conventional Viterbi algorithm (gray line) and the minimax Viterbi algorithm (black line). The two tablatures and the line chart share a common horizontal time axis, that is, a point on the line chart between two notes in the tablature indicates the transition probability between the two notes.

Figure 2 shows the results for the C major scale starting from C3. From the line chart of the transition probability, we see that the minimum value of the gray line (the conventional Viterbi algorithm) at the sixth transition is smaller than any value of the black line (the minimax Viterbi algorithm), that is, the minimax Viterbi algorithm gives a larger minimum transition probability. As for the tablatures, the one obtained by the minimax Viterbi algorithm is more natural and one that actual guitarists would choose.

Figure 3 shows the results for the opening part of "Romance Anonimo." From the line chart of the transition probability, we see that the gray line (the conventional Viterbi algorithm) keeps higher values at the cost of two very small values while the black line (the minimax Viterbi algorithm) avoids such very small values although it keeps relatively lower values. From the line charts of Figures

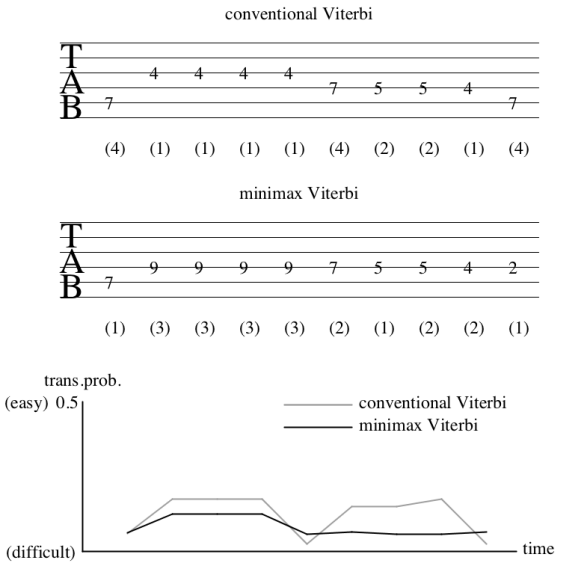


Figure 3. The results of fingering decision for the opening part of "Romance Anonimo" (only top notes). Comparing the two tablatures, the one obtained by the minimax Viterbi algorithm avoids using the pinky finger and suppresses changing between strings. We see from the line chart that the conventional Viterbi algorithm keeps higher values at the cost of two very small values while the minimax Viterbi algorithm avoids very small values although it keeps relatively lower values.

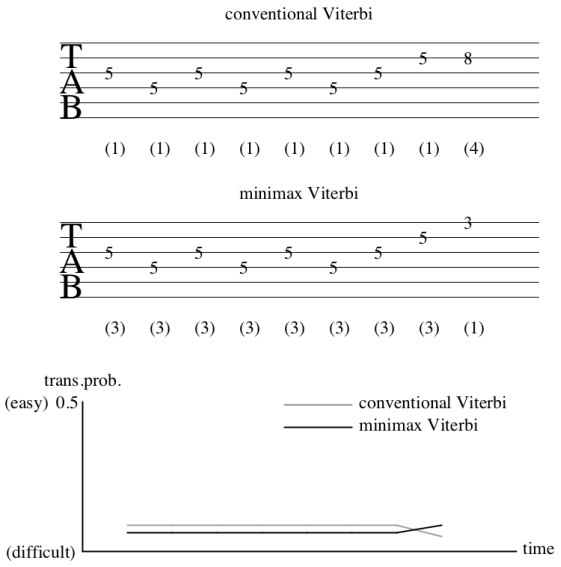


Figure 4. The results of fingering decision for the opening part of "Eine Kleine Nachtmusik" (only top notes). Comparing the two tablatures, the one obtained by the minimax Viterbi algorithm uses the ring finger (instead of the index finger) for the first eight notes to avoid using the pinky finger for the ninth note. The slight difference in the transition probability for the first eight notes comes from the difference in the difficulty of using the index finger and the pinky finger.

2 and 3, we see that the minimax Viterbi algorithm actually minimizes the maximum difficulty for playing a given phrase and makes the most difficult move easier, which can not be done by the conventional Viterbi algorithm. As for the resulting tablatures, while the one obtained by the conventional Viterbi algorithm uses the pinky finger twice and changes between strings three times, the one obtained by the minimax Viterbi algorithm does not use the pinky finger and changes between strings only once.

Figure 4 shows the results for the opening part of "Eine Kleine Nachtmusik." In both fingerings, the first eight notes are played with a single finger that presses down multiple strings across a single fret. The top tablature obtained by the conventional Viterbi algorithm uses the index finger for the first eight notes and the pinky finger for the ninth note while the middle one obtained by the minimax Viterbi algorithm prefers the ring finger for the first eight notes to avoid using the pinky finger for the ninth note. The slight difference in the transition probability for the first eight notes comes from the difference in the difficulty of the form $p_H(h_j)$ in (4) defined by the finger number h_j.

5. CONCLUSION

We have introduced a variant of the Viterbi algorithm termed the minimax Viterbi algorithm that finds the sequence of the hidden states that maximizes the minimum transition probability, and demonstrated the performance of the algorithm with guitar fingering decision based on a synthetic HMM. Fingering decision using our proposed variant has turned out to be able to minimize the maximum difficulty of the move required for playing a given phrase. We have compared the resulting fingerings by the conventional Viterbi algorithm and the minimax Viterbi algorithm to see that our proposed variant is capable of making the most difficult move easier that can not be done by the conventional one. Those observations give rise to interests in the interpolation between the conventional Viterbi algorithm and the minimax Viterbi algorithm. We consider that such an interpolation can be implemented using the L^p-norm of a real vector, which is the absolute sum of the vector elements for $p=1$ and the maximum absolute value for $p=\infty$, and is one of our future study plans. We hope that the present work draws the researcher's attention to the new "minimax paradigm" in automatic fingering decision.

6. ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Number 26240025.

7. REFERENCES

[1] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[2] Yoshua Bengio and Paolo Frasconi. An input output HMM architecture. *Advances in neural information processing systems*, 7:427–434, 1995.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[4] Gen Hori, Hirokazu Kameoka, and Shigeki Sagayama. Input-output HMM applied to automatic arrangement for guitars. *Journal of Information Processing*, 21(2):264–271, 2013.

[5] Wakana Nagata, Shinji Sako, and Tadashi Kitamura. Violin fingering estimation according to skill level based on hidden Markov model. In *Proceedings of International Computer Music Conference and Sound and Music Computing Conference (ICMC/SMC2014)*, pages 1233–1238, Athens, Greece, 2014.

[6] Eita Nakamura, Nobutaka Ono, and Shigeki Sagayama. Merged-output HMM for piano fingering of both hands. In *Proceedings of International Society for Music Information Retrieval Conference (ISMIR2014)*, pages 531–536, Taipei, Taiwan, 2014.

[7] Lawrence R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[8] Daniele P Radicioni, Luca Anselma, and Vincenzo Lombardo. A segmentation-based prototype to compute string instruments fingering. In *Proceedings of Conference on Interdisciplinary Musicology (CIM04)*, volume 17, pages 97–104, Graz, Austria, 2004.

[9] Aleksander Radisavljevic and Peter Driessen. Path difference learning for guitar fingering problem. In *Proceedings of International Computer Music Conference*, volume 28, Miami, USA, 2004.

[10] Samir I Sayegh. Fingering for string instruments with the optimum path paradigm. *Computer Music Journal*, 13(3):76–84, 1989.

[11] Daniel R Tuohy and Walter D Potter. A genetic algorithm for the automatic generation of playable guitar tablature. In *Proceedings of International Computer Music Conference*, pages 499–502, 2005.

[12] Andrew J Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

MIXTAPE: DIRECTION-BASED NAVIGATION IN LARGE MEDIA COLLECTIONS

João Paulo V. Cardoso **Luciana Fujii Pontello** **Pedro H. F. Holanda**
Bruno Guilherme **Olga Goussevskaia** **Ana Paula C. da Silva**
Computer Science Department, Universidade Federal de Minas Gerais (UFMG), Brazil
jpcardoso@ufmg.br, lucianafujii@dcc.ufmg.br, holanda@dcc.ufmg.br,
brunoguilherme@dcc.ufmg.br, olga@dcc.ufmg.br, ana.coutosilva@dcc.ufmg.br

ABSTRACT

In this work we explore the increasing demand for novel user interfaces to navigate large media collections. We implement a scalable data structure to store and retrieve similarity information and propose a novel navigation framework that uses geometric vector operations and real-time user feedback to direct the outcome. In particular, we implement this framework in the domain of music. To evaluate the effectiveness of the navigation process, we propose an automatic evaluation framework, based on synthetic user profiles, which allows to quickly simulate and compare navigation paths using different algorithms and datasets. Moreover, we perform a real user study. To do that, we developed and launched Mixtape [1] , a simple web application that allows users to create playlists by providing real-time feedback through liking and skipping patterns.

1. INTRODUCTION

Internet cloud and streaming services have become the state-of-the-art in terms of storage and access to media collections. Even though the storage problem of media collections seems to have been practically solved with cloud-based applications, a challenge still remains in conceptualizing and developing novel interfaces to explore them. User interfaces are expected to be intuitive and easy, yet flexible and powerful in understanding and delivering what users expect to see.

In this work we propose a framework that uses real-time user feedback to provide direction-based navigation in large media collections. The navigation framework is comprised of a data structure to store and retrieve similarity information and a novel navigation interface that allows users to explore the content of the collection in a personalized way. We begin by focusing on the music domain, because the intrinsic usage pattern behind listening to music is favorable to the design and verification of a dynamic real-time feedback based system.

We define media item-to-item similarity based on user-generated data, assuming that two items are similar if they frequently co-occur in a user's profile history. Media co-occurrence information is increasingly available through many online social networks. For example, in the domain of music, such usage information can be collected from Last.fm, a social music site. Collected co-occurrence data is usually sparse (not all pairs of items will have co-occurred at least once in the collected dataset) and nevertheless might occupy a lot of memory space ($\Omega(n^2)$, where n is the size of the collection). To guarantee $O(n)$ space complexity and $O(1)$ query complexity of all-pairs similarity information, we transform the collected pairwise co-occurrence values into a multi-dimensional Euclidean space, by using nonlinear dimensionality reduction [21].

Our main contribution is a novel randomized navigation algorithm, based on the geometry of the constructed similarity space. Each navigation session is modeled as a Monte Carlo simulation: given a starting item and a set of close neighbors in the similarity space, each neighbor is assigned a probability of being the *next* current item. If the returned next item is not quite what the user wants to see, they can skip it, so the previous item is used as the seed again. To define these probabilities, we propose a geometric vector-based approach, which explores the notion of direction, using user feedback and the Euclidean distances between items to establish a concept of "direction inertia", which creates a tendency for users to "keep going" in the direction of the items they enjoy and "turn away" from items, or regions, they don't like.

The evaluation of the resulting system is twofold. First, we propose an automatic evaluation framework, based on synthetic user profiles, which allows to quickly simulate and compare navigation paths using different algorithms and datasets. We also propose two basic metrics: number of skips per like ratio and smoothness of consecutively accepted items in a navigation session. Second, we evaluate real-user interaction with the system. To do that, we developed and launched Mixtape, a simple web application that allows users to create playlists by providing real-

[1] www.projectmixtape.org


© João Paulo V. Cardoso, Luciana Fujii Pontello, Pedro H. F. Holanda, Bruno Guilherme, Olga Goussevskaia, Ana Paula Couto da Silva. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** João Paulo V. Cardoso, Luciana Fujii Pontello, Pedro H. F. Holanda, Bruno Guilherme, Olga Goussevskaia, Ana Paula Couto da Silva. "Mixtape: Direction-based navigation in large media collections", 17th International Society for Music Information Retrieval Conference, 2016.

time feedback through liking and skipping patterns. Overall, we analyzed over $2,000$ simulated and $2,000$ real-user navigation sessions in a map comprised of more than $62,000$ songs. Besides analyzing quantitative parameters, such as the proportion of skipped to accepted songs and the smoothness of the generated trajectories, we gathered feedback left by users and analyzed what they expect and appreciate in a media navigation system.

2. RELATED WORK

A closely related line of research to this work is automatic playlist generation. There are techniques that use statistical analysis of radio streams [4, 5, 15, 22], are based on multidimensional metric spaces [2,4,9,13,16,17], explore audio content [3,8,14,23], and user skipping behavior [18]. In particular, Chen et al [4] model playlists as Markov chains, which are generated through the Latent Markov Embedding (LME) machine learning algorithm, using online radio streams as a training set. We use this algorithm as a baseline in our experiments. The idea to embed co-occurrence information into a multi-dimensional space has been explored before, e.g., in [1,2,9,13], where the authors focus mostly on visual exploration of a collection. The idea to use skipping behavior to generate playlists has been explored in [18], however, the presented algorithms do not scale to large collections. Our work goes beyond playlist generation, providing a real-time flexible navigation interface that receives immediate user feedback through skipping behavior to guide the user within the music collection towards directions chosen on-the-fly.

3. NAVIGATION FRAMEWORK

Our goal is to design a media navigation framework comprised of two main components: (1) A scalable data structure to store and retrieve item-to-item similarity information; (2) Directed-based navigation functions, that take the current item and user feedback in real time and return the next item; moreover, we want the navigation output to be computationally efficient and nondeterministic, so the user can be surprised with new items in each navigation sequence.

3.1 Item-to-item similarity representation

In this work, we use the assumption that similarity between two items can be deduced by analyzing usage habits of a large number of media users. More specifically, we assume that the more often two items co-occur in the same user's profile, the more similar they are. So we define pairwise similarity between two items by using cosine similarity: $cos(i,j) = coocc(i,j)/\sqrt{occ(i)occ(j)}$, where $coocc(i,j)$ is the number of co-occurrences between two items and $occ(i)$ the individual occurrences in the users' profiles.

Since co-occurrence data is typically sparse, i.e., only a few pairwise similarities are known, we applied the Isomap method [21], which extends classical multidimensional scaling (MDS) [6] by incorporating the geodesic distances imposed by an (intermediate) weighted graph. We defined the weight of an edge as the complement of the cosine similarity, $(w(i,j) = 1 - cos(i,j))$ and built a graph G with these weights.

To generate the map we calculated the complete nXn distance matrix from G and then applied the classical MDS algorithm in this matrix. Building a new d-dimensional Euclidean space such that $d << n$. The final space is a nXd matrix. Note that, for larger datasets one can use approximate algorithms, such as LMDS or LINE [7, 20].

3.2 Navigation functions

In order to guide the navigation process, a navigation session is treated as a run of a Monte Carlo simulation, in which the choice of the next item depends on the current item and a probability function that assigns different probabilities to each of its neighboring nodes. Given a starting item, the navigation system retrieves the set K of its nearest neighbors in the Euclidean space, and uses them as candidates to be the next item. Once an item $k_i \in K$ is chosen to be next, users can provide immediate feedback to the system by accepting or skipping it explicitly, through user interface feedback, or implicitly by *skipping* it. In case the new item is accepted, it becomes the current item, and the process starts again. The probability function should have a strong influence on the overall outcome of the navigation.

Parameter $|K|$ is used to vary the size of each "step" of the navigation process. It can be configured as a constant, or to be variable. In our experiments, good results were achieved using exponentially growing step size:

$$|K| = \begin{cases} 2|K|, & \text{if the previous item was skipped} \\ |K_0|, & \text{otherwise,} \end{cases}$$

where $|K_0|$ is configurable minimum neighborhood size. In our experiments we used $|K_0| = 10$ and $|K| \leq 640$.

Map navigation: We start with the following basic approach, to which we refer as Map, that explores the idea that users prefer to navigate through items that are close to each other in the Euclidean space. We define the probability of node $k_i \in K$ to be *next* as:

$$P_i^{nextMap} = \begin{cases} 1/|K|, & \text{if } k_i \in K; \\ 0, & \text{otherwise,} \end{cases}$$

Vector navigation: Vector navigation explores the notion of direction of navigation, assuming users would like to travel through different regions in the space. To do so, it treats the possible steps in the space as vectors. The *hop vector* $\vec{ab}$ of any given hop from item A to item B can be derived from the straight line between them (see Figure 1.1).

As the navigation progresses, the system keeps a *direction vector* $\vec{V}$, which is recalculated after every hop. This vector represents the directions in which the system has recently moved. As a simplified example, consider the following sequence from item A to item D (see Figure 1.2). $\vec{V_0}$ was derived from the first hop, $\vec{ab}$. $\vec{V_1}$ is half the sum of $\vec{V_0}$ and $\vec{bc}$, which was the second hop. $\vec{V_2}$ is half the sum of $\vec{V_1}$ and $\vec{cd}$, and the process goes on.

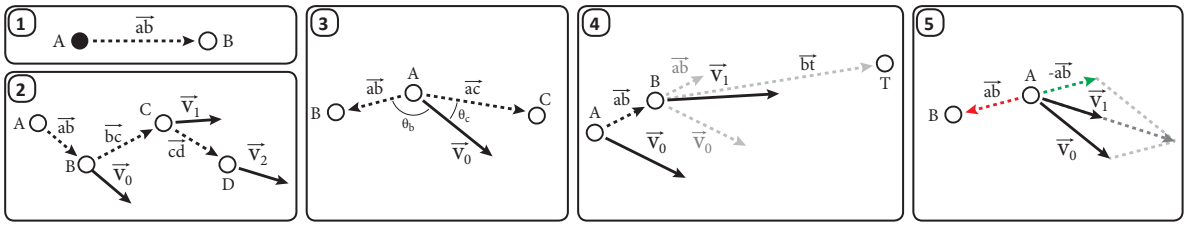


Figure 1. Direction-based navigation (Vector algorithm)

In each step, the system calculates the probabilities of suggesting each neighbor by comparing the current direction vector $\vec{V}$ to each of the vectors towards the $|K|$ considered candidates (i.e., to vectors from the current to neighbor nodes). For example, consider the decision to move from item A with current *direction vector* $\vec{V_0}$ to two neighbors, B and C (see Figure 1.3).

The more aligned the direction and the candidate vectors are, the smaller the angle θ will be, and the higher the probability of suggesting that neighbor. In Figure 1.3, since $\theta_c < \theta_b$, node C will have a higher probability to be the next item.

It is also possible to set a target destination T, which is an arbitrary point on the map, where the navigation should try to go to. This adds a third element to the direction vector update procedure, by creating vector $\vec{bt}$ in each update and also adding it to $\vec{V}$. Let's consider the situation in which a hop was accepted from item A to item B, and the target destination T was configured (see Figure 1.4). Note that $\vec{V_1}$ is made by adding vectors $\vec{V_0}$, $\vec{ab}$ and $\vec{bt}$ and dividing the module of the resulting vector by three.

Last but not least, feedback is incorporated by considering that, when a user skips a suggestion, it would be interesting to increase the probability of suggesting something different from the skipped item. So, when an item is skipped, the system does not change the current node, and the opposite vector is added to $\vec{V}$. Consider the example in Figure 1.5, where item B was skipped, and so $\vec{V_1}$ was calculated by adding $\vec{V_0}$ to $-\vec{ab}$ to reflect the user's preference.

Defining the method formally: Consider a set K of closest neighbors of *current node* A and the current *direction vector* $\vec{V}$ with the respective angles θ_i between $\vec{V}$ and each hop vector $a\vec{k_i}, k_i \in K$. Also, consider the optional parameter with the location of a target destination T. We define the direction-based navigation function using the following weight variables:

$$w_i = 1 + cos(\theta_i) = 1 + \frac{a\vec{k_i} \bullet \vec{V}}{|a\vec{k_i}||\vec{V}|}.$$

Note that the weight is proportional to the cosine of the angle between the current direction vector $\vec{V}$ and the direction of each neighbor k_i relative to the current node A. We add 1 to avoid negative values. Finally, we define the probability of node $k_i \in K$ to be *next* as:

$$P_i^{nextVec} = \frac{w_i}{\sum_{j=1}^{|K|} w_j}.$$

Note that we have a proper probability distribution, since the sum over probabilities $P_i^{nextVec}, i \in K$ is 1.

After the next item has been returned, say k_i, and the user has provided feedback by accepting or skipping it, we update the direction vector $\vec{V}$ of node A as follows:

$$\vec{V} = \begin{cases} (a\vec{k_i} + \vec{V})/2, & \text{if } k_i \text{ was accepted} \\ (-a\vec{k_i} + \vec{V})/2, & \text{if } k_i \text{ was skipped.} \end{cases}$$

If target destination T has been defined, then the calculation also includes the new target vector $\vec{at}$ between the chosen item and the target destination:

$$\vec{V} = \begin{cases} (a\vec{k_i} + \vec{V} + \vec{at})/3, & \text{if } k_i \text{ was accepted} \\ (-a\vec{k_i} + \vec{V} + \vec{at})/3, & \text{if } k_i \text{ was skipped.} \end{cases}$$

If the item was accepted, node k_i becomes the next current node. Otherwise, the current node does not change, and only the direction vector is updated. Note that this approach is domain-independent and uses nothing but the coordinates of the embedding itself. It also carries an explicit dependency on user feedback, since $\vec{V}$ is determined by the user's skipping behavior.

4. MUSIC DOMAIN

The navigation framework described in Section 3 can be applied to different media domains. In this work, we focus on the domain of music.

4.1 Last.fm Dataset

In order to define music similarity, we assume that the more frequently two songs co-occur in a user's listening history, the more similar they are. We collected co-occurrence data from `Last.fm`, a social music site that tracks user musical tastes, from November, 2014 to March, 2015. More specifically, we collected the top-25 most listened songs of each user, reaching a total of 372,899 users, 2,060,173 tracks, and 374,402 artists. Moreover, we also collected a total of 1,006,236 user-generated tags, associated with songs. In particular, 75% of songs have had at least one associated tag in our dataset. We considered a subset of 983,010 tracks in our dataset with a known MBID [2], from which we selected another subset of 83,180 tracks that co-occurred 5 or more times, forming a connected component of 62,352 songs. A detailed characterization of the dataset can be found in [19].

[2] MusicBrainz Identifier (MBID) is a reliable and unambiguous form of music identification (`musicbrainz.org`).

The connected graph with 62,352 vertices enabled us to run IsoMap [21] and Multidimensional Scaling (MDS) [6] without any approximations. By parallelizing parts of the algorithm, we computed the all-pairs shortest path matrix of size $62,352 \times 62,352$, in 7 minutes on a server with 50 GB of RAM and 16 CPU cores, and computed the embedding into 100 dimensions in approximately 2 hours on the same server. Note that a larger collection could have been embedded using a less computationally intensive approximate algorithm, such as LMDS or LINE [7, 20]. An evaluation of the embedding process can be found in [12].

4.2 Mixtape

We wanted to collect real user feedback in order to evaluate the navigation framework in the music domain. For this purpose, we developed Mixtape, a web-based application with a simple user interface (see Figure 2). On the server side, a k-d tree was loaded with a 100-dimensional space of $62,352$ tracks. On the client side, the design goal was to provide a minimalist user interface that fully explored the navigation functions defined in Section 3.2. Each user would choose the starting song and then be presented with one suggestion at a time, with explicit feedback-generating actions. The user interface is comprised of a playlist, on the left, which shows the songs the user has accepted or skipped, and a Youtube video window, which finds and plays the current suggested item. Users can then decide whether they like the song or not, using the *like* and *dislike* buttons, one of which the user must press in order to receive the next suggestion. In case the user does not press anything and listens to the entire song, we assume they liked it, and consider the song as accepted. There is also a settings button, which allows the user to switch between different navigation functions.

5. EXPERIMENTS

The evaluation of the proposed navigation framework in the domain of music is twofold: Firstly, we propose an automatic evaluation framework and perform an extensive analysis based on simulated user profiles. Furthermore, since real users might behave differently, and the perception of a song is subjective, we observed how real users interacted with our Mixtape application. As a result, we were able to evaluate not only how effective and engaging the proposed navigation system is, but also how well the simulated user profiles approximated real user behavior.

5.1 Simulated user profiles

To test the navigation framework, we simulated synthetic user profiles, in which hypothetical users intend to listen to 20 songs (about one hour of music), and count the number of skips (songs that are skipped by the simulated user profile following the algorithm described below) until 20 songs are accepted. A similar evaluation approach was used in [18]. We simulated two types of users:

Tag-based user profile: This user profile is based on tag information and the notion of transition between two regions on the map. Recall from Section 4.1 that we collected over 1,006,236 user-generated tags, associated with songs. We assume this user wishes to listen to a sequence of songs that transitions from initial tag T_i to final tag T_f.

To do that, the simulated user accepts all songs associated with tag T_i in the first $1/3$ of the navigation path (skips otherwise), accepts songs with tags T_i or T_f in the second $1/3$, and accepts only songs with tag T_f in the last $1/3$ of the path, comprised of a total of 20 items. Note that real users do not necessarily know what tags are associated to particular tracks. Since these users are hypothetical, we can use the collected tag information for simulation purposes.

We manually selected tag transitions among the top 200 most popular tags in our dataset. We noticed these tags could be divided in three categories: Mood tags, such as *Chill, Upbeat, Relaxing*, Genre tags, such as such as *Rock, Hip Hop, Folk*, and Age tags, such as *60's, 90's, 2000's*. We then paired them up manually, selecting 14 transitions to experiment with. For each tag transition ($T_i \Rightarrow T_f$), we considered a navigation path starting at the most popular song associated to tag T_i, and applying the skipping rule until a path of 20 accepted songs was achieved.

Artist-based user profile: This user profile is based on artist information and the notion that certain users wish to listen to songs by artists they already know. Since this user wishes to listen to preferred artists, whenever the suggested song is by an artist contained in the user's history, it is accepted. Otherwise, it is skipped. We collected the complete listening histories of 20 users to simulate this user profile, and started the playlist at a random song within each user's profile. Moreover, for this experiment only users whose profiles were *not* used to construct the embedding were simulated.

5.2 Baselines

As baselines, we tested the following approaches:

LME: Logistic Markov Embedding [4, 16], a probabilistic approach that models sequences in a Euclidean space using radio streams as a training set. We used the implementation available at the authors' homepage, with all parameters set to default values, except for $\alpha = 5$ (this value resulted in superior overall performance), as our dataset did not have music sequences, only music occurrences in a user profile, we used the "yes-complete" dataset (also made available by the authors) in a combination with our dataset, since LME needs a sequence of items as input, which resulted in an intersection set of 31,544 items with our dataset. We made one modification to the LME algorithm by incorporating user feedback when computing the next item. More specifically, whenever an item n_j has been skipped after a previously accepted item n_i, we recompute the probabilities at n_i setting $Pr(n_j|n_i) = 0$, and maintaining n_i as the current item.

Random: A random song is returned, considering all songs in the dataset.

Random Tag: A random song with tag T is returned. This baseline was used for the tag-based navigation evaluation.

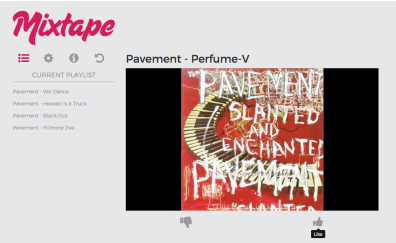


Figure 2. Mixtape screenshot

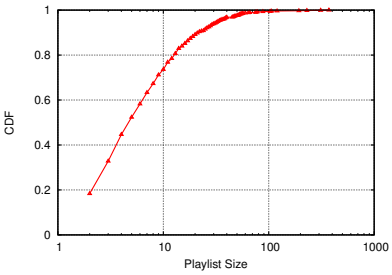


Figure 3. Mixtape user study: playlist length distribution

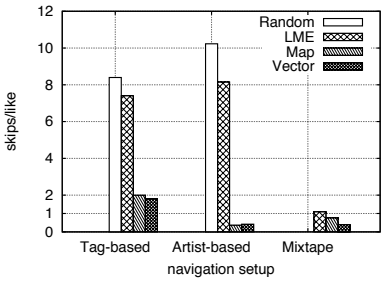


Figure 4. Total number of skips per like ratio: simulated versus real-user profiles

5.3 Mixtape user study setup

We collected all user actions on Mixtape over a course of 2 weeks, resulting in the participation of over 800 users, generating a total of over 2000 navigation sessions. In order to compare the performance of different navigation algorithms, each navigation session was randomly assigned either Map, Vector or LME algorithms (with undefined tag parameters), but the user could explicitly change the algorithm in the settings menu as well. Users could also choose the Random approach, however, since the engagement in this setting was very low, we did not include it in the plots.

5.4 Results

In our experiments, we measure the effectiveness of the two navigation approaches proposed in this work (Map and Vector) and the two baselines (Random and LME) in three navigation setups: simulated tag-based user profiles, simulated artist-based user profiles, and real users on Mixtape. We use two main metrics: skipping behavior and playlist smoothness, defined below. Each scenario was executed 20 times, and all figures show the 95% confidence interval.

Skipping behavior: Figure 3 shows the CDF of playlist length generated on Mixtape. It can be seen that almost 30% of the playlists [3] contain 10 tracks or more, and almost 15% have size 20 or longer, which shows that many people really engaged with the application.

In Figure 4 we compare the ratio of the total number of skips (dislikes) and the total number of accepted songs (likes) in playlists generated by all navigation algorithms for simulated and real user profiles. Analyzing the simulated user profiles, we can see that the baseline algorithms present several times more skips per like (LME: $skips/like > 7.5$, Random: $skips/like > 8$) than Map and Vector ($skips/like < 2$). Map and Vector have similar results and perform especially well in the artist-based simulated setup ($skips/like < 0.5$), which shows they are more effective not only in directing the user between different regions in the space, but also in presenting the user with music by preferred artists.

Looking at Mixtape results on Figure 4, we can see that all three approaches perform well on average ($skips/like < 2$). LME has still more skips than likes ($skips/like > 1$), whereas Map and Vector have significantly more likes than skips (Map: $skips/like < 0.8$, Vector: $skips/like < 0.4$), indicating that users enjoyed the vast majority of the suggested songs, especially by the Vector algorithm. Note that Vector outperforms Map for real users, indicating that the direction in the map, provided by the real-time feedback, does matter for real users.

Comparing real and synthetic user profiles, we note that LME performed much better with real rather than simulated users. That might be because real users are more open-minded and accept more diversity in their playlists. Nevertheless, the number of skips per like for Vector and Map on Mixtape was similar to the simulated artist-based user profiles, indicating that in some aspects the simulation was accurate in portraying a real user.

In Figures 5 through 7 we analyze the number of skips along each step of the navigation process. In Figure 5 the number of skips per step decreases in the second third and then reaches a maximum in the beginning of the third part of the playlist for all algorithms. This illustrates how the algorithms react to the simulated tag-based navigation setup. Afterwards, however, we can see that Map and Vector quickly decrease the number of skips, as opposed to LME and Random, showing that the former algorithms succeed in adjusting the direction of the navigation towards the destination tag.

Playlist smoothness: In Figures 8 through 10 we analyze how similar consecutive songs are on a navigation path, by measuring the cosine similarity of the artists of consecutive (accepted) songs. [4] Note that in Figure 8 we plot the RandomTag baseline instead of Random, to shed light on the following question: if the objective of tag-based simulations is to recommend songs with a given tag, why not simply choose songs from the database that have that tag? That method might work when we ignore the relationship between songs in a playlist. However, we argue that a playlist should be more than a group of songs with a given tag–it should present a relationship between the songs. We can see that RandomTag and LME baselines provide almost zero similarity along the navigation path, even though RandomTag only returns songs with accepted tags, i.e.,

[3] We refer to the sequence of tracks accepted, or liked, by a user in one navigation session as a *playlist*.

[4] We define artist similarity as the cosine similarity computed from artist co-occurrence in our dataset.

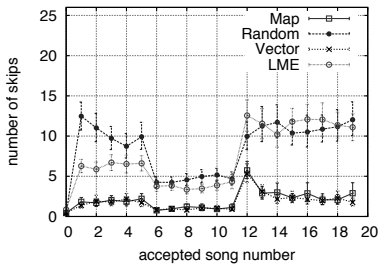


Figure 5. Skips along playlists: simulated tag-based navigation

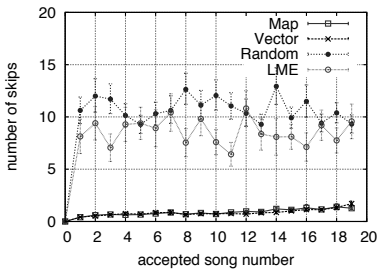


Figure 6. Skips along playlists: simulated artist-based navigation

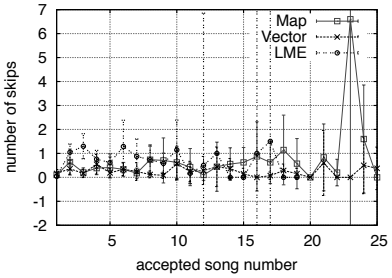


Figure 7. Skips along playlists: real-user navigation (Mixtape)

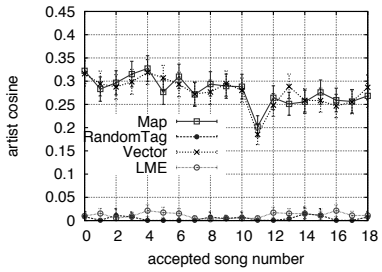


Figure 8. Playlist smoothness: simulated tag-based navigation

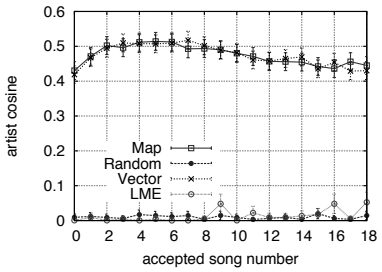


Figure 9. Playlist smoothness: simulated artist-based navigation

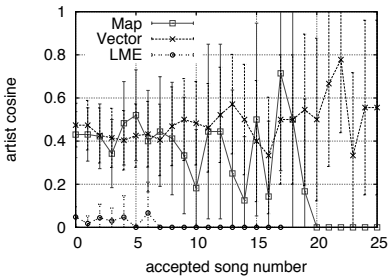


Figure 10. Playlist smoothness: real-user navigation (Mixtape)

makes zero skips in the tag-based simulation setup. Map and Vector, on the other hand, trace highly smooth navigation paths, offering the user songs with high similarity to the previously chosen songs, especially in the artist-based simulation setup (Figure 9, artist cosine > 0.4). Figure 10 [5] shows that Map and, especially, Vector playlists on Mixtape also present high similarity between consecutive items, indicating that people prefer smooth, rather than abrupt, transitions in their navigation paths.

User feedback: To enhance our perception about how users perceive our Mixtape application, we created a short online survey, linked from the Mixtape application, which was answered by 44 unidentified subjects. The users were not provided with any information about the navigation algorithms. They were asked to provide feedback on the experience of using Mixtape, leading to the following numbers: 95% enjoyed the songs suggested by Mixtape, and only 11% of the users were not able to find most of the songs they were searching for (recall from Section 4.1 that we used a reduced sample of the map in our experiments: 62,352 songs with co-occurrence at least 5.).

Interestingly, 70% of the participants said they discovered new artists or songs. Most people said they didn't change the navigation policy and they didn't know which policy they used during their navigation. From those who did experiment with different policies, they equally enjoyed Map and Vector approaches (even though, on average, users skipped less songs when using direction-based navigation).

[5] The CIs in Figures 7 and 10 are high, due to insufficient data for certain song numbers.

To sum up this first user study, we can conclude that users enjoyed navigating music collections by giving their real-time feedback and that the navigation allowed them to discover previously unknown songs they enjoyed.

6. CONCLUSION

In this work we proposed a navigation framework for large media collections and evaluated an implementation of the framework in the domain of music. Potentially, the same ideas could be applied to other kinds of media, e.g. movies or TV shows [10, 11]. Rather than creating fixed playlists, our approach allows users to provide feedback through skipping behavior and direct the navigation process in real-time. We evaluated the framework through simulation of more that 2,000 synthetic navigation paths and performed a real user study by launching Mixtape, a web application with a minimalist user interface that allows people to navigate a collection of over 60,000 music tracks. We analyzed over 2,000 playlists generated by over 800 real users and received positive feedback about the application. When comparing playlists generated by Mixtape and simulated hypothetical users, we could observe several similarities, indicating that in some aspects the simulation was accurate in portraying a real user. Moreover, not only did this user study serve as validation of the proposed framework, but it also provided insights into what users look for and appreciate in a media navigation system. [6]

[6] Acknowledgments: This work is supported in part by CNPq, FAPEMIG and LG Electronics in cooperation with Brazilian Federal Government through Brazilian Informatics Law (n. 8.2.48/1991).

7. REFERENCES

[1] E. Bernhardsson. Systems and methods of selecting content items using latent vectors, August 18 2015. US Patent 9,110,955.

[2] Lukas Bossard, Michael Kuhn, and Roger Wattenhofer. Visually and Acoustically Exploring the High-Dimensional Space of Music. In *IEEE International Conference on Social Computing (SocialCom), Vancouver, Canada*, August 2009.

[3] Pedro Cano, Markus Koppenberger, and Nicolas Wack. Content-based music audio recommendation. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 211–212. ACM, 2005.

[4] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *18th ACM SIGKDD*, 2012.

[5] Shuo Chen, Jiexun Xu, and Thorsten Joachims. Multi-space probabilistic sequence modeling. In *19th ACM SIGKDD*, 2013.

[6] Trevor F. Cox and M.A.A. Cox. *Multidimensional Scaling*. Chapman and Hall/CRC, 2000.

[7] Vin De Silva and Joshua B Tenenbaum. Sparse multidimensional scaling using landmark points. Technical report, Technical report, Stanford University, 2004.

[8] Arthur Flexer, Dominik Schnitzer, Martin Gasser, and Gerhard Widmer. Playlist generation using start and end songs. In Juan Pablo Bello, Elaine Chew, and Douglas Turnbull, editors, *ISMIR*, pages 173–178, 2008.

[9] Olga Goussevskaia, Michael Kuhn, Michael Lorenzi, and Roger Wattenhofer. From web to map: Exploring the world of music. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08*, volume 1, 2008.

[10] Pedro Holanda, Bruno Guilherme, Joao Paulo V. Cardoso, Ana Paula Couto da Silva, and Olga Goussevskaia. Mapeando o universo da midia usando dados gerados por usuarios em redes sociais online. In *The 33rd Brazilian Symposium on Computer Networks and Distributed Systems (SBRC)*, 2015.

[11] Pedro Holanda, Bruno Guilherme, Ana Paula Couto da Silva, and Olga Goussevskaia. TV goes social: Characterizing user interaction in an online social network for TV fans. In *Engineering the Web in the Big Data Era - 15th International Conference, ICWE 2015, Rotterdam, The Netherlands, June 23-26, 2015, Proceedings*, pages 182–199, 2015.

[12] Pedro Holanda, Bruno Guilherme, Luciana Fujii Pontello, Ana Paula Couto da Silva, and Olga Goussevskaia. Mixtape application: Music map methodology and evaluation. Technical report, Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brazil, May 2016.

[13] Michael Kuhn, Roger Wattenhofer, and Samuel Welten. Social audio features for advanced music retrieval interfaces. In *Proceedings of the international conference on Multimedia*, pages 411–420. ACM, 2010.

[14] Beth Logan. Content-based playlist generation: Exploratory experiments. In *ISMIR*, 2002.

[15] François Maillet, Douglas Eck, Guillaume Desjardins, and Paul Lamere. Steerable playlist generation by learning song similarity from radio station playlists. In *ISMIR*, 2009.

[16] Joshua L Moore, Shuo Chen, Thorsten Joachims, and Douglas Turnbull. Learning to embed songs and tags for playlist prediction. In *ISMIR*, pages 349–354, 2012.

[17] Joshua L Moore, Shuo Chen, Douglas Turnbull, and Thorsten Joachims. Taste over time: The temporal dynamics of user preferences. In *ISMIR*, 2013.

[18] Elias Pampalk, Tim Pohle, and Gerhard Widmer. Dynamic playlist generation based on skipping behavior. In *ISMIR*, 2005.

[19] Luciana Fujii Pontello, Pedro Holanda, Bruno Guilherme, Joao Paulo V. Cardoso, Olga Goussevskaia, and Ana Paula Couto da Silva. Mixtape application: Last.fm data characterization. Technical report, Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brazil, May 2016.

[20] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *24th International Conference on World Wide Web*, pages 1067–1077, 2015.

[21] J. B. Tenenbaum, V. Silva, and J. C. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323, 2000.

[22] Douglas R Turnbull, Justin A Zupnick, Kristofer B Stensland, Andrew R Horwitz, Alexander J Wolf, Alexander E Spirgel, Stephen P Meyerhofer, and Thorsten Joachims. Using personalized radio to enhance local music discovery. In *CHI'14*, 2014.

[23] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*, 2013.

MUSICAL NOTE ESTIMATION FOR F0 TRAJECTORIES OF SINGING VOICES BASED ON A BAYESIAN SEMI-BEAT-SYNCHRONOUS HMM

Ryo Nishikimi[1] **Eita Nakamura**[1] **Katsutoshi Itoyama**[1] **Kazuyoshi Yoshii**[1]
[1] Graduate School of Informatics, Kyoto University , Japan
{nishikimi, enakamura}@sap.ist.i.kyoto-u.ac.jp, {itoyama, yoshii}@kuis.kyoto-u.ac.jp

ABSTRACT

This paper presents a statistical method that estimates a sequence of discrete musical notes from a temporal trajectory of vocal F0s. Since considerable effort has been devoted to estimate the frame-level F0s of singing voices from music audio signals, we tackle musical note estimation for those F0s to obtain a symbolic musical score. A naïve approach to musical note estimation is to quantize the vocal F0s at a semitone level in every time unit (*e.g.*, half beat). This approach, however, fails when the vocal F0s are significantly deviated from those specified by a musical score. The onsets of musical notes are often delayed or advanced from beat times and the vocal F0s fluctuate according to singing expressions. To deal with these deviations, we propose a Bayesian hidden Markov model that allows musical notes to change in semi-synchronization with beat times. Both the semitone-level F0s and onset deviations of musical notes are regarded as latent variables and the frequency deviations are modeled by an emission distribution. The musical notes and their onset and frequency deviations are jointly estimated by using Gibbs sampling. Experimental results showed that the proposed method improved the accuracy of musical note estimation against baseline methods.

1. INTRODUCTION

Singing voice analysis is one of the most important topics in the field of music information retrieval because singing voice usually forms the melody line of popular music and it has a strong impact on the mood and impression of a musical piece. The widely studied tasks in singing voice analysis are fundamental frequency (F0) estimation [1, 4, 5, 8, 10, 15, 22] and singing voice separation [9, 13] for music audio signals. These techniques can be used for singer identification [11, 23], Karaoke systems based on singing voice suppression [2, 20], and a music listening system that helps a user focus on a particular musical element (*e.g.*, vocal part) for deeper music understanding [7].

© Ryo Nishikimi, Eita Nakamura, Katsutoshi Itoyama, Kazuyoshi Yoshii. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Ryo Nishikimi, Eita Nakamura, Katsutoshi Itoyama, Kazuyoshi Yoshii. "Musical Note Estimation for F0 Trajectories of Singing Voices based on a Bayesian Semi-beat-synchronous HMM", 17th International Society for Music Information Retrieval Conference, 2016.

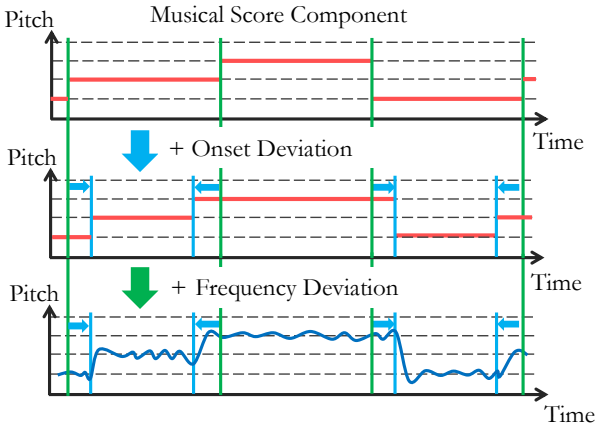


Figure 1: The process generating F0 trajectories of singing voices.

In this study we tackle a problem called *musical note estimation* that aims to recover a sequence of musical notes from an F0 trajectory of singing voices. While a lot of effort has been devoted to F0 estimation of singing voices, musical note estimation should be investigated additionally to complete automatic music transcription, *i.e.*, convert the estimated F0 trajectory to a musical score containing only discrete symbols. If beat information is available, a naïve approach to this problem is to quantize the vocal F0s contained in every time unit (*e.g.*, half beat) into a semitone-level F0 with a majority vote [7]. This approach, however, often fails to work when the vocal F0s are significantly deviated from exact semitone-level F0s specified by a musical score or the melody is sung in a tight or lazy singing style such that the onsets of musical notes are significantly advanced or delayed from exact beat times.

To solve this problem, we propose a statistical method based on a hidden Markov model (HMM) that represents how a vocal F0 trajectory is generated from a sequence of latent musical notes (Fig. 1). The F0s of musical notes in a musical score can take only discrete values with the interval of semitones and tend to vary at a beat, half-beat, or quarter-beat level. The vocal F0 trajectory in an actual performance, on the other hand, is a continuous signal that can dynamically and smoothly vary over time. To deal with both types of F0s from a generative viewpoint, we formulate a semi-beat-synchronous HMM (SBS-HMM) allowing the continuous F0s of a sung melody to deviate from the discrete F0s of written musical notes along the time and frequency directions. In the proposed HMM, the semitone-level F0s and onset deviations of musical notes are encoded

as latent variables and the F0 deviations of musical notes are modeled by emission probability distributions. Given an F0 trajectory and beat times, all the variables and distributions are estimated jointly using Gibbs sampling.

2. RELATED WORK

This section introduces related work on singing voices.

2.1 Pitch Estimation of Singing Voice

Many studies on estimating a vocal F0 trajectory in a music audio signal have been conducted [1,4,5,8,10,15,22]. Subharmonic summation (SHS) [8] is a method in which the fundamental frequency for each time is determined by calculating the sum of the powers of the harmonic components of each candidate fundamental frequency $\{f_0, \ldots, f_M\}$. PreFEst [5] is a method that estimates the F0 trajectories of a melody and a bass line by extracting the most predominant harmonic structure from a polyphonic music audio signal. Ikemiya et al. [10] proposed a method in which singing voice separation and F0 estimation are performed mutually. First a singing voice is separated, from the spectrogram obtained by the short-time Fourier transform (STFT) for a music audio signal, by using a robust principal component analysis (RPCA) [9], and then a vocal F0 trajectory is obtained with the Viterbi algorithm by using SHS for a separated singing voice. Salamon et al. [22] used the characteristics of vocal F0 contours for melody extraction. Durrieu et al. [4] proposed a method for melody extraction in which the main melody is represented as a source-filter model and the accompaniment of the music mixture is represented as a non-negative matrix factorization (NMF)-based model. De Cheveigné et al. [1] proposed a autocorrelation based method for fundamental frequency estimation which is expanded to decrease an error rate. This method is called YIN. Mauch et al. [15] extended YIN in a probabilistic way to output multiple pitch candidates. This method is called pYIN.

2.2 Note Estimation of Singing Voice

A method for estimating the sequence of musical notes by quantizing pitches of a vocal F0 trajectory has been proposed. A majority-vote method described in Sec. 1 was implemented in Songle [7]. The method has a limit because it doesn't consider the singing expression nor the typical occurrence of pitches in succession. Paiva et al. [17] proposed a method that has five stages and detects melody notes in polyphonic musical signals, and Raphael [19] proposed an HMM-based method that simultaneously estimates rhythms, tempos, and notes from a solo singing voice acoustic signal. Poliner et al. [18] proposed a method based on a support vector machines (SVM) classifier which doesn't need the assumption that a musical pitch is realized as a set of harmonics of a particular fundamental. Laaksonen [12] proposed a melody transcription method that uses chord information, and Ryynänen et al. [21] proposed a method for transcribing the melody, bass line, and chords in polyphonic music. A software tool called Tony developed by Mauch et al. [14] estimates musical notes from the output of pYIN by Viterbi-decoding of an HMM.

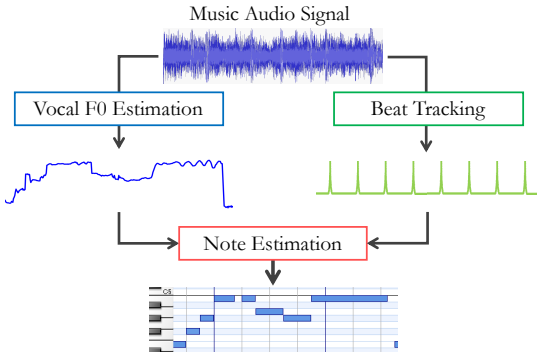


Figure 2: Overview of the proposed musical note estimation method based on a semi-beat-synchronous HMM.

2.3 Analysis of Vocal F0 Trajectories

Studies on extracting the personality and habit of singing expression from vocal F0 trajectories have been conducted. Ohishi et al. [16] proposed a model that represents the generating process of vocal F0 trajectories in consideration of the time and frequency deviations. In that model the vocal F0 trajectory consists of three components: note, expression, and fine deviation components. The note component contains the note transition and overshoot, and the expression component contains vibrato and portamento. The note and expression components are represented as the outputs of second-order linear systems driven by the note and expression commands. The note and expression commands represent the sequence of musical notes and the musical expressive intentions, respectively. The note command and the expression command are represented with HMMs. Although the method can extract the personality of the singing expression from vocal F0 trajectories, it assumes that the music score is given in advance and cannot be directly applied for note estimation.

3. PROPOSED METHOD

This section explains the proposed method for estimating a sequence of latent musical notes from the observed vocal F0 trajectories by formulating an SBS-HMM which represents the generating process of the observations. An observed F0 trajectory is stochastically generated by imparting frequency and onset deviations to a step-function-like F0 trajectory that varies exactly on a 16th-note-level grid according to a music score. The semitone-level F0s (called pitches for simplicity in this paper) between adjacent grid lines and the onset deviations are represented as latent variables (states) of the HMM. Since the frequency deviations are represented by emission probability distributions of the HMM, a semi-beat-synchronous step-function-like F0 trajectory is generated in the latent space and its finely-fluctuated version is then generated in the observed space.

3.1 Problem Specification

The problem of musical note estimation is formally defined (Fig. 2) as follows:

Input: a vocal F0 trajectory $\boldsymbol{X} = \{x_t\}_{t=1}^{T}$ and 16th-note-level beat times $\boldsymbol{\psi} = \{\psi_n\}_{n=1}^{N}$ automatically estimated from a music audio signal.
Output: a sequence of pitches $\boldsymbol{Z} = \{z_n\}_{n=1}^{N}$.

Here, t is the time frame index, T is the number of time frames in the target signal, x_t indicates a log-frequency in cents at frame t, N is the number of beat times, ψ_n is the n-th beat time, and z_n indicates a pitch between ψ_{n-1} and ψ_n taking one of $\{\mu_1, \ldots, \mu_K\}$, where K is the number of kinds of pitches that can appear in a music score. The beginning and end of music are represented as $\psi_0 = 1$ and $\psi_N = T{+}1$ respectively. In this paper we assume that each z_n corresponds to a 16th note for simplicity. A longer musical note is represented by a subsequence of $\{z_n\}_{n=1}^{N}$ having the same pitch in succession.

3.2 Model Formulation

We explain how to formulate the SBS-HMM that simultaneously represents the pitch transitions and onset and frequency deviations of musical notes.

3.2.1 Modeling Pitch Transitions

A sequence of latent pitches $\boldsymbol{Z}$ forms a first-order Markov chain given by

$$z_n | z_{n-1}, \boldsymbol{A} \sim \mathrm{Categorical}(z_n | \boldsymbol{a}_{z_{n-1}}), \tag{1}$$

where $\boldsymbol{A} = [\boldsymbol{a}_1^T, \cdots, \boldsymbol{a}_K^T]$ is a K-by-K transition probability matrix such that $\sum_{k=1}^{K} a_{jk} = 1$ for any j. The initial latent state z_1 is determined as follows:

$$z_1 | \boldsymbol{\pi} \sim \mathrm{Categorical}(z_1 | \boldsymbol{\pi}), \tag{2}$$

where $\boldsymbol{\pi} = [\pi_1, \cdots, \pi_K]^T$ is a K-dimensional vector such that $\sum_{k=1}^{K} \pi_k = 1$.

3.2.2 Modeling Onset Deviations

The onset deviations of musical notes $\boldsymbol{\tau} = \{\tau_n\}_{n=1}^{N}$ are represented as discrete latent variables taking integer values between $-G$ and G. Let $\phi_n = \psi_n + \tau_n$ be the actual onset time of the n-th musical note. Note that $\tau_0 = 0$ and $\tau_N = 0$ at the beginning and end of a vocal F0 trajectory. We assume that τ_n is stochastically generated as follows:

$$\tau_n | \boldsymbol{\rho} \sim \mathrm{Categorical}(\tau_n | \boldsymbol{\rho}), \tag{3}$$

where $\boldsymbol{\rho} = [\rho_{-G}, \ldots, \rho_G]^T$ is a $(2G{+}1)$-dimensional vector such that $\sum_{g=-G}^{G} \rho_g = 1$.

3.2.3 Modeling Frequency Deviations

The observed F0 x_t $(\phi_{n-1} \leq t < \phi_n)$ is stochastically generated by imparting a probabilistic frequency deviation to the semitone-level pitch μ_{z_k} assigned to each beat interval. Assuming that x_t is independently generated at each frame, the emission probability of the n-th beat interval in the case of $z_n = k, \tau_{n-1} = f, \tau_n = g$ is given by

$$b_{nkfg} \equiv \left\{ \prod_{t=\phi_{n-1}}^{\phi_n - 1} p(x_t | z_n = k) \right\}^{\frac{1}{\phi_n - \phi_{n-1}}}, \tag{4}$$

where $p(x_t | z_n = k)$ is the emission probability of each frame. To balance the effects of transition probabilities and emission probabilities, we exponentiate the product of emission probabilities of frames in a beat interval by the number of frames in a beat interval. We use as $p(x_t | z_n)$ the Cauchy distribution, which is robust against outliers and is defined by

$$\mathrm{Cauchy}(x; \mu, \lambda) = \frac{\lambda}{\pi \{(x - \mu)^2 + \lambda^2\}}, \tag{5}$$

where μ is a location parameter that defines the mode of the distribution and λ is a scale parameter. When the pitch of the n-th beat interval is $z_n = k$, μ takes the value μ_k. The scale parameter takes a value that does not depend on the pitch z_n.

Since actual vocal F0s are significantly deviated from those specified by a musical score, the scale parameter of a Cauchy distribution is allowed to change according to the difference of adjacent F0s; *i.e.*, $\Delta x_t \equiv x_t - x_{t-1}$. The scale parameter is set to be proportional to the absolute value of Δx_t and defined for each frame t as follows:

$$\lambda_t = c \cdot |\Delta x_t| + d, \tag{6}$$

where c is an coefficient and d is a constant term. If $d = 0$, $p(x_t | z_n)$ cannot be calculated when $\lambda_t = 0$ and $\Delta x_t = 0$. To avoid this problem, we introduce the constant term d.

3.2.4 Incorporating Prior Distributions

We put conjugate Dirichlet priors on model parameters $\boldsymbol{A}$, $\boldsymbol{\pi}$, and $\boldsymbol{\rho}$ as follows:

$$\boldsymbol{a}_j \sim \mathrm{Dirichlet}(\boldsymbol{a}_j | \boldsymbol{\xi}_j), \tag{7}$$

$$\boldsymbol{\pi} \sim \mathrm{Dirichlet}(\boldsymbol{\pi} | \boldsymbol{\zeta}), \tag{8}$$

$$\boldsymbol{\rho} \sim \mathrm{Dirichlet}(\boldsymbol{\rho} | \boldsymbol{\eta}), \tag{9}$$

where $\boldsymbol{\xi}_j = [\xi_1, \ldots, \xi_K]^T$ and $\boldsymbol{\zeta} = [\zeta_1, \ldots, \zeta_K]^T$ are K-dimensional vectors and $\boldsymbol{\eta} = [\eta_{-G}, \ldots, \eta_G]^T$ is a $(2G + 1)$-dimensional vector.

We then put on gamma priors on nonnegative Cauchy parameters c and d as follows:

$$c \sim \mathrm{Gamma}(c | c_0, c_1), \tag{10}$$

$$d \sim \mathrm{Gamma}(d | d_0, d_1), \tag{11}$$

where c_0 and d_0 are shape parameters and c_1 and d_1 are rate parameters.

3.3 Bayesian Inference

The goal of Bayesian inference is to calculate the posterior distribution $p(\boldsymbol{Z}, \boldsymbol{\tau}, \boldsymbol{A}, \boldsymbol{\pi}, \boldsymbol{\rho}, c, d | \boldsymbol{X})$. Since this computation is analytically intractable, we use Markov chain Monte Carlo (MCMC) methods for sampling the values of those variables. Let $\boldsymbol{\Theta} = \{\boldsymbol{A}, \boldsymbol{\pi}, \boldsymbol{\rho}\}$ be a set of model parameters. To get samples of $\boldsymbol{\Theta}$, the Gibbs sampling algorithm is used. To get samples of sequential latent variables $\boldsymbol{Z}$ and $\boldsymbol{\tau}$, on the other hand, a kind of blocked Gibbs sampling algorithms called a forward filtering-backward sampling algorithm is used. These steps are iterated in a similar way

to an expectation maximization (EM) algorithm called the Baum-Welch algorithm used for unsupervised learning of an HMM. Since the conjugacy is not satisfied for the distributions regarding c and d, we use the Metropolis-Hastings (MH) algorithm.

3.3.1 Inferring Latent Variables $\boldsymbol{Z}$ and $\boldsymbol{\tau}$

We explain how to sample a sequence of latent variables $\boldsymbol{Z}$ and $\boldsymbol{\tau}$. For each beat interval, we calculate the probability given by

$$\beta_{njf} = p(z_n = j, \tau_n = f | z_{n+1:N}, \tau_{n+1:N}, x_{1:T}), \quad (12)$$

where $z_{n+1:N}$, $\tau_{n+1:N}$, and $x_{1:T}$ represent $z_{n+1}, \ldots, z_N$, $\tau_{n+1}, \ldots, \tau_N$, and $x_1, \ldots, x_T$, respectively. The latent variables of the n-th beat interval (z_n, τ_n) are sampled in accordance with β_{njf}. The calculation of Eq. (12) and the sampling of the latent variables are performed by using forward filtering-backward sampling.

In forward filtering, we recursively calculate the probability given by

$$\alpha_{nkg} = p(X_{10\tau_1}, \ldots, X_{(n-1)\tau_{n-2}f}, X_{nfg}, z_n{=}k, \tau_n{=}g),$$

where X_{nfg} represents the observations x_t in the beat interval from ϕ_{n-1} to ϕ_n when $\tau_{n-1} = f$ and $\tau_n = g$. α_{nkg} is calculated as follows:

$$\begin{aligned}
\alpha_{1kg} &= p(X_{10g}, z_1 = k, \tau_1 = g) \\
&= p(X_{10g} | z_1 = k, \tau_1 = g) p(z_1 = k) p(\tau_1 = g) \\
&= b_{1k0g} \pi_k \rho_g, \qquad (13) \\
\alpha_{nkg} &= p(X_{10\tau_1}, \ldots, X_{nfg}, z_n = k, \tau_n = g) \\
&= \sum_{f=-G}^{G} p(X_{nfg} | z_n = k, \tau_{n-1} = f, \tau_n = g) \\
&\quad \cdot \sum_{j=1}^{K} p(X_{10\tau_1}, \ldots, X_{(n-1)\tau_{n-2}f}, z_{n-1}{=}j, \tau_{n-1}{=}f) \\
&\quad \cdot p(z_n{=}k | z_{n-1}{=}j) p(\tau_n{=}g) \\
&= \sum_{f=-G}^{G} b_{nkfg} \sum_{j=1}^{K} \alpha_{(n-1)jf} a_{jk} \rho_g. \qquad (14)
\end{aligned}$$

In backward sampling, Eq. (12) is calculated in the n-th beat interval by using the value of α_{nkg}, and the states (z_n, τ_n) are sampled recursively. When the $(n{+}1)$-th sampled states are $(z_{n+1}, \tau_{n+1}) = (k, g)$, β_{njf} is calculated as follows:

$$\begin{aligned}
\beta_{njf} &\propto p(X_{(n+1)fg} | z_{n+1} = k, \tau_n = f, \tau_{n+1} = g) \\
&\quad \cdot p(z_{n+1} = k | z_n = j) p(\tau_{n+1} = g) \\
&\quad \cdot p(X_{10\tau_1}, \ldots, X_{n\tau_n f}, z_n = j, \tau_n = f) \\
&= b_{(n+1)kfg} a_{jk} \rho_g \alpha_{njf}. \qquad (15)
\end{aligned}$$

Specifically, the latent variables (z_N, τ_N) are sampled in accordance with the probability given by

$$\beta_{Njf} \propto \alpha_{Njf}. \quad (16)$$

3.3.2 Learning Model Parameters $\boldsymbol{A}$, $\boldsymbol{\pi}$, and $\boldsymbol{\rho}$

We explain how to learn the values of $\boldsymbol{\Theta}$. In a sequence of latent variables $\{z_n, \tau_n\}_{n=1}^N$ which are sampled in backward sampling, the number of transitions such that $z_n = j$ and $z_{n+1} = k$ is represented as s_{jk} and the number of onset deviations such that $\tau_n = g$ is represented as u_g. The value of v_k is 1 at $z_1 = k$, and else where is 0. The parameters a_{jk}, ρ_g and π_k are updated by sampling from the posterior distributions given by

$$\begin{aligned}
p(\boldsymbol{a}_j | \boldsymbol{\xi}_j + \boldsymbol{s}_j) &= \mathrm{Dirichlet}(\boldsymbol{a}_j | \boldsymbol{\xi}_j + \boldsymbol{s}_j), \qquad (17) \\
p(\boldsymbol{\rho} | \boldsymbol{\eta} + \boldsymbol{u}) &= \mathrm{Dirichlet}(\boldsymbol{\rho} | \boldsymbol{\eta} + \boldsymbol{u}), \qquad (18) \\
p(\boldsymbol{\pi} | \boldsymbol{\zeta} + \boldsymbol{v}) &= \mathrm{Dirichlet}(\boldsymbol{\pi} | \boldsymbol{\zeta} + \boldsymbol{v}), \qquad (19)
\end{aligned}$$

where $\boldsymbol{s}_j = [s_{j1}, \ldots, s_{jK}]^T$, $\boldsymbol{u} = [u_{-G}, \ldots, u_G]^T$, and $\boldsymbol{v} = [v_1, \ldots, v_K]^T$.

3.3.3 Learning Cauchy Parameters c and d

To estimate the parameters c and d, we use the MH algorithm. It is hard to analytically calculate the posterior distributions of c and d because a Cauchy distribution doesn't have conjugate prior distributions. When the values of c and d are respectively c_i and d_i, we define proposal distributions of c and d as follows:

$$\begin{aligned}
q_c(c | c_i) &= \mathrm{Gamma}(c | \gamma c_i, \gamma), \qquad (20) \\
q_d(d | d_i) &= \mathrm{Gamma}(d | \delta d_i, \delta), \qquad (21)
\end{aligned}$$

where γ and δ are hyperparameters of the proposal distributions. Using the value of c^* sampled from $q_c(c|c_i)$, we calculate the acceptance ratio given by

$$g_c(c^*, c_i) = \min\left\{1, \frac{f_c(c^*) q_c(c_i | c^*)}{f_c(c_i) q_c(c^* | c_i)}\right\}, \quad (22)$$

where $f_c(c)$ is a likelihood function given by

$$\begin{aligned}
f_c(c) &\equiv p(c | x_{1:T}, z_{1:N}, \tau_{1:N}, \Theta, d_i) \\
&\propto \prod_{n=1}^{N} \rho_n b_{nz_n\tau_{n-1}\tau_n} \prod_{n=2}^{N} a_{z_{n-1}z_n} \pi_{z_1} q(c) \\
&= \prod_{n=1}^{N} \rho_n \left\{ \prod_{t=\phi_{n-1}}^{\phi_n - 1} \mathrm{Cauchy}(x_t | \mu_{z_n}, \lambda_t^c) \right\}^{\frac{1}{\phi_n - \phi_{n-1}}} \\
&\quad \cdot \prod_{n=2}^{N} a_{z_{n-1}z_n} \pi_{z_1} \mathrm{Gamma}(c | c_0, c_1), \qquad (23) \\
\lambda_t^c &= c_i \cdot \Delta x_t + d_i, \qquad (24)
\end{aligned}$$

Finally, if the value of $g_c(c^*, c_i)$ is larger than the random number r sampled from a uniform distribution on the interval $[0, 1]$, then $c_{i+1} = c^*$, and otherwise $c_{i+1} = c_i$, where c_0 is sampled from the prior distribution $q(c)$.

The value of d is updated in the same way as that of c. Using the value of d^* sampled from $q_d(d|d_i)$, we calculate the acceptance criteria given by

$$g_d(d^*, d_i) = \min\left\{1, \frac{f_d(d^*) q_d(d_i | d^*)}{f_d(d_i) q_d(d^* | d_i)}\right\}, \quad (25)$$

where $f_d(d)$ is a likelihood function given by

$$\begin{aligned} f_d(d) &\equiv p(d|x_{1:T}, z_{1:N}, \tau_{1:N}, \Theta, c_{i+1}) \\ &\propto \prod_{n=1}^{N} \rho_n b_{n z_n \tau_{n-1} \tau_n} \prod_{n=2}^{N} a_{z_{n-1} z_n} \pi_{z_1} q(c) \\ &= \prod_{n=1}^{N} \rho_n \left\{ \prod_{t=\phi_{n-1}}^{\phi_n - 1} \mathrm{Cauchy}(x_t | \mu_{z_n}, \lambda_t^d) \right\}^{\frac{1}{\phi_n - \phi_{n-1}}} \\ &\quad \cdot \prod_{n=2}^{N} a_{z_{n-1} z_n} \pi_{z_1} \mathrm{Gamma}(d | d_0, d_1), \end{aligned} \tag{26}$$

$$\lambda_t^d = c_{i+1} \cdot \Delta x_t + d_i, \tag{27}$$

Finally, if the value of $g_d(d^*, d_i)$ is larger than the random number r sampled from a uniform distribution on the interval $[0, 1]$, then $d_{i+1} = d^*$, and otherwise $d_{i+1} = d_i$, where d_0 is sampled from the prior distribution $q(d)$.

3.4 Viterbi Decoding

A latent sequence of musical notes is estimated by using the Viterbi algorithm that uses the parameters at the time when the likelihood given by

$$p(x_{1:T}) = \sum_{k=1}^{K} \sum_{g=-G}^{G} \alpha_{nkg} \tag{28}$$

is the maximum in the learning process. The musical notes that we want to estimate are the latent variables that maximize the value given by $p(\boldsymbol{Z}, \boldsymbol{\tau} | \boldsymbol{X})$. In the Viterbi algorithm, we define ω_{nkg} as follows:

$$\begin{aligned} &\omega_{nkg} = \\ &\max_{\substack{z_{1:n-1} \\ \tau_{1:n-1}}} \ln p(X_{10\tau_1}, \ldots, X_{n\tau_{n-1}g}, z_{1:n-1}, z_n{=}k, \tau_{1:n-1}, \tau_n{=}g), \end{aligned} \tag{29}$$

and ω_{nkg} is calculated recursively with the equations

$$\omega_{1kg} = \ln \rho_g + \ln b_{1k0g} + \ln \pi_k, \tag{30}$$

$$\omega_{nkg} = \ln \rho_g + \max_f \left[\ln b_{nkfg} + \max_j \left\{ \ln a_{jk} + \omega_{(n-1)jf} \right\} \right]. \tag{31}$$

In the recursive calculation of ω_{nkg}, when the states that maximize the value of ω_{nkg} are $z_{n-1} = j, \tau_{n-1} = f$, those states are memorized as $h_{nk}^{(z)} = j, h_{ng}^{(\tau)} = f$. After calculating $\{\omega_{Nkg}\}_{k=1,g=-G}^{K,G}$ with Eqs. (30) and (31), the sequence of latent variables $\{z_n, \tau_n\}_{n=1}^{N}$ is recursively estimated with the equations given by

$$(z_N, \tau_N) = \arg\max_{k,g} \{\omega_{nkg}\}, \tag{32}$$

$$z_n = h_{(n+1)z_{n+1}}^{(z)}, \tag{33}$$

$$\tau_n = h_{(n+1)\tau_{n+1}}^{(\tau)}. \tag{34}$$

The note sequence is retrieved by revising the onset deviations represented by the estimated latent variables $\{\tau_n\}_{n=1}^{N}$.

Model	Concordance rate
SBS-HMM	66.3 ± 1.0
Majority vote	56.9 ± 1.1
Frame-based HMM	56.1 ± 1.1
BS-HMM	67.0 ± 1.0

Table 1: Average concordance rates and their standard errors.

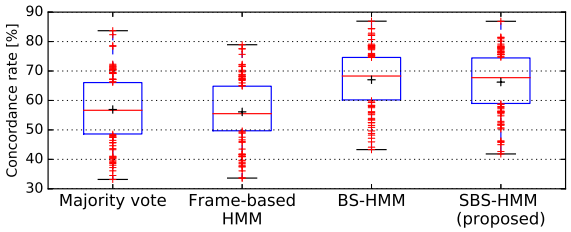


Figure 3: Concordance rates [%]. In the box plots, the red line indicates the median, the blue box indicates the range from the first to third quantile, the black cross indicates the mean, and the outliers are plotted with red crosses.

4. EVALUATION

We conducted an experiment to evaluate the proposed and previous methods in the accuracy of estimating musical notes from vocal F0 trajectories.

4.1 Experimental Conditions

The 100 pieces of popular music in RWC database [6] were used for the experiments. For each song, we trained model parameters, estimated the sequence of musical notes, and measured the accuracy of estimated musical notes. The input F0 trajectories were obtained from monaural music acoustic signals by the method of Ikemiya et al. [10]. We used the beat times obtained by a beat tracking system by Durand et al. [3]. This system estimates the beat times in units of a whole note, and the interval between adjacent beat times were divided into 16 equal intervals to obtain the beat times for 16th-note units were calculated.

For the proposed method, the sequence of musical notes was estimated with the Viterbi algorithm. The hyperparameters were $\boldsymbol{\xi} = \mathbb{1}, \boldsymbol{\zeta} = \mathbf{1}, \boldsymbol{\eta} = \mathbf{1}$, $c_0 = d_1 = d_0 = d_1 = 1$, where $\mathbb{1}$ and $\mathbf{1}$ respectively represent the matrix and vector whose elements are all ones. The parameters of the proposal distributions were $\gamma = \delta = 1$. The maximum value G that τ_n could take was $G = 5$ (i.e., 50 cents).

A majority-vote method was tested as a baseline. It estimates a musical note in each time unit corresponding to a 16th note by taking a majority vote for vocal F0s in the time unit. For comparison, a frame-based HMM and a beat-synchronous HMM (BS-HMM) were also tested. The frame-based HMM assumes that all beat intervals have only on frame. The BS-HMM is the same as SBS-HMM except that the onset deviation is not considered.

The estimated sequence of musical notes was compared with the ground-truth MIDI data synchronized to the vocal melody, and the concordance rate, i.e., the rate of frames in which pitches are correctly estimated, was used as the evaluation measure.

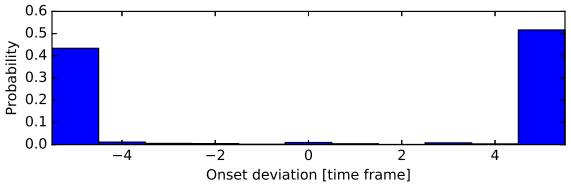


Figure 4: Example of a learned distribution of the model parameter ρ.

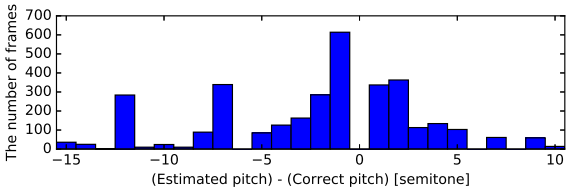


Figure 5: A example of pitch estimation error. The case that an estimated pitch is equal to a correct pitch is omitted.

4.2 Experimental Results

The results of note estimation are listed in Table 1 and Figure 3. The proposed model clearly outperformed the majority-vote method and the frame-based HMM in the average concordance rate. On the other hand, the average concordance rates for the proposed model and BS-HMM were similar and the difference was not statistically significant.

The results indicate that the model of music scores by transition probabilities and that of frequency deviations by output probabilities are both effective for improving the accuracy of musical note estimation. The cause of the result that the model of onset deviations did not improve the accuracy is probably that the model parameter ρ was not properly learned (Fig. 4). Capturing onset deviations by a single categorical distribution would be difficult, since onset depends on the duration of the pitches on either side of the onset, and on the over all tempo of the song. It would be necessary to model onset deviations in detail, for example by using a separate hidden state to represent the F0s during pitch transition.

4.2.1 Pitch Estimation Error

Two types of errors were mainly observed (Fig. 5). The first type was caused by singing styles of singers, and appears as errors that span one semitone and two semitones. This means that the frequency deviations affect the accuracy of note estimation. The second type was caused by the error of F0 estimation, and appears as the errors that span seven semitones and twelve semitones. The intervals of seven semitones and 12 semitones correspond to a perfect fifth and an octave, respectively.

4.2.2 Singing Style Extraction and Robustness

Example results of note estimation in Figure 6 show the potential of the proposed model to capture the singers' singing style. In the upper figure, the onset at the first beat is significantly delayed from the beat time. Whereas the proposed model correctly detected the delayed onset, the majority-vote method mis-identified the beat position of the pitch onset. The lower figure is an example of vibrato. With the majority-vote method, the estimation result was affected by the large frequency deviation. With the proposed method, on the other hand, the robustness due to the Cauchy distribution enabled the correct estimation of the pitch without being affected by the vibrato.

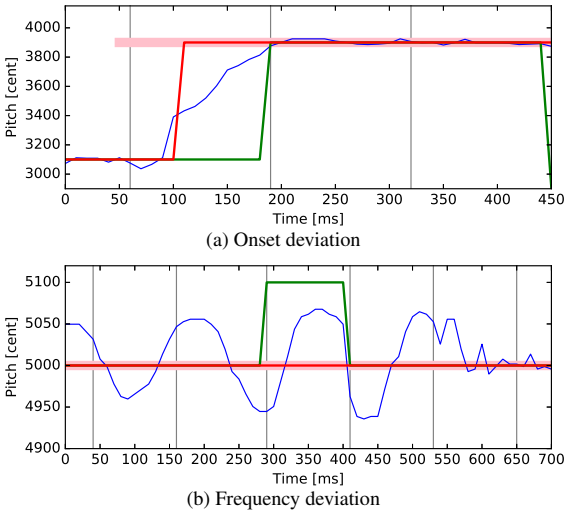


Figure 6: Examples of note estimation results. The pink, blue, green, red, and black vertical lines respectively indicate a MIDI note which is the ground-truth, the F0 trajectory of a singing voice including onset deviations, the pitches estimated by the majority-vote method, the pitches estimated by the proposed method, and the beat times estimated in advance.

5. CONCLUSION

This paper presented a method for estimating the musical notes of music from the vocal F0 trajectory. When modeling the process generating the vocal F0 trajectory, we considered not only the musical score component but also onset deviation and frequency deviation. The SBS-HMM estimated pitches more accurately than the majority-vote method and the frame-based method.

The onset deviation and frequency deviation that were obtained using the proposed method are important for grasping the characteristics of singing expression. Future work includes precise modeling of vocal F0 trajectories based on second-order filters and extraction of individual singing expression styles. In the proposed method, F0 estimation, beat tracking, and musical note estimation are conducted separately. It is necessary to integrate these methods. The proposed method cannot deal with the non-vocal regions in actual music, so we plan to also appropriately deal with non-vocal regions.

Acknowledgement: This study was partially supported by JST OngaCREST Project, JSPS KAKENHI 24220006, 26700020, 26280089, 16H01744, and 15K16054, and Kayamori Foundation.

6. REFERENCES

[1] A. de Cheveigné and H. Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[2] A. Dobashi, Y. Ikemiya, K. Itoyama, and K. Yoshii. A music performance assistance system based on vocal, harmonic, and percussive source separation and content visualization for music audio signals. *SMC*, 2015.

[3] S. Durand, J. P. Bello, B. David, and G. Richard. Downbeat tracking with multiple features and deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 409–413. IEEE, 2015.

[4] J.-L. Durrieu, G. Richard, B. David, and C. Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Transactions on Audio, Speech, and Language Processing,*, 18(3):564–575, 2010.

[5] M. Goto. PreFEst: A predominant-F0 estimation method for polyphonic musical audio signals. *Proceedings of the 2nd Music Information Retrieval Evaluation eXchange*, 2005.

[6] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical and jazz music databases. In *The International Society for Music Information Retrieval (ISMIR)*, volume 2, pages 287–288, 2002.

[7] M. Goto, K. Yoshii, H. Fujihara, M. Mauch, and T Nakano. Songle: A web service for active music listening improved by user contributions. In *The International Society for Music Information Retrieval (ISMIR)*, pages 311–316, 2011.

[8] D. J. Hermes. Measurement of pitch by subharmonic summation. *The journal of the acoustical society of America*, 83(1):257–264, 1988.

[9] P.-S. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 57–60. IEEE, 2012.

[10] Y. Ikemiya, K. Yoshii, and K. Itoyama. Singing voice analysis and editing based on mutually dependent F0 estimation and source separation. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 574–578. IEEE, 2015.

[11] Y. E. Kim and B. Whitman. Singer identification in popular music recordings using voice coding features. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, volume 13, page 17, 2002.

[12] A. Laaksonen. Automatic melody transcription based on chord transcription. In *The International Society for Music Information Retrieval (ISMIR)*, pages 119–124, 2014.

[13] Y. Li and D. Wang. Separation of singing voice from music accompaniment for monaural recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1475–1487, 2007.

[14] M. Mauch, C. Cannam, R. Bittner, G. Fazekas, J Salamon, J. Dai, J. Bello, and S Dixon. Computer-aided melody note transcription using the Tony software: Accuracy and efficiency. In *Proceedings of the First International Conference on Technologies for Music Notation and Representation - TENOR2015*, pages 23–30, Paris, France, 2015.

[15] M. Mauch and S. Dixon. pYIN: A fundamental frequency estimator using probabilistic threshold distributions. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 659–663. IEEE, 2014.

[16] Y. Ohishi, H. Kameoka, D. Mochihashi, and K. Kashino. A stochastic model of singing voice F0 contours for characterizing expressive dynamic components. In *INTERSPEECH*, pages 474–477, 2012.

[17] R. P. Paiva, T. Mendes, and A. Cardoso. On the detection of melody notes in polyphonic audio. In *The International Society for Music Information Retrieval (ISMIR)*, pages 175–182, 2005.

[18] G. E. Poliner and D. P. W. Ellis. A classification approach to melody transcription. *The International Society for Music Information Retrieval (ISMIR)*, pages 161–166, 2005.

[19] C. Raphael. A graphical model for recognizing sung melodies. In *The International Society for Music Information Retrieval (ISMIR)*, pages 658–663, 2005.

[20] M. Ryynänen, T. Virtanen, J. Paulus, and A. Klapuri. Accompaniment separation and karaoke application based on automatic melody transcription. In *2008 IEEE International Conference on Multimedia and Expo*, pages 1417–1420. IEEE, 2008.

[21] M. P. Ryynänen and A. P. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, 2008.

[22] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[23] W.-H. Tsai and H.-M. Wang. Automatic singer recognition of popular music recordings via estimation and modeling of solo vocal signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):330–341, 2006.

ON THE EVALUATION OF RHYTHMIC AND MELODIC DESCRIPTORS FOR MUSIC SIMILARITY

Maria Panteli, Simon Dixon
Centre for Digital Music, Queen Mary University of London, United Kingdom
{m.panteli, s.e.dixon}@qmul.ac.uk

ABSTRACT

In exploratory studies of large music collections where often no ground truth is available, it is essential to evaluate the suitability of the underlying methods prior to drawing any conclusions. In this study we focus on the evaluation of audio features that can be used for rhythmic and melodic content description and similarity estimation. We select a set of state-of-the-art rhythmic and melodic descriptors and assess their invariance with respect to transformations of timbre, recording quality, tempo and pitch. We create a dataset of synthesised audio and investigate which features are invariant to the aforementioned transformations and whether invariance is affected by characteristics of the music style and the monophonic or polyphonic character of the audio recording. From the descriptors tested, the scale transform performed best for rhythm classification and retrieval and pitch bihistogram performed best for melody. The proposed evaluation strategy can inform decisions in the feature design process leading to significant improvement in the reliability of the features.

1. INTRODUCTION

With the significant number of music information retrieval techniques and large audio collections now available it is possible to explore general trends in musical style evolution [11, 16]. Such exploratory studies often have no ground truth to compare to and therefore any conclusions are subject to the validity of the underlying tools. In music content-based systems this often translates to the ability of the audio descriptors to correctly and sufficiently represent the music-specific characteristics.

In this study we focus on the evaluation of audio features that can be used for rhythmic and melodic content description and similarity estimation. We are particularly interested in audio features that can be used to describe world music recordings. We propose an evaluation framework which aims to simulate the challenges faced in the analysis of recorded world music collections, such as processing noisy recordings or audio samples exhibiting a variety of music style characteristics. In particular, we define transformations with respect to timbre, recording quality, tempo and key and assess the invariance of a set of state-of-the-art rhythmic and melodic descriptors.

A number of studies have dealt with the evaluation of audio features and specifically of rhythmic and melodic descriptors. Robustness is usually addressed in the design process where certain decisions ensure tempo or key invariance of the features. For example, rhythmic descriptors have been designed to achieve partial [5,6] or complete [7] tempo invariance, and melodic descriptors exist which are tempo and/or key invariant [1,22,24]. Robustness to audio quality has been also addressed for MFCC and chroma features (describing timbre and harmony respectively) [21]. The relevance of the features is not guaranteed even if a classification task seems successful. For example, unbalanced datasets can lead to high accuracies in genre classification tasks [18], or high rhythm classification accuracies can be achieved with (only) tempo information [4, 6] indicating that other audio features used had limited relevant contribution to the task.

To be perceptually valid, and useful in real-world collections, the representations need to be invariant to subtle changes in tempo, key (or reference pitch), recording quality and timbre. Additionally, to be usable in cross-cultural studies, the features need to be agnostic to properties of particular music cultures. For instance, pitch representations should not depend on the 12-tone equal temperament tuning, and rhythm representations should not depend on specific Western metric structures such as $\frac{4}{4}$ metre.

We examine a selection of audio features for rhythm and melody to assess their suitability for scientific studies of world music corpora subject to the above constraints. To achieve this we test classification and retrieval performance of multiple rhythm and melody features on a controlled dataset, which allows us to systematically vary timbre, tempo, pitch and audio quality. The main contributions of the paper are the controlled dataset, which we make freely available, and the proposed evaluation strategy to assess robustness and facilitate the feature design and selection process.

2. FEATURES

We present details of three descriptors from each category (rhythm and melody), chosen from the literature based on their performance on related classification and retrieval tasks. In the paragraphs below we provide a short sum-

© Maria Panteli, Simon Dixon. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Maria Panteli, Simon Dixon. "On the evaluation of rhythmic and melodic descriptors for music similarity", 17th International Society for Music Information Retrieval Conference, 2016.

mary of these features and discuss further considerations of their design. Our implementations of the features follow the specifications published in the corresponding research papers but are not necessarily exact replicas.

2.1 Rhythm

We start our investigation with state-of-the-art rhythmic descriptors that have been used in similarity tasks including genre and rhythm classification [5, 7, 12]. The rhythmic descriptors we use here share the general processing pipeline of two consecutive frequency analyses [15]. First, a spectrogram representation is calculated, usually with frequencies on the mel scale. The fluctuations in its "rows", i.e. the frequency bands, are then analysed for their rhythmic frequency content over larger windows. This basic process has multiple variations, which we explain below.

For comparison purposes we fix the sampling rate at 44100 Hz for all features. Likewise, the spectrogram frame size is 40 ms with a hop size of 5 ms. All frequency bins are mapped to the mel scale. The rhythmic periodicities are calculated on 8-second windows with a hop size of 0.5 seconds. In the second step we compute the periodicities within each mel band then average across all bands, and finally summarise a recording by taking the mean across all frames.

Onset Patterns (OP). The defining characteristic of Onset Patterns is that the mel frequency magnitude spectrogram is post-processed by computing the first-order difference in each frequency band and then subtracting the mean and half-wave rectifying the result. The resulting onset function is then frequency-analysed using the discrete Fourier transform [5, 6, 14]. We omit the post-processing step of transforming the resulting linear fluctuation frequencies to $\log_2$-spaced frequencies. The second frame decomposition results in an $F \times P_O$ matrix with $F = 40$ mel bands and $P_O = 200$ periodicities linearly spaced up to 20 Hz.

Fluctuation Patterns (FP). Fluctuation patterns differ from onset patterns by using a log-magnitude mel spectrogram, and by the additional application of psychoacoustic models (e.g. loudness and fluctuation resonance models) to weight perceptually relevant periodicities [12]. We use the MIRToolbox [8] implementation of fluctuation patterns with the parameters specified at the beginning of Section 2.1. Here, we obtain an $F \times P_F$ matrix with $F = 40$ mel bands and $P_F = 1025$ periodicities of up to 10 Hz.

Scale Transform (ST). The scale transform [7], is a special case of the Mellin transform, a scale-invariant transformation of the signal. Here, the scale invariance property is exploited to provide tempo invariance. When first introduced, the scale transform was applied to the autocorrelation of onset strength envelopes spanning the mel scale [7]. Onset strength envelopes here differ from the onset function implemented in OP by the steps of post-processing the spectrogram. In our implementation we apply the scale transform to the onset patterns defined above.

2.2 Melody

Melodic descriptors selected for this study are based on intervals of adjacent pitches or 2-dimensional periodicities of the chromagram. We use a chromagram representation derived from an NMF-based approximate transcription.

For comparison purposes we fix the following parameters in the design of the features: sampling rate at 44100 Hz, variable-Q transform with 3 ms hop size and pitch resolution at 60 bins per octave (to account for microtonality), secondary frame decomposition (where appropriate) using an 8-second window and 0.5-second hop size, and finally averaging the outcome across all frames in time.

Pitch Bihistogram (PB). The pitch bihistogram [22] describes how often pairs of pitch classes occur within a window d of time. It can be represented as an n-by-n matrix P where n is the number of pitch classes and element p_{ij} denotes the count of co-occurrences of pitch classes i and j. In our implementation, the pitch content is wrapped to a single octave to form a chromagram with 60 discrete bins and the window length is set to $d = 0.5$ seconds. The feature values are normalised to the range $[0, 1]$. To approximate key invariance the bihistogram is circularly shifted to $p_{i-\hat{i},j-\hat{i}}$ where $p_{\hat{i}\hat{j}}$ denotes the bin of maximum magnitude. This does not strictly represent tonal structure but rather relative prominence of the pitch bigrams.

2D Fourier Transform Magnitudes (FTM). The magnitudes of the 2-dimensional Fourier transform of the chromagram describe periodicities in both frequency and time axes. This feature renders the chromagram key-invariant, but still carries pitch content information, and has accordingly been used in cover song recognition [1,9]. In our implementation, chromagrams are computed with 60 bins per octave and no beat-synchronisation. The FTM is applied with the frame decomposition parameters stated above. We select only the first 50 frequency bins which correspond to periodicities up to 16 Hz.

Intervalgram (IG). The intervalgram [24] is a representation of chroma vectors averaged over different windows in time and cross-correlated with a local reference chroma vector. In the implementation we use,we reduce this to one window size $d = 0.5$, and cross-correlation is computed on every pair of chroma vectors from successive windows.

In this study we place the emphasis on the evaluation framework and provide a baseline performance of (only) a small set of features. The study could be extended to include more audio descriptors and performance accuracies could be compared in order to choose the best descriptor for a given application.

3. DATA

For our experiments we compiled a dataset of synthesised audio, which allowed us to control transformations under which ideal rhythmic and melodic descriptors should be invariant. In the sections below we present the dataset of selected rhythms and melodies and detailed description of their transformations.

Melody		Rhythm	
Description	No.	Description	No.
Dutch Folk (M)	5	Afro-American (M)	5
Classical (M)	5	North-Indian (M)	5
Byzantine (M)	5	African (M)	5
Pop (M)	5	Classical (M)	5
Classical (P)	5	EDM (P)	5
Pop (P)	5	Latin-Brazilian (P)	5

Table 1: The dataset of rhythms and melodies transformed for feature robustness evaluation. (M) is monophonic and (P) polyphonic as described in Section 3.1.

3.1 Material

We compiled 30 melodies and 30 rhythms extracted from a variety of musical styles with both monophonic and polyphonic structure (Table 1). In particular, we collect MIDI monophonic melodies of classical music used in the MIREX 2013: Discovery of Repeated Themes and Sections task [1], MIDI monophonic melodies of Dutch folk music from the Meertens Tune Collections [23], fundamental frequency (F0) estimates of monophonic pop melodies from the MedleyDB dataset [2], fundamental frequency (F0) estimates of monophonic Byzantine religious music [13], MIDI polyphonic melodies of classical music from the MIREX 2013 dataset, and fundamental frequency (F0) estimates of polyphonic pop music from the MedleyDB dataset. These styles exhibit differences in the melodic pitch range, for example, classical pieces span multiple octaves whereas Dutch folk and Byzantine melodies are usually limited to a single octave range. Pitch from fundamental frequency estimates allows us to also take into account vibrato and microtonal intervals. This is essential for microtonal tuning systems such as Byzantine religious music, and for melodies with ornamentation such as recordings of the singing voice in the Dutch folk and pop music collections.

For rhythm we collect rhythmic sequences common in Western classical music traditions [17], African music traditions [20], North-Indian and Afro-American traditions [19], Electronic Dance Music (EDM) [3], and Latin-Brazilian traditions. [2] These rhythms span different metres such as $\frac{11}{8}$ in North-Indian, $\frac{12}{8}$ in African, $\frac{4}{4}$ in EDM, and $\frac{6}{8}$ in Latin-Brazilian styles. The rhythms for Western, African, North-Indian, Afro-American traditions are constructed from single rhythmic patterns whereas EDM and Latin-Brazilian rhythms are constructed with multiple patterns overlapping in time. We refer to the use of a single pattern as 'monophonic' and of multiple patterns as 'polyphonic' for consistency with the melodic dataset.

[1] http://www.tomcollinsresearch.net/mirex-pattern-discovery-task.html

[2] http://www.formedia.ca/rhythms/5drumset.html

3.2 Transformations

Intuitively, melodies and rhythms retain their character even if the music is transposed to a different tonality, played at a (slightly) different tempo or under different recording conditions. These are variations that we expect to find in real-world corpora, and to which audio features should be reasonably invariant. Indeed, the cover song identification literature suggests that invariance of features in terms of key transpositions and tempo shifts is desirable [1, 22, 24]; for rhythm description, the existing literature mainly focuses on tempo invariance and robustness against recording quality [5, 6]. We add to this list the requirement of invariance to slight changes in timbre for both melody and rhythm description [3] . Overall, we test our features for robustness in tempo, pitch, timbre and recording quality by systematically varying these parameters to produce multiple versions of each melody and rhythm (Table 2). We apply only one transformation at a time while keeping the other factors constant. The 'default' version of a rhythm or melody is computed using one of the 25 timbres available, fixing the tempo at 120 bpm, and, for melody, keeping the original key as expressed in the MIDI or F0 values. The dataset is made available online [4] .

Timbre (Timb): For a given sequence of MIDI notes or fundamental frequency estimates we synthesise audio using sine waves with time-varying parameters. The synthesised timbres vary from harmonic to inharmonic sounds and from low to high frequency range. For a given set of rhythm sequences we synthesise audio using samples of different (mainly percussive) instruments [5] . Beyond the typical drum set sounds (kick, snare, hi-hat), we include percussive instruments of different music traditions such as the Indian mridangam, the Arabic daf, the Turkish darbuka, and the Brazilian pandeiro. Overall, we use 25 different timbres for each melody and rhythm in the dataset.

Recording Quality (RecQ): Large music archives usually contain material recorded under a variety of recording conditions, and are preserved to different degrees of fidelity. We use the Audio Degradation Toolbox [10] to create 25 audio degradations that we expect to be representative of what is found in such archives. Amongst the degradations we consider are effects of prominent reverb (live recordings), overlaid random noise (old equipment), added random sounds including speech, birds, cars (field recording), strong compression (MP3), wow sampling, high or low pass filtering (vinyl or low quality microphone).

Global tempo shifts (GTemp): We define 'small' variations the tempo changes of up to 20% of the original tempo (in this case centred at 120 bpm), which we assume will leave the character of melodies and rhythms intact. In particular, we use 25 tempo shifts distributed in the range $[-20, 20]$ (excluding 0) percent slower or faster than the original speed.

[3] The timbre transformations we consider are not expected to vastly alter the perception of a rhythm or melody.

[4] https://code.soundsoftware.ac.uk/projects/rhythm-melody-feature-evaluation

[5] http://www.freesound.org

Transformations	Values
Timbre	25 distinct timbres (similar frequency range and instrument)
Rec. Quality	25 degradations including reverb, compression, wow, speech, noise
Global Tempo	25 values in $[-20, 20]$ percent deviation from original tempo
Key Transp.	25 values in $[-10, 10]$ semitones deviation from original key
Local Tempo	25 values in $[-20, 20]$ percent deviation from original tempo

Table 2: Transformations for assessing feature invariance.

Key transpositions/Local tempo shifts (KeyT/LTemp): For melodic descriptor robustness we consider transposing the audio with respect to 25 key transpositions in the range $[-10, 10]$ (excluding 0) semitones from the original key. These shifts include microtonal intervals e.g. a transposition of 1.5 semitones up as one expects to find in world music singing examples. For rhythmic descriptor robustness we consider instead small step changes of the tempo. We introduce a local tempo change for a duration of 2 (out of 8) seconds centred around the middle of the recording. This is common in, for example, performances of amateur musicians where they might unintentionally speed up or slow down the music. Similar to global tempo transformation we use 25 shifts in the range $[-20, 20]$ percent slower or faster than the original speed.

While the above transformations do not define an exhaustive list of effects and variations found in world music corpora they provide a starting point for assessing feature robustness. The dataset can be expanded in future work to include more transformations and parameter values. For this study we restrict to the abovementioned 4 transformations with 25 values each (Table 2). For our dataset of 30 rhythms and 30 melodies this results in a total of 3000 transformed rhythms and 3000 transformed melodies.

4. EVALUATION STRATEGY

With the proposed evaluation strategy we would like to assess feature robustness with respect to the transformations and transformation values presented above in Section 3.2. Additionally we would like to check whether the performance of the features relates to particularities of the music style for the styles presented in Section 3.1. Lastly, since our dataset consists of monophonic and polyphonic melodies and rhythms, we would also like to check whether the features are influenced by the monophonic or polyphonic character of the audio signal.

Robustness evaluation is performed on the dataset of 3000 transformed rhythms and 3000 transformed melodies (Section 3.1). Considering the variety of MIR tasks and corresponding MIR models, we choose to assess feature performance accuracy in both classification and retrieval experiments as explained below. In our experiments we include a variety of classifiers and distance metrics to cover a wide range of audio feature similarity methods.

We first verify the power of the features to classify different melodies and rhythms. To do so we employ four classifiers: K-Nearest Neighbors (KNN) with 1 neighbor and Euclidean distance metric, Support Vector Machine (SVM) with a linear kernel, Linear Discriminant Analysis (LDA) with 20 components, and Gaussian Naive Bayes. We use 5-fold cross-validation for all classification experiments. In each case the prediction target is one of the 30 rhythm or melody 'families'. For each of the 3000 transformed rhythms or melodies we output the classification accuracy as a binary value, 1 if the rhythm or melody was classified correctly and 0 otherwise.

As reassuring as good classification performance is, it does not imply that a melody or rhythm and its transformations cluster closely in the original feature space. Accordingly, we choose to use a similarity-based retrieval paradigm that more directly reflects the feature representations. For each of the 30 rhythms or melodies we choose one of the 25 timbres as the default version of the rhythm or melody, which we use as the query. We rank the 2999 candidates based on their distance to the query and assess the recall rate of its 99 transformations. Each transformed rhythm or melody is assigned a score of 1 if it was retrieved in the top $K = 99$ results of its corresponding query and 0 otherwise. We compare four distance metrics, namely Euclidean, cosine, correlation and Mahalanobis.

For an overview of the performance of the features we compute the mean accuracy across all recordings for each classification or retrieval experiment and each feature. To better understand why a descriptor is successfull or not in the corresponding classification or retrieval task we further analyse the performance accuracies with respect to the different transformations, transformation values, music style and monophonic versus polyphonic character. To achieve this we group recordings by, for example, transformation, and compute the mean accuracy for each transformation. We discuss results in the section below.

5. RESULTS

The mean performance accuracy of each feature and each classification or retrieval experiment is shown in Table 3. Overall, the features with the highest mean classification and retrieval accuracies are the scale transform (ST) for rhythm and the pitch bihistogram (PB) for melody.

5.1 Transformation

We consider four transformations for rhythm and four for melody. We compute the mean accuracy per transformation by averaging accuracies of recordings from the same transformation. Results for rhythm are shown in Table 4 and for melody in Table 5. Due to space limitations we present results for only the best, on average, classifier (KNN) and similarity metric (Mahalanobis) as obtained in Table 3. We observe that onset patterns and fluctuation patterns show, on average, lower accuracies for transformations based on global tempo deviations. This is expected as the aforementioned descriptors are not tempo invariant.

	Rhythm			Melody		
Metric	ST	OP	FP	PB	IG	FTM
Classification						
KNN	**0.86**	0.71	0.68	**0.88**	0.83	0.86
LDA	**0.82**	0.66	0.59	**0.83**	0.82	0.82
NB	**0.80**	0.62	0.58	**0.84**	0.76	0.81
SVM	**0.87**	0.66	0.59	0.86	0.86	**0.87**
Retrieval						
Euclidean	**0.65**	0.47	0.42	**0.80**	0.56	0.67
Cosine	**0.66**	0.47	0.42	**0.80**	0.55	0.68
Correlation	**0.66**	0.47	0.42	**0.80**	0.54	0.67
Mahalanobis	**0.61**	0.48	0.40	**0.81**	0.60	0.72

Table 3: Mean accuracy of the rhythmic and melodic descriptors for the classification and retrieval experiments.

Metric	Feature	Timb	GTemp	RecQ	LTemp
Classification					
KNN	ST	**0.98**	**0.90**	**0.93**	0.62
KNN	OP	0.97	0.20	0.92	**0.75**
KNN	FP	0.91	0.18	0.92	0.71
Retrieval					
Mahalan.	ST	**0.95**	**0.36**	**0.91**	**0.25**
Mahalan.	OP	0.94	0.00	0.88	0.13
Mahalan.	FP	0.62	0.01	0.87	0.09

Table 4: Mean accuracies of the rhythmic descriptors under four transformations (Section 3.1).

In the rhythm classification task, the performance of the scale transform is highest for global tempo deviations but it is lowest for local tempo deviations. We believe this is due to the scale transform assumption of a constant periodicity over the 8-second frame, an assumption that is violated when local tempo deviations are introduced. We also note that fluctuation patterns show lower performance accuracies for transformations of the timbre compared to the onset patterns and scale transform descriptors.

5.2 Transformation Value

We also investigate whether specific transformation values affect the performance of the rhythmic and melodic descriptors. To analyse this we compute mean classification accuracies averaged across recordings of the same transformation value (there are 25 values for each of 4 transformations so 100 mean accuracies in total). Due to space limitations we omit the table of results and report only a summary of our observations.

Onset patterns and fluctuation patterns exhibit low classification accuracies for almost all global tempo deviations whereas scale transform only shows a slight performance degradation on global tempo deviations of around $\pm 20\%$. For local tempo deviations, scale transform performs poorly at large local deviations (magnitude $> 15\%$) whereas onset patterns and fluctuation patterns show higher accuracies for these particular parameters. All descriptors seem to be robust to degradations of the recording quality with the exception of a wow effect that causes all rhythmic descriptors to perform poorly. Onset patterns and fluctuation patterns perform poorly also in the degradation of a radio-broadcast compression.

Metric	Feature	Timb	GTemp	RecQ	KeyT
Classification					
KNN	PB	0.97	**0.99**	**0.78**	0.76
KNN	IG	0.95	**0.99**	0.62	0.77
KNN	FTM	**0.98**	0.96	0.71	**0.79**
Retrieval					
Mahalan.	PB	**0.94**	**0.98**	**0.78**	0.53
Mahalan.	IG	0.70	0.91	0.33	0.46
Mahalan.	FTM	0.87	0.88	0.57	**0.57**

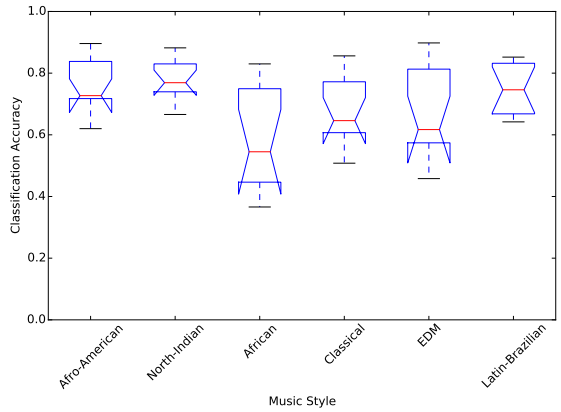
Table 5: Mean accuracies of the melodic descriptors under four transformations (Section 3.1).

For melody classification, all features perform poorly on key transpositions of more than 6 semitones up amd a wow effect degradation. Pitch bihistogram also performs poorly in transpositions between $2.5 - 5$ semitones down. Intervalgram and Fourier transform magnitudes perform badly also for reverb effect degradations and noisy recordings with overlaid wind, applause, or speech sound effects.

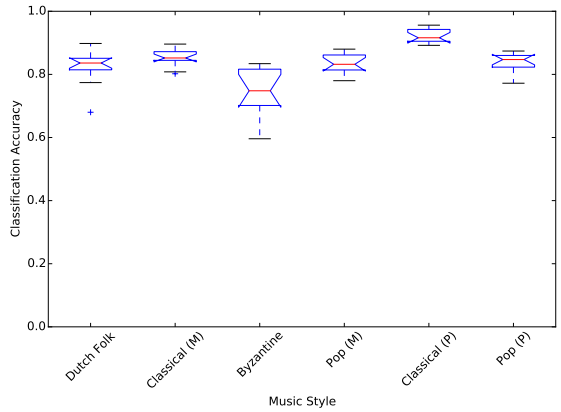
5.3 Music Style

Our dataset consists of rhythms and melodies from different music styles and we would like to test whether the robustness of the features is affected by the style. To achieve this we average classification accuracies across recordings of the same style. We have 6 styles for rhythm with 500 recordings in each style and likewise for melody. This gives us 6 mean accuracies for each feature and each classification experiment. We summarise results in a boxplot as shown in Figure 1. We also perform two sets of multiple paired t-tests with Bonferroni correction, one for rhythmic and one for melodic descriptors, to test whether mean classification accuracies per style are significantly different.

Using the paired t-tests with multiple comparison correction we observe that the majority of pairs of styles are significantly different at the Bonferroni significance level $alpha = 0.003$ for both the rhythmic and melodic descriptors. In particular the accuracies for classification and retrieval of African rhythms are significantly different from all other styles. Western classical rhythms are significantly different from all other styles except the EDM rhythms, and North-Indian rhythms are significantly different from all other styles except the EDM and Latin-Brazilian rhythms. For melody, the accuracies for the Byzantine and polyphonic pop styles are significantly different from all other styles. The descriptors that perform particularly badly with respect to these styles are the fluctuation patterns for rhythm and the intervalgram for melody. We use our current results as an indication of which styles might possibly affect the performance of the features but leave the analysis of the intra-style similarity for future work.



(a) Rhythm



(b) Melody

Figure 1: Box plot of classification accuracies of the rhythmic (top) and melodic (bottom) descriptors for each style.

5.4 Monophonic versus Polyphonic

Our dataset consists of monophonic and polyphonic melodies and rhythms and we would like to test whether the performance of the features is affected by the monophonic or polyphonic character. Similar to the preceding analysis, we average performance accuracies across all monophonic recordings and across all polyphonic recordings. We perform two paired t-tests, one for rhythmic and one for melodic descriptors, to test whether mean classification accuracies of monophonic recordings are drawn from a distribution with the same mean as the polyphonic recordings distribution. At the $\alpha = 0.05$ significance level the null hypothesis is not rejected for rhythm, $p = 0.25$, but is rejected for melody, $p < 0.001$. The melodic descriptors achieve on average higher classification accuracies for polyphonic ($M = 0.88$, $SD = 0.02$) than monophonic recordings ($M = 0.82$, $SD = 0.04$).

6. DISCUSSION

We have analysed the performance accuracy of the features under different transformations, transformation values, music styles, and monophonic versus polyphonic structure. Scale transform achieved the highest accuracy for rhythm classification and retrieval, and pitch bihistogram for melody. The scale transform is less invariant to transformations of the local tempo, and the pitch bihistogram to transformations of the key. We observed that the descriptors are not invariant to music style characteristics and that the performance of melodic descriptors depends on the pitch content being monophonic or polyphonic.

We have performed this evaluation on a dataset of synthesised audio. While this is ideal for adjusting degradation parameters and performing controlled experiments like the ones presented in this study, it may not be representative of the analysis of real-world music recordings. The latter involve many challenges, one of which is the mix of different instruments which results in a more complex audio signal. In this case rhythmic or melodic elements may get lost in the polyphonic mixture and further preprocessing of the spectrum is needed to be able to detect and isolate the relevant information.

Our results are based on the analysis of success rates on classification or retrieval tasks. This enabled us to have an overview of the performances of different audio features across several factors: transformation, transformation value, style, monophonic or polyphonic structure. A more detailed analysis could involve a fixed effects model where the contribution of each factor to the performance accuracy of each feature is tested individually.

In this evaluation we used a wide range of standard classifiers and distance metrics with default settings. We have not tried to optimise parameters nor use more advanced models since we wanted the evaluation to be as independent of the application as possible. However, depending on the application different models could be trained to be more robust to certain transformations than others and higher performance accuracies could be achieved.

7. CONCLUSION

We have investigated the invariance of audio features for rhythmic and melodic content description of diverse music styles. A dataset of synthesised audio was designed to test invariance against a broad range of transformations in timbre, recording quality, tempo and pitch. Considering the criteria and analyses in this study the most robust rhythmic descriptor is the scale transform and melodic descriptor the pitch bihistogram. Results indicated that the descriptors are not completely invariant to characteristics of the music style and lower accuracies were particularly obtained for African and EDM rhythms and Byzantine melodies. The performance of the melodic features was slightly better for polyphonic than monophonic content. The proposed evaluation framework can inform decisions in the feature design process leading to significant improvement in the reliability of the features.

8. ACKNOWLEDGEMENTS

MP is supported by a Queen Mary Principal's research studentship and the EPSRC-funded Platform Grant: Digital Music (EP/K009559/1).

9. REFERENCES

[1] T. Bertin-Mahieux and D. P. W. Ellis. Large-scale cover song recognition using the 2D Fourier transform magnitude. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 241–246, 2012.

[2] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello. MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 155–160, 2014.

[3] M. J. Butler. *Unlocking the Groove*. Indiana University Press, Bloomington and Indianapolis, 2006.

[4] S. Dixon, F. Gouyon, and G. Widmer. Towards Characterisation of Music via Rhythmic Patterns. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 509–516, 2004.

[5] T. M. Esparza, J. P. Bello, and E. J. Humphrey. From Genre Classification to Rhythm Similarity: Computational and Musicological Insights. *Journal of New Music Research*, 44(1):39–57, 2014.

[6] A. Holzapfel, A. Flexer, and G. Widmer. Improving tempo-sensitive and tempo-robust descriptors for rhythmic similarity. In *Proceedings of the 8th Sound and Music Computing Conference*, pages 247–252, 2011.

[7] A. Holzapfel and Y. Stylianou. Scale Transform in Rhythmic Similarity of Music. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):176–185, 2011.

[8] O. Lartillot and P. Toiviainen. A Matlab Toolbox for Musical Feature Extraction From Audio. In *International Conference on Digital Audio Effects*, pages 237–244, 2007.

[9] M. Marolt. A mid-level representation for melody-based retrieval in audio collections. *IEEE Transactions on Multimedia*, 10(8):1617–1625, 2008.

[10] M. Mauch and S. Ewert. The Audio Degradation Toolbox and Its Application to Robustness Evaluation. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 83–88, 2013.

[11] M. Mauch, R. M. MacCallum, M. Levy, and A. M. Leroi. The evolution of popular music: USA 19602010. *Royal Society Open Science*, 2(5):150081, 2015.

[12] E. Pampalk, A. Flexer, and G. Widmer. Improvements of Audio-Based Music Similarity and Genre Classification. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 634–637, 2005.

[13] M. Panteli and H. Purwins. A Quantitative Comparison of Chrysanthine Theory and Performance Practice of Scale Tuning, Steps, and Prominence of the Octoechos in Byzantine Chant. *Journal of New Music Research*, 42(3):205–221, 2013.

[14] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 525–530, 2009.

[15] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *The Journal of the Acoustical Society of America*, 103(1):588–601, 1998.

[16] J. Serrà, Á. Corral, M. Boguñá, M. Haro, and J. L. Arcos. Measuring the Evolution of Contemporary Western Popular Music. *Scientific Reports*, 2, 2012.

[17] S. Stober, D. J. Cameron, and J. A. Grahn. Classifying EEG recordings of rhythm perception. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 649–654, 2014.

[18] B. L. Sturm. Classification accuracy is not enough. *Journal of Intelligent Information Systems*, 41(3):371–406, 2013.

[19] E. Thul and G. T. Toussaint. A Comparative Phylogenetic-Tree Analysis of African Timelines and North Indian Talas. In *Bridges Leeuwarden: Mathematics, Music, Art, Architecture, Culture*, pages 187–194, 2008.

[20] G. Toussaint. Classification and phylogenetic analysis of African ternary rhythm timelines. In *Meeting Alhambra, ISAMA-BRIDGES Conference*, pages 25–36, 2003.

[21] J. Urbano, D. Bogdanov, P. Herrera, E. Gómez, and X. Serra. What is the effect of audio quality on the robustness of MFCCs and chroma features. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 573–578, 2014.

[22] J. van Balen, D. Bountouridis, F. Wiering, and R. Veltkamp. Cognition-inspired Descriptors for Scalable Cover Song Retrieval. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 379–384, 2014.

[23] P. Van Kranenburg, M. de Bruin, L. P. Grijp, and F. Wiering. *The Meertens Tune Collections*. Meertens Institute, Amsterdam, no. 1 edition, 2014.

[24] T. C. Walters, D. A. Ross, and R. F. Lyon. The Intervalgram: An Audio Feature for Large-scale Melody Recognition. In *9th International Conference on Computer Music Modeling and Retrieval*, pages 19–22, 2012.

ON THE POTENTIAL OF SIMPLE FRAMEWISE APPROACHES TO PIANO TRANSCRIPTION

Rainer Kelz, Matthias Dorfer, Filip Korzeniowski,
Sebastian Böck, Andreas Arzt, Gerhard Widmer
Department of Computational Perception, Johannes Kepler University Linz, Austria
rainer.kelz@jku.at

ABSTRACT

In an attempt at exploring the limitations of simple approaches to the task of piano transcription (as usually defined in MIR), we conduct an in-depth analysis of neural network-based framewise transcription. We systematically compare different popular input representations for transcription systems to determine the ones most suitable for use with neural networks. Exploiting recent advances in training techniques and new regularizers, and taking into account hyper-parameter tuning, we show that it is possible, by simple bottom-up frame-wise processing, to obtain a piano transcriber that outperforms the current published state of the art on the publicly available MAPS dataset – without any complex post-processing steps. Thus, we propose this simple approach as a new baseline for this dataset, for future transcription research to build on and improve.

1. INTRODUCTION

Since their tremendous success in computer vision in recent years, neural networks have been used for a large variety of tasks in the audio, speech and music domain. They often achieve higher performance than hand-crafted feature extraction and classification pipelines [20]. Unfortunately, using this model class brings along considerable computational baggage in the form of hyper-parameter tuning. These hyper-parameters include architectural choices such as the number and width of layers and their type (e.g. dense, convolutional, recurrent), learning rate schedule, other parameters of the optimization scheme and regularizing mechanisms. Whereas for computer vision these successes were possible using raw pixels as the input representation, in the audio domain there seems to be an additional complication. Here the choices for how to best represent the input range from spectrograms, logarithmically filtered spectrograms over constant-Q transforms to even the raw audio itself [10].

© Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, Gerhard Widmer. "On the Potential of Simple Framewise Approaches to Piano Transcription", 17th International Society for Music Information Retrieval Conference, 2016.

This is a tedious problem, and there seem to be only two solutions to it: manual hyper-parameter selection, where a human expert tries to make decisions based on her past experience, or automatic hyper-parameter optimization as discussed in [4, 11, 28]. In this work we pursue a mixed strategy. As a first step, we systematically find the most suitable input representation, and progress from there with human expert knowledge to find best performing architectures for the task of framewise piano transcription.

A variety of neural network architectures has been used specifically for framewise transcription of piano notes from monaural sources. Some transcription systems are separated into an *acoustic model* and a *musical language model*, such as [7, 26, 27], whereas in others there is no such distinction [2, 6, 23]. As shown in [26], models that utilize *musical language models* perform better than those without, albeit the differences seem to be small. We focus on the *acoustic model* here, neglecting the complementary language model for now.

2. INPUT, METHODS AND PARAMETERS

In what follows, we will describe the input representations we compared, and give a brief overview of techniques for training and regularizing neural networks.

2.1 Input Representation

Time-frequency representations in the form of spectrograms still seem to have a distinct advantage over the *raw* audio input, as mentioned in [10]. The exact parameterization of spectrograms is not entirely clear however, so we try to address this question in a systematic way. We investigate the suitability of different types of spectrograms and constant-Q transforms as input representations for neural networks and compare four types of input representations: spectrograms with linearly spaced bins *S*, spectrograms with logarithmically spaced bins *LS*, spectrograms with logarithmically spaced bins and logarithmically scaled magnitude *LM*, as well as the constant-Q transform *CQT* [8]. The filterbank for *LS* and *LM* has a linear response (and lower resolution) for the lower frequencies, and a logarithmic response for the higher frequencies. We vary the sample rate $sr \in \{22050, 44100\}$ [Hz], number of bands per octave $nb \in \{12, 24, 36, 48\}$, whether or not frames undergo circular shift $cs \in \{\text{on}, \text{off}\}$, the amount of zero padding $zp \in \{\times 0, \times 1, \times 2\}$, and whether or not

	sr	zp	cs	nb	$norm$
CQT	×			×	
S	×	×	×		
LS	×	×	×	×	×
LM	×	×	×	×	×

Table 1: For each spectrogram type, these are the parameters that were varied. See text for a description of the value ranges.

to use area normalized filters when filter banks are used $norm \in \{\text{yes}, \text{no}\}$. Furthermore, we re-scale the magnitudes of the spectrogram bins to be in the range $[0, 1]$. Table 1 specifies which parameters are varied for which input type. For the computation of spectrograms we used Madmom [5] and for the constant-Q transform we used the Yaafe library [21].

2.2 Model Class and Suitability

Formally, neural networks are functions with the structure

$$\begin{aligned} \text{net}_k(\mathbf{x}) &= \mathbf{W}_k f_{k-1}(\mathbf{x}) + \mathbf{b}_k \\ f_k(\mathbf{x}) &= \sigma_k(\text{net}_k(\mathbf{x})) \\ f_0(\mathbf{x}) &= \mathbf{x} \end{aligned}$$

where $\mathbf{x} \in \mathbb{R}^{w_{in}}$, $f_k : \mathbb{R}^{w_{k-1}} \to \mathbb{R}^{w_k}$, σ is any element-wise nonlinear function, $\mathbf{W}_k$ is a matrix in $\mathbb{R}^{w_k \times w_{k-1}}$ called the *weight matrix*, and $\mathbf{b}_k \in \mathbb{R}^{w_k}$ is a vector called *bias*. The subscript $k \in \{0, \ldots, L\}$ is the index of the layer, with $k = 0$ denoting the input layer.

Choosing a very narrow definition on purpose, what we mean by a *model class* F is a fixed number of layers L, a fixed number of layer widths $\{w_0, \ldots w_L\}$ and fixed types of nonlinearities $\{\sigma_0, \ldots \sigma_L\}$. A *model* means an instance f from this class, defined by its weights alone. References to the whole collection of weights will be made with Θ.

For the task of framewise piano transcription we define the *suitability* of an input representation in terms of the performance of a simple classifier on this task, when given exactly this input representation.

Assuming we can reliably mitigate the risk of overfitting, we would like to argue that this method of determining suitable input representations, and using them for models with higher capacity, is the best we can do, given a limited computational budget.

Using a low-variance, high-bias model class, the perceptron, also called *logistic regression* or *single-layer* neural network, we learn a *spectral template* per note. To test whether the results stemming from this analysis are really relevant for higher-variance, lower-bias model classes, we run the same set of experiments again, employing a multi layer perceptron with exactly one hidden layer, colloquially called a *shallow net*. This small extension already gives the network the possibility to learn a shared, distributed representation of the input. As we will see, this has a considerable effect on how suitability is judged.

2.3 Nonlinearities and Initialization

Common choices for nonlinearities include the *logistic function* $\sigma(a) = \frac{1}{1+e^{-a}}$, *hyperbolic tangent* $\sigma(a) = \tanh a$, and *rectified linear units (ReLU)* $\sigma(a) = \max(0, a)$. Nonlinearities are necessary to make neural networks universal function approximators [16]. According to [13, 15], using ReLUs as the nonlinearities in neural networks leads to better behaved gradients and faster convergence because they do not saturate.

Before training, the weight matrices are initialized randomly. The scale of this initialization is crucial and depends on the used nonlinearity as well as the number of weights contributing to the activation [13, 15]. Proper initialization plays an even bigger role when networks with more than one hidden layer are trained [31]. This is also important for the transcription setting we use, as the output layer of our networks uses the logistic function, which is prone to saturation effects. Thus we decided on using ReLUs throughout, initialized with a uniform distribution having a scale of $\pm\sqrt{2} \cdot \sqrt{\frac{2}{w_{k-1}+w_k}}$. For the last layer with the logistic nonlinearity, we omit the gain factor of $\sqrt{2}$, as advised in [13].

2.4 Weight Decay

To reduce overfitting and regularizing the network, different priors can be imposed on the network weights. Usually a Gaussian or Laplacian prior is chosen, corresponding to an L_2 or L_1 penalty term on connection weights, added to the cost function $\mathcal{L}_{reg} = \mathcal{L} + \lambda \sum_k \|\text{vec}(\mathbf{W}_k)\|_{1|2}$ [25,32], where $\mathcal{L}$ is an arbitrary, unregularized cost function and λ governs the extent of regularization. Adding both of these penalty terms corresponds to a technique called *Elastic Net* [33]. It is pointed out in [1] that using L_2 regularization plays a similar role as *early stopping* and thus might be omitted. An L_1 penalty on the other hand leads to sparser weights, as it has a tendency to drive weights with irrelevant contributions to zero.

2.5 Dropout

Applying dropout to a layer zeroes out a fraction of the activations of a hidden layer of the network. For each training case, a different random fraction is dropped. This prevents units from co-adapting, and relying too much on each other's presence, as reasoned in [30]. Dropout increases robustness to noise, improves the generalization ability of networks and mitigates the risk of overfitting. Additionally dropout can be interpreted as model-averaging of exponentially many models [30].

2.6 Batch Normalization

Batch normalization [18] seeks to produce networks whose individual activations per layer are zero-mean and unit-variance. This is ensured by normalizing the activations for each mini-batch at each training step. This effectively limits how far the activation distribution can drift away from zero-mean, unit-variance during training. Not only

does this alleviate the need of the weights of the subsequent layer to adapt to a changing input distribution during training, it also keeps the nonlinearities from saturating and in turn speeds up training. It has additional regularizing effects, which become more apparent the more layers a network has. After training is stopped, the normalization is performed for each layer and for the whole training set.

2.7 Layer Types

We employ three different types of layer. Their respective functions can all be viewed as matrix expressions in the end, and thus can be made to fit into the framework described in Section 2.2. For the sake of readability, we simply describe their function in a procedural way.

A *dense* layer consists of a dense matrix - vector pair $(\mathbf{W}, \mathbf{b})$ together with a nonlinearity. The input is transformed via this affine map, and then passed through a nonlinearity.

A *convolutional* layer consists of a number C_k of convolution kernels of a certain size $\{(\mathbf{W}_c, \mathbf{b}_c)\}_{c=0}^{C_k}$ together with a non-linearity. The input is convolved with each convolution kernel, leading to C_k different feature maps to which the same nonlinearity is applied.

Max pooling layers are used in convolutional networks to provide a small amount of translational invariance. They select the units with maximal activation in a local neighborhood (w_t, w_f) in time and frequency in a feature map. This has beneficial effects, as it makes the transcriber invariant to small changes in tuning.

Global average pooling layers are used in all-convolutional networks to compute the mean value of feature maps.

2.8 Architectures

There is a fundamental choice between a network with all dense layers, a network with all convolutional layers, and a mixed approach, where usually the convolutional layers are the first ones after the input layer followed by dense layers. Pooling layers, batch normalization and dropout application are interleaved. For all networks we have to choose the number of layers, how many hidden units per layer to use and when to interleave a regularization layer. For convolutional networks we have to choose the number of filter kernels and their extent in time and frequency direction.

2.9 Networks for Framewise Polyphonic Piano Transcription

The output layer of all considered model classes has 88 units, in line with the playable notes on most modern pianos, and the output nonlinearity is the logistic function, whose output ranges lie in the interval $[0, 1]$.

The loss function being minimized is the frame- and element-wise applied *binary crossentropy*

$$\mathcal{L}_{bce}^{(t)}(\mathbf{y}_t, \hat{\mathbf{y}}_t) = -(\mathbf{y}_t \cdot \log(\hat{\mathbf{y}}_t) + (1 - \mathbf{y}_t) \cdot \log(1 - \hat{\mathbf{y}}_t))$$

where $\hat{\mathbf{y}}_t = f_L(\mathbf{x}_t)$ is the output vector of the network, and $\mathbf{y}_t$ the ground truth at time t. As the overall loss over the whole training set we take the mean

$$\mathcal{L} = \frac{1}{T} \sum_{t=1}^{T} \mathcal{L}_{bce}^{(t)}$$

For the purpose of computing the performance measures, the prediction of the network is thresholded to obtain a binary prediction $\bar{\mathbf{y}}_t = \hat{\mathbf{y}}_t > 0.5$.

2.10 Optimization

The simplest way to adapt the weights Θ of the network to minimize the loss is to take a small step with length α in the direction of steepest descent:

$$\Theta_{i+1} = \Theta_i - \alpha \cdot \frac{\partial \mathcal{L}}{\partial \Theta}$$

Computing the true gradient $\frac{\partial \mathcal{L}}{\partial \Theta} = \frac{1}{T} \sum_{t=1}^{T} \frac{\partial \mathcal{L}_{bce}^{(t)}}{\partial \Theta}$ requires a sum over the length of the whole training set, and is computationally too costly. For this reason, the gradient is usually only approximated from an i.i.d. random sample of size $M \ll T$. This is called *mini-batch stochastic gradient descent*. There are several extensions to this general framework, such as *momentum* [24], *Nesterov momentum* [22] or *Adam* [19], which try to smooth the gradient estimate, correct small missteps or adapt the learning rate dynamically, respectively. Additionally we can set a *learning rate schedule* that controls the temporal evolution of the learning rate.

3. DATASET AND MEASURES

The computational experiments have been performed with the MAPS dataset [12]. It provides MIDI-aligned recordings of a variety of classical music pieces. They were rendered using different hi-quality piano sample patches, as well as real recordings from an upright Disklavier. This ensures clean annotation and therefore almost no label-noise. For all performance comparisons the following framewise measures on the validation set are used:

$$P = \sum_{t=0}^{T-1} \frac{TP[t]}{TP[t] + FP[t]}$$

$$R = \sum_{t=0}^{T-1} \frac{TP[t]}{TP[t] + FN[t]}$$

$$F_1 = \frac{2 \cdot P \cdot R}{P + R}$$

The train-test folds are those used in [26] which were published online [1]. For each fold, the validation set consists of 43 tracks randomly removed from the train set,

[1] http://www.eecs.qmul.ac.uk/~sss31/TASLP/info.html

deviating from the 26 used in [26], and leading to a division of 173-43-54 between the three sets. Note that the test sets are the *same*, and are referred to as *configuration I* in [26]. The exact splits for *configuration II* were not published. We had to choose them ourselves, using the same methodology, which has the additional constraint that *only recordings of the real piano* are used for testing, resulting in a division of 180-30-60. This constitutes a more realistic setting for piano transcription.

4. ANALYSIS OF RELATIVE HYPER-PARAMETER IMPORTANCE

To identify and select an appropriate input representation and determine the most important hyper-parameters responsible for high transcription performance, a multi-stage study with subsequent fANOVA analysis was conducted, as described in [17]. This is similar in spirit to [14], albeit on a smaller scale.

4.1 Types of Representation

To isolate the effects of different input representations on the performance of different model classes, only parameters for the spectrogram were varied according to Table 1. This leads to 204 distinct input representations. The hyper-parameters for the model class as well as the optimization scheme were held *fixed*. To make our estimates more robust, we conducted multiple runs for the same type of input.

The results for each model class are summarized in Table 2, containing the three most influential hyper-parameters and the percentage of variability in performance they are responsible for. The most important hyper-parameter for both model classes is the type of spectrogram used, followed by pairwise interactions. Please note that the numbers in the percentage column are mainly useful to judge the *relative* importance of the parameters. We will see these relative importances put into a larger context later on.

In Figure 1, we can see the mean performance attainable with different types of spectrograms for both model classes. The error bars indicate the standard deviation for the spread in performance, caused by the rest of the varied parameters. Surprisingly, the spectrogram with logarithmically spaced bins and logarithmically scaled magnitude, LM, enables the shallow net to perform best, even though it is a clear mismatch for logistic regression. The lower performance of the constant-Q transform was quite unexpected in both cases and warrants further investigation.

4.2 Greater context

Attempting a full grid search on all possible input representation and model class hyper-parameters described in Section 2 to compute their true marginalized performance is computationally too costly. It is possible however to compute the *predicted* marginalized performance of a hyper-parameter efficiently from a smaller subsample of the space, as shown in [17]. All parameters are randomly varied to sample the space as evenly as possible, and a random forest of 100 regression trees is fitted to the measured performance. This allows to *predict* the marginalized performance of individual hyper-parameters. Table 3 contains the list of hyper-parameters varied.

Model Class	Pct	Parameters
Logistic Regression	48.6%	Spectrogram Type
	16.9%	Spectrogram Type × Normed Area Filters
	10.4%	Spectrogram Type × Sample Rate
Shallow Net	68.4%	Spectrogram Type
	20.8%	Spectrogram Type × Sample Rate
	5.7%	Sample Rate

Table 2: The three most important parameters determining input representation for different model classes

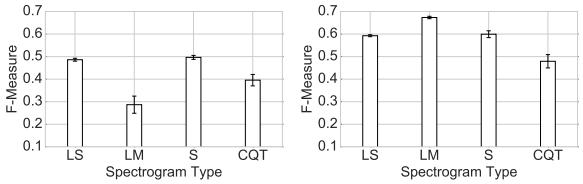


Figure 1: (a) Mean logistic regression performance dependent on spectrogram (b) Mean shallow net performance dependent on type of spectrogram

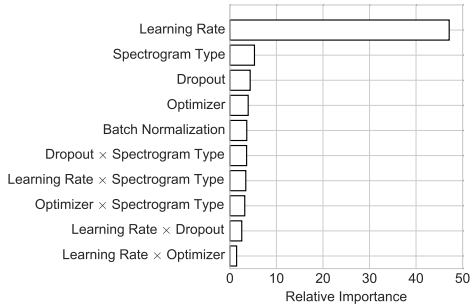


Figure 2: Relative importance of the first 10 hyper-parameters for the *shallow net* model class.

The percentage of variance in performance these hyper-parameters are responsible for, can be seen in Figure 2 for the 10 most important ones. A total of 3000 runs with random parameterizations were made.

Analyzing the results of all the runs tells us that the most important hyper-parameters are *Learning Rate* (47.11%), and *Spectrogram Type* (5.28%). The importance of the learning rate is in line with the findings in [14]. Figure 2 shows the relative importances of the first 10 hyper-parameters, and Figure 3 shows the predicted marginal performance of the learning rate dependent on its value (on a logarithmic scale) in greater detail.

Optimizer (Plain SGD, Momentum, Nesterov Momentum, Adam)
Learning Rate (0.001, 0.01, 0.1, 0.5, 1.0, 2.0, 10.0, 50.0, 100.0)
Momentum (Off, 0.7, 0.8, 0.9)
Learning rate Scheduler (On, Off)
Batch Normalization (On, Off)
Dropout (Off, 0.1, 0.3, 0.5)
L_1 Penalty (Off, 1e-07, 1e-08, 1e-09)
L_2 Penalty (Off, 1e-07, 1e-08, 1e-09)

Table 3: The list of additional hyper-parameters varied, and their ranges.

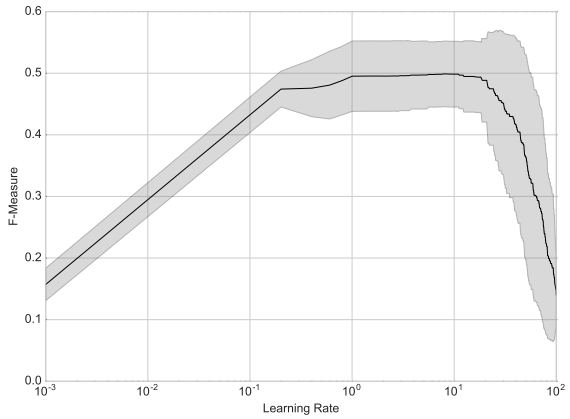


Figure 3: Mean predicted performance for the *shallow net* model class, dependent on learning rate (on a logarithmic scale). The dark line shows the mean performance, and the gray area shows the standard deviation.

5. STATE OF THE ART MODELS

Having completed the analysis of input representation, more powerful model classes were tried: a deep neural network consisting entirely of dense layers (*DNN*), a mixed network with convolutional layers directly after the input followed by dense layers (*ConvNet*), and an all-convolutional network (*AllConv* [29]). Their architectures are described in detail in Table 4. To the best of our knowledge, this is the first time an all-convolutional net has been proposed for the task of framewise piano transcription.

We computed a logarithmically filtered spectrogram with logarithmic magnitude from audio with a sample rate of 44.1 kHz, a filterbank with 48 bins per octave, normed area filters, no circular shift and no zero padding. The choices for circular shift and zero padding ranged very low on the importance scale, so we simply left them switched off. This resulted in only 229 bins, which are logarithmically spaced in the higher frequency regions, and almost linearly spaced in the lower frequency regions as mentioned in Section 2.1. The dense network was presented one frame at a time, whereas the convolutional network was given a context in time of two frames to either side of the current frame, summing to 5 frames in total.

All further hyper-parameter tuning and architectural choices have been left to a human expert. Models within a model class were selected based on average F-measure across the four validation sets. An automatic search via a hyper-parameter search algorithm for these larger model classes, as described in [4, 11, 28] is left for future work (the training time for a convolutional model is roughly $8-9$ hours on a Tesla K40 GPU, which leaves us with $204{\cdot}3{\cdot}4{\cdot}8$ hours (variants $\times$ #models $\times$ #folds $\times$ hours per model), or on the order of $800 - 900$ days of compute time to determine the best input representation exactly).

	DNN	*ConvNet*	*AllConv*
	Input 229	Input 5x229	Input 5x229
	Dropout 0.1	Conv 32x3x3	Conv 32x3x3
	Dense 512	Conv 32x3x3	Conv 32x3x3
	BatchNorm	BatchNorm	BatchNorm
	Dropout 0.25	MaxPool 1x2	MaxPool 1x2
	Dense 512	Dropout 0.25	Dropout 0.25
	BatchNorm	Conv 64x3x3	Conv 32x1x3
	Dropout 0.25	MaxPool 1x2	BatchNorm
	Dense 512	Dropout 0.25	Conv 32x1x3
	BatchNorm	Dense 512	BatchNorm
	Dropout 0.25	Dropout 0.5	MaxPool 1x2
	Dense 88	Dense 88	Dropout 0.25
			Conv 64x1x25
			BatchNorm
			Conv 128x1x25
			BatchNorm
			Dropout 0.5
			Conv 88x1x1
			BatchNorm
			AvgPool 1x6
			Sigmoid
# Params	691288	1877880	284544

Table 4: Model Architectures

For these powerful models, we followed practical recommendations for training neural networks via gradient descent found in [1]. Particularly relevant is the way of setting the initial learning rate. Strategies that dynamically adapt the learning rate, such as *Adam* or *Nesterov Momentum* [19, 22] help to a certain extent, but still do not spare us from tuning the initial learning rate and its schedule.

We observed that using a combination of batch normalization and dropout together with very simple optimization strategies leads to low validation error fairly quickly, in terms of the number of epochs trained. The strategy that worked best for determining the learning rate and its schedule was trying learning rates on a logarithmic scale, starting at 10.0, until the optimization did not diverge anymore [1], then training until the validation error flattened out for a few epochs, then multiplying the learning rate with a factor from the set $\{0.1, 0.25, 0.5, 0.75\}$. The rates and schedules we finally settled on were:

- *DNN*: SGD with Momentum, $\alpha = 0.1, \mu = 0.9$ and halving of α every 10 epochs
- *ConvNet*: SGD with Momentum, $\alpha = 0.1, \mu = 0.9$ and a halving of α every 5 epochs
- *AllConv*: SGD with Momentum, $\alpha = 1.0, \mu = 0.9$ and a halving of α every 10 epochs

The results for framewise prediction on the MAPS dataset can be found in Table 6. It should be noted that we compare straightforward, simple, and largely un-smoothed

systems (ours) with hybrid systems [26]. There is a small degree of temporal smoothing happening when processing spectrograms with convolutional nets. The term *simple* is supposed to mean that the resulting models have a small amount of parameters and the models are composed of a few low-complexity building blocks. All systems are evaluated on the same train-test splits, referred to as *configuration I* in [26] as well as on *realistic* train-test splits, that were constructed in the same fashion as *configuration II* in [26].

Model Class	P	R	F_1
Hybrid DNN [26]	65.66	70.34	67.92
Hybrid RNN [26]	67.89	70.66	69.25
Hybrid ConvNet [26]	72.45	76.56	74.45
DNN	76.63	70.12	73.11
ConvNet	80.19	**78.66**	**79.33**
AllConv	**80.75**	75.77	78.07

Table 5: Results on the MAPS dataset. Test set performance was averaged across 4 folds as defined in *configuration I* in [26].

Model Class	P	R	F_1
DNN [26]	-	-	59.91
RNN [26]	-	-	57.67
ConvNet [26]	-	-	64.14
DNN	75.51	57.30	65.15
ConvNet	74.50	**67.10**	**70.60**
AllConv	**76.53**	63.46	69.38

Table 6: Results on the MAPS dataset. Test set performance was averaged across 4 folds as defined in *configuration II* in [26].

6. CONCLUSION

We argue that the results demonstrate: the importance of proper choice of input representation, and the importance of hyper-parameter tuning, especially the tuning of learning rate and its schedule; that convolutional networks have a distinct advantage over their deep and dense siblings, because of their context window and that all-convolutional networks perform nearly as well as mixed networks, although they have far fewer parameters. We propose these straightforward, framewise transcription networks as a new state-of-the art baseline for framewise piano transcription for the MAPS dataset.

7. ACKNOWLEDGEMENTS

This work is supported by the European Research Council (ERC Grant Agreement 670035, project CON ESPRESSIONE), the Austrian Ministries BMVIT and BMWFW, the Province of Upper Austria (via the COMET Center SCCH) and the European Union Seventh Framework Programme FP7 / 2007-2013 through the GiantSteps project (grant agreement no. 610591). We would like to thank all developers of Theano [3] and Lasagne [9] for providing comprehensive and easy to use deep learning frameworks. The Tesla K40 used for this research was donated by the NVIDIA Corporation.

8. REFERENCES

[1] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*, pages 437–478. Springer, 2012.

[2] Taylor Berg-Kirkpatrick, Jacob Andreas, and Dan Klein. Unsupervised Transcription of Piano Music. In *Advances in Neural Information Processing Systems*, pages 1538–1546, 2014.

[3] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU Math Expression Compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, 2010.

[4] James Bergstra, Dan Yamins, and David D. Cox. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in Science Conference*, pages 13–20, 2013.

[5] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. madmom: a new Python Audio and Music Signal Processing Library. *arXiv preprint arXiv:1605.07008*, 2016.

[6] Sebastian Böck and Markus Schedl. Polyphonic piano note transcription with recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 121–124. IEEE, 2012.

[7] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

[8] Judith C. Brown. Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*, 89(1):425–434, 1991.

[9] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Daniel Maturana, Martin Thoma, Eric Battenberg, Jack Kelly, Jeffrey De Fauw, Michael Heilman, diogo149, Brian McFee, Hendrik Weideman, takacsg84, peterderivaz, Jon, instagibbs, Dr. Kashif Rasul, CongLiu, Britefury, and Jonas Degrave. Lasagne: First release., August 2015.

[10] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 6964–6968. IEEE, 2014.

[11] Katharina Eggensperger, Matthias Feurer, Frank Hutter, James Bergstra, Jasper Snoek, Holger Hoos, and Kevin Leyton-Brown. Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In *NIPS workshop on Bayesian Optimization in Theory and Practice*, 2013.

[12] Valentin Emiya, Roland Badeau, and Bertrand David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(6):1643–1654, 2010.

[13] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[14] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *arXiv preprint arXiv:1503.04069*, 2015.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *arXiv preprint arXiv:1502.01852*, 2015.

[16] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[17] Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. An efficient approach for assessing hyperparameter importance. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 754–762, 2014.

[18] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[19] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[20] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436444, May 2015.

[21] Benoit Mathieu, Slim Essid, Thomas Fillon, Jacques Prado, and Gaël Richard. YAAFE, an Easy to Use and Efficient Audio Feature Extraction Software. In *Proceedings of the International Conference on Music Information Retrieval*, pages 441–446, 2010.

[22] Yurii Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[23] Ken O'Hanlon and Mark D. Plumbley. Polyphonic piano transcription using non-negative matrix factorisation with group sparsity. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 3112–3116. IEEE, 2014.

[24] Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[25] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088), 1986.

[26] Siddartha Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939, May 2016.

[27] Siddhartha Sigtia, Emmanouil Benetos, Nicolas Boulanger-Lewandowski, Tillman Weyde, Artur S. d'Avila Garcez, and Simon Dixon. A hybrid recurrent neural network for music transcription. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 2061–2065. IEEE, 2015.

[28] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[29] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for Simplicity: The All Convolutional Net. *arXiv preprint arXiv:1412.6806*, 2014.

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[31] David Sussillo. Random Walks: Training Very Deep Nonlinear Feed-Forward Networks with Smart Initialization. *arXiv preprint arXiv:1412.6558*, 2014.

[32] Peter M. Williams. Bayesian Regularization and Pruning Using a Laplace Prior. *Neural Computation*, 7(1):117–143, Jan 1995.

[33] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

PHRASE-LEVEL AUDIO SEGMENTATION OF JAZZ IMPROVISATIONS INFORMED BY SYMBOLIC DATA

Jeff Gregorio and Youngmoo E. Kim
Drexel University, Dept. of Electrical and Computer Engineering
{jgregorio, ykim}@drexel.edu

ABSTRACT

Computational music structure analysis encompasses any model attempting to organize music into qualitatively salient structural units, which can include anything in the heirarchy of large scale form, down to individual phrases and notes. While much existing audio-based segmentation work attempts to capture repetition and homogeneity cues useful at the form and thematic level, the time scales involved in phrase-level segmenation and the avoidance of repetition in improvised music necessitate alternate approaches in approaching jazz structure analysis. Recently, the Weimar Jazz Database has provided transcriptions of solos by a variety of eminent jazz performers. Utilizing a subset of these transcriptions aligned to their associated audio sources, we propose a model based on supervised training of a Hidden Markov Model with ground-truth state sequences designed to encode melodic contours appearing frequently in jazz improvisations. Results indicate that representing likely melodic contours in this way allows a low-level audio feature set containing primarily timbral and harmonic information to more accurately predict phrase boundaries.

1. INTRODUCTION

Music structure analysis is an active area of research within the Music Information Retrieval (MIR) community with utility extending to a wide range of MIR applications including song similarity, genre recognition, audio thumbnailing, music indexing systems, among others. Musical structure can be defined in terms of any qualitatively salient unit, from large scale form (e.g. intro, verse, chorus, etc.), to melodic themes and motifs, down to individual phrases and notes.

Paulus [8] categorizes existing approaches to audio-based structural analysis according to perceptual cues assumed to have central importance in determination of structure, namely into those based on *repetition*, *novelty*, and *homogeneity*. A music structure analysis task typically involves a boundary detection step, where individual sections are assumed to be homogeneous, and transitions between sections associated with a high degree of novelty. Novelty is assumed to be indicated by large changes in one or more time-series feature representations that may correspond to perceptually-salient shifts in timbre, rhythm, harmony, or instrumentation. Predicted boundaries can then be used to obtain segments which can be grouped according to similarity in the employed feature space(s). Alternatively, repetition-based methods may be used to identify repeated segments and boundaries directly.

Due to the difficulty in reliabily estimating individual note onsets and pitches, much existing work on music segmentation at the phrase level has been limited to single instruments in the symbolic domain. In a meta-analysis of symbolic phrase segmentation work, Rodríguez López [7] showed that two of the best performing rule-based models in comparative studies include Cambouropoulos's Local Boundary Detection Model (LBDM) [2] and Temperley's *Grouper* [11]. Both relate to Gestalt discontinuity principles, placing phrase boundaries using heuristics derived in part from features of consecutive note onset times, including inter-onset intervals (IOI) and offset-onset intervals (OOI). The LBDM model additionally uses pitch contour information, assuming discontinuity strength is increased by large inter-pitch intervals (IPI). *Grouper* also incorporates knowledge of metrical context and assumes a prior distribution of phrase lengths.

The proposed work focuses on musical structure at the phrase level, specifically identification of phrase boundaries from audio signals. Though we do not directly predict note onsets, durations, or pitches available in symbolic representations, we take advantage of audio-aligned MIDI transcriptions in the supervised training of a Hidden Markov Model (HMM). Using a primarily timbral and harmonic audio feature represenation, we hope to aid in the prediction of phrase boundaries by exploiting correlations between timbral/harmonic cues and common melodic phrase contours represented in the dataset.

2. MOTIVATION

In the audio domain, most existing structural segmentation work attempts to model to large scale form. The self distance matrix (SDM) is a useful representation in this modality, where entries $SDM(i,j) = d(\mathbf{x_i}, \mathbf{x_j})$ represent the distance between all combinations of feature vectors $\mathbf{x_i}$ and $\mathbf{x_j}$ by some distance metric d. This representation lends itself well to identifiable patterns associated with ho-

© Jeff Gregorio and Youngmoo E. Kim. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jeff Gregorio and Youngmoo E. Kim. "Phrase-level Audio Segmentation of Jazz Improvisations Informed by Symbolic Data", 17th International Society for Music Information Retrieval Conference, 2016.

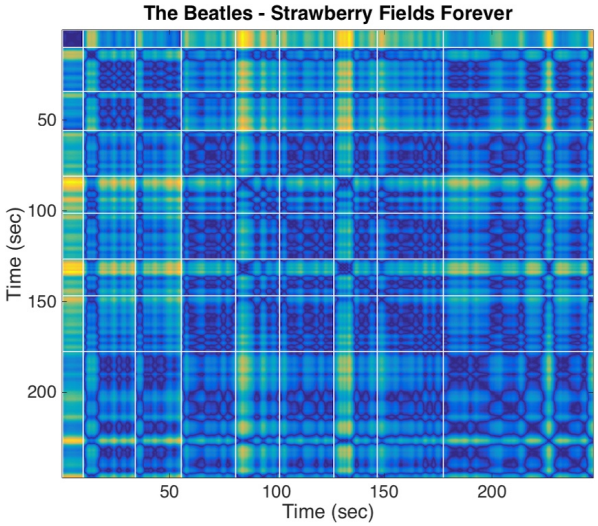


Figure 1. Self-distance matrix with form-level annotations plotted as white lines. Section boundaries often coincide with large blocks of relatively small distance, and some repetitions can be seen as stripes parallel to the main diagonal.

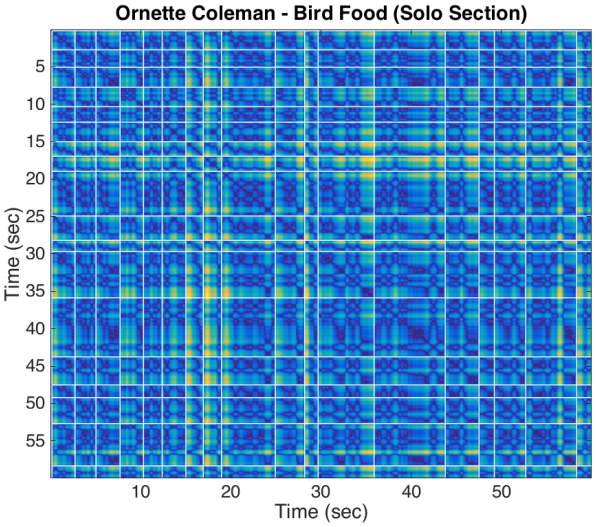


Figure 2. Self-distance matrix with annotated phrase boundaries plotted as white lines. Note the absence of off-diagonal striping patterns indicative of repetition, and the infrequent occurrence of large homogeneous blocks over the duration of entire phrases.

mogeneity, novelty, and repetition principles. Homogeneity within a section is generally associated with low-valued blocks representing small distance, novelty can be seen in the form of transitions between low and high value, and repetition manifests as stripes of low value parallel to the main diagonal. Figure 1 shows an example SDM computed using the timbral and harmonic feature space described in Section 4. Note this matrix is smoothed by averaging distance values from multiple frames around each index, as described in [8], but hasn't been filtered or beat-aligned to enhance repetition patterns, though some are visible.

When attempting audio segmentation at the phrase level, overall feature space homogeneity within single segments may be an unsafe assumption given the shorter time scales involved, in which a performer might employ expressive modulation of timbre. Furthermore, while melodic ideas in a jazz improvisation may be loosely inspired by a theme, extended repetition is usually avoided in favor of maximally unique melodies within a single performance. This context suggests that repetition-based approaches useful for identifying large-scale forms and themes may be inappropriate. Figure 2 shows an example SDM computed at the same resolution as Figure 1 over 60 seconds of a jazz improvisation. Note that while this SDM contains block patterns associated with homogeneity, they don't necessarily align well with entire phrases. This SDM is also almost completely missing any identifiable repetition patterns.

There does exist, however, some degree of predictability in jazz phrase structure that an ideal model should exploit, albeit across a corpus rather than within a single track. We propose a system based on supervised training of a Hidden Markov Model with a low-level audio feature set designed to capture novelty in the form of large timbral and harmonic shifts indicative of phrase boundaries. Existing HMM approaches have included unsupervised training, with a fixed-number of hidden states assumed to correspond to form-level sections [9, 10], instrument mixtures [5], or simply a mid-level feature representation [6]. Our model differs from existing HMM-based approaches in that it attempts to represent common elements of jazz phrase structure directly in the topology of the network, where the ground-truth state sequences are derived from inter-pitch intervals in audio-aligned transcriptions. Likely sequences of predicted states should therefore correspond to melodic contours well-represented in the training data, aiding in the detection of phrase boundaries.

3. DATASET AND PREPROCESSING

In 2014, the first version of the Weimar Jazz Database (*WJazzD*) was released as part of the larger Jazzomat Research Project [1]. The database contains transcriptions of monophonic solos by eminent jazz performers, well representing the evolution of the genre over the 20th century. The database was later expanded to include 299 solo transcriptions from 225 tracks, 70 performers, 11 instruments (soprano/alto/tenor/tenor-c/baritone sax, clarinet, trumpet, cornet, trombone, vibraphone, and guitar) and 7 styles (Traditional, Swing, Bebop, Hardbop, Cool, Postbop, and Free Jazz). The transcriptions were initially generated using state-of-the-art automatic transcription tools and manually corrected by musicology students. In addition to the transcriptions, the database contains a rich collection of metadata and human labels including phrase boundaries, underlying chord changes, form-level sections, and beat locations.

3.1 MIDI to Audio Alignment

The lack of availability of the original audio tracks used as source material for the database's transcriptions presents some difficulty in taking full advantage of the possibilities for supervised machine learning methods using acoustic features. Toward this end, we were able to obtain 217 of 225 tracks containing the solo(s) as transcribed. Metadata available in *WJazzD* indicate the starting and ending timestamps of the solo sections at a 1-second resolution, which is insufficient for determining ground truth for phrase boundaries associated with note onset times. Additionally, pulling audio files from various sources introduces further uncertainty, as many tracks appear on both original releases and compilations which may differ slightly in duration or other edits.

To obtain ground truth, we trim the original tracks according to provided solo timestamps and employ a tool created by Dan Ellis [4] which uses Viterbi alignment on beat-tracked versions of original audio and resynthesized MIDI to modify and output an aligned MIDI file. Upon inspection, 90 extracted solos produced a suitable alignment that required minimal manual corrections. We parse the database and handle conversion to and from MIDI format in Matlab using the MIDI Toolbox [3], making extensive use of the convenient note matrix format and pianoroll visualizations.

4. AUDIO FEATURES

To represent large timbral shifts, we use spectral flux and centroid features derived from the short-time Fourier transform (STFT), and spectral entropy derived from the power spectral density (PSD) of the audio tracks, sampled at $22050Hz$ with a FFT size of 1024 samples, Hamming windowed, with 25% overlap. Due to our interest in the lead instrument only, features are computed on a normalized portion of the spectrum between $500 - 5000Hz$ to remove the influence of prominent bass lines while preserving harmonic content of the lead instrument.

We also compute two features based on Spectral Contrast, a multi-dimensional feature computed as the decibel difference between the largest and smallest values in seven octave bands of the STFT. Since the resolution of this feature is dependent on the number of frequency bins in each octave, we use a larger FFT of size 4096, which gives meaningful values in four octaves above $800Hz$ without sacrificing much time resolution. The first feature reduces spectral contrast to one dimension by taking the mean difference of all bands between frames. Observing that large positive changes in spectral contrast correlate well with annotated phrase boundaries, we half-wave rectify this feature. The second feature takes the seven-dimensional Euclidian distance between spectral contrast frames.

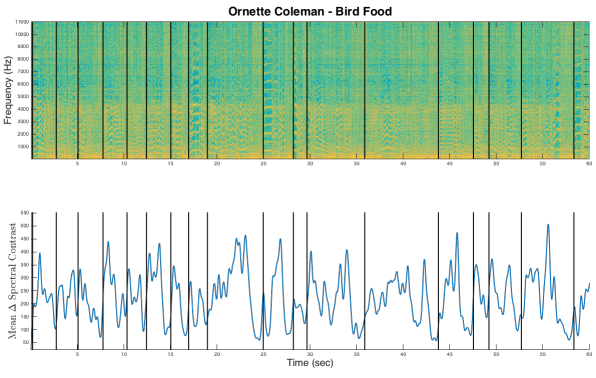


Figure 3. Mean positive difference between spectral contrast frames, plotted against annotated phrase boundaries.

Finally, we include a standard Chromagram feature, which is a 12-dimensional feature representing the contribution in the audio signal to frequencies associated with the twelve semitones in an octave. While the chromagram includes contributions from fundamental frequencies of interest, it also inevitably captures harmonics and un-pitched components. Noting that even precise knowledge of absolute pitch of the lead instrument would be uninformative in determining whether any note were the begining of a phrase, we collapse this feature to a single dimension by taking the Euclidian distance between frames, with the intention of capturing harmonic shifts that may be correlated with phrase boundaries and melodic contours.

All features are temporally smoothed by convolving with a gaussian kernel of 21 samples. All elements in the feature vectors are squared to emphasize peaks. We then double the size of the feature set by taking the first time difference of each feature, which amounts to a second difference for the spectral contrast and chroma features. Later, when evaluating, each feature in the training and testing sets is standardized to zero mean and unit variance using statistics of the training set features.

5. MELODIC CONTOUR VIA HMM

Hidden Markov Models represent the joint distribution of some hidden state sequence $\mathbf{y} = \{y_1, y_2, ..., y_N\}$ and a corresponding sequence of observations $\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_N}\}$, or equivalently the state transition probabilities $P(y_i|y_{i-1})$ and emission probabilities $P(\mathbf{x_i}|y_i)$.

HMMs have been used in various forms for music structure analysis, lending well to the sequential nature of the data, with hidden states often assumed to correspond to some perceptually meaningful structural unit. Unsupervised approaches use feature observations and an assumed number of states as inputs to the Baum-Welch algorithm to estimate the model parameters, which can then be used with the Viterbi algorithm to estimate the most likely sequence of states to have produced the observations.

Paulus notes that unsupervised HMM-based segmentation tends to produce unsatisfactory results on form-level segmentation tasks due to observations relating to individual sound events [8], a shortcoming which has led to observed state sequences being treated as a mid-level representations in subsequent work. We revisit HMMs as a segmentation method specifically for phrase-level analysis due to the particular importance of parameters of individual sound events rather than longer sections. Specifi-

cally, we postulate that phrases drawn from the jazz vocabulary follow predictable melodic contours, and incorporating ground-truth knowledge of the distribution and temporal evolution of these contours as observed in a large dataset of phrases through supervised training may help in identification of phrase boundaries.

6. EXPERIMENTS

We first evaluate a 2-state HMM, with states $y_i \in \{0, 1\}$ corresponding to the absence or presence (respectively) of the lead instrument during the i^{th} audio frame. Though a 2-state graphical model is trivial and offers no advantages over any other supervised classification method, we include it here simply as a basis for comparison with the multi-state models to evaluate the efficacy of adding states based on ground-truth pitch contour.

To estimate an upper bound on expected performance of our audio-based models, we evaluate two symbolic segmentation models using features of precisely known note pitches and onset times. First, we evaluate the Local Boundary Detection Model (LBDM) implementation offered by the MIDI Toolbox [3]. The LBDM outputs a continuous boundary strength, which we use to tune a boundary prediction threshold for maximum f-score via cross validation. Second, we train a 2-state HMM, where the model state y_i then corresponds to the i^{th} note rather than the i^{th} audio frame, and takes the value 1 if the note is the first in a phrase, and 0 otherwise. Observations $\mathbf{x_i}$ similarly correspond to features of individual note events including the IOI, OOI, and IPI. Results of the symbolic segmentation models are shown in Table 1(a).

To directly encode melodic contour in the network topology for the multi-state, audio-based HMMs, we extract ground-truth state sequences based on quantization levels of the observed inter-pitch interval (IPI) in the transcription. The following indicate the state of the network following each IPI, where the state remains for the duration of the note, rounded to the nearest audio frame:

$$\textbf{5-State} \quad y_i = \begin{cases} 0 & \text{lead instrument absent} \\ 1 & \text{first phrase note} \\ 2 & IPI < 0 \\ 3 & IPI = 0 \\ 4 & IPI > 0 \end{cases}$$

$$\textbf{7-State} \quad y_i = \begin{cases} 0 & \text{lead instrument absent} \\ 1 & \text{first phrase note} \\ 2 & IPI < -5 \\ 3 & -5 \leq IPI < 0 \\ 4 & IPI = 0 \\ 5 & 0 < IPI \leq 5 \\ 6 & IPI > 5 \end{cases}$$

The 5-state model simply encodes increasing/decreasing/unison pitch in the state sequence. The 7-state model further quantizes increasing and decreasing pitch into intervals greater than and less than a perfect fourth. Each HMM requires a discrete observation sequence, so the 10-dimensional audio feature set described in Section 4 is discretized via clustering using a Gaussian Mixture Model (GMM) with parameters estimated via Expectation-Maximization (EM).

We note that in the solo transcriptions, there are many examples of phrase boundaries that occur between two notes played legato (i.e. the offset-onset interval is zero or less than the time duration of a single audio frame). When parsing the MIDI data and associated boundary note annotations to determine the state sequence for each audio frame, in any such instance where state 1 is not preceded by state 0, we force a 0-1 transition to allow the model to account for phrase boundaries that aren't based primarily on temporal discontinuity cues.

7. RESULTS

Evaluation of each network is performed via six fold cross-validation, where each fold trains the model on five styles as provided by the *WJazzD* metadata, and predicts on the remaining style. We note that *WJazzD* encompasses seven styles, but the 90 examples successfully aligned to corresponding audio tracks did not include any traditional jazz. Though the sequence of states predicted by the model include the contour-based states, our reported results only consider accuracy in predicting a transition to state 1 in all cases.

Precision, recall, and f-score metrics reported in form-level segmentation experiments typically consider a true positive to be a boundary identified within 0.5 and 3 seconds of the ground truth. Considering the short time scales involved with phrase-level segmentation, we report metrics considering a true positive to be within one beat and one half beat, as determined using each solo's average tempo

Model	P_n	R_n	F_n
LBDM	0.7622	0.7720	0.7670
HMM, 2-State	0.8225	0.8252	0.8239

(a) Symbolic models

Model	P_{1B}	R_{1B}	F_{1B}
HMM, 2-State	0.6114	0.5584	0.5837
HMM, 5-State	0.5949	0.6586	0.6251
HMM, 7-State	0.6116	0.6565	0.6333

(b) Audio models, true positive within one beat of annotation

Model	$P_{0.5B}$	$R_{0.5B}$	$F_{0.5B}$
HMM, 2-State	0.4244	0.3876	0.4052
HMM, 5-State	0.4039	0.4472	0.4245
HMM, 7-State	0.4212	0.4521	0.4361

(c) Audio models, true positive within half beat of annotation

Table 1. Segmentation results

annotation provided by *WJazzD*. For reference, the mean time per beat in the 90 aligned examples is 0.394 seconds, with a standard deviation of 0.178 seconds. All beat durations were less than 1 second.

We report precision, recall, and f-score computed over all examples, all folds, and 5 trials. Reported results use 30 Gaussian components as discrete observations in the audio-based models, and 5 components for the symbolic model, and are summarized in Table 1. For greater insight into the model's performance in different stylistic contexts, we also present the cross-validation results across the six styles in Figure 4.

8. DISCUSSION

ANOVA and post-hoc analysis reveals both multi-state models yielding increased recall over the 2-state model ($F_{2,1332} = 30.62$, $p < 10^{-13}$), and increased f-score ($F_{2,1332} = 11.28$, $p < 10^{-4}$) with no significant difference in precision. Interestingly, the most significant recall increases from addition of the melodic contour states within styles include hardbop ($F_{2,297} = 15.68, p < 10^{-6}$), postbop ($F_{2,432} = 12.22$, $p < 10^{-5}$), and swing styles ($F_{2,177} = 6.73, p < 10^{-3}$).

These increases in recall within a style also correlate well with a high proportion of occurrences of phrase boundaries with no temporal discontinuity. These account for 22% of all phrase boundaries in hardbop, 18% in postbop, and 29% in swing, while accounting for 17%, 7%, and 4% in bebop, cool, and free jazz, respectively. We believe this suggests that incorporating ground-truth melodic contour allows the model to account for the relationship between contours indicative of phrase boundaries and their associated timbral and harmonic shifts.

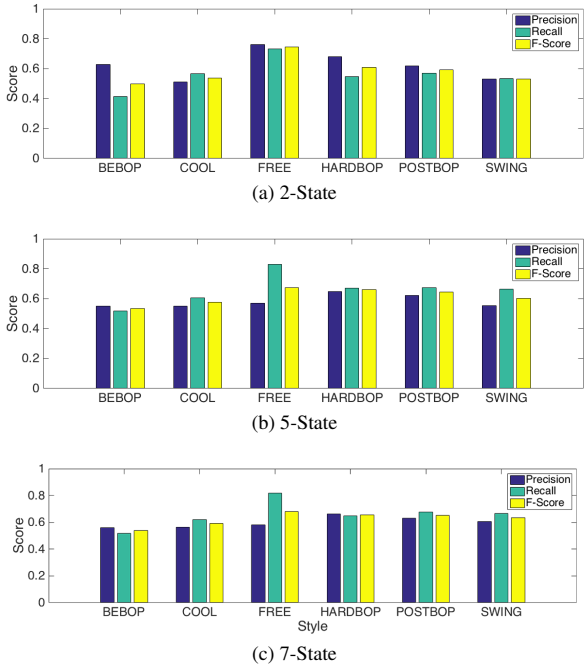
Manual inspection of segmentation results tend to reinforce this idea, as shown in Figure 5. The 2-state model fails to identify four phrase boundaries preceded by very small inter-onset intervals (6th, 15th, 18th, and 21st phrases), while the 7-state model correctly identifies three (6th, 18th, and 21st), at the cost of some tendency toward over-segmentation (in this case).

9. CONCLUSIONS

Evaluation of a 2-state HMM established a baseline phrase segmentation accuracy by detecting the presence or absence of the lead instrument, which presents some difficulty in predicting phrase boundaries based on harmonic and melodic cues with little to no temporal discontinuity. Incorporating a ground-truth state sequence in the multi-state HMMs using melodic contour information derived from the transcription yielded statistically significant increases in recall in styles containing a high proportion of these phrase boundaries.

Although our feature set does not attempt to predict pitches of individual notes, we believe the increased recall associated with the multi-state models indicates the model is exploiting a relationship between timbral and harmonic observations and melodic contours associated with phrase boundaries. These precise relationships are undoubtedly dependent on the timbre of the instrument, yet demonstrate some general utility when trained on a range of lead instruments.

While the attempted representation of melodic contour

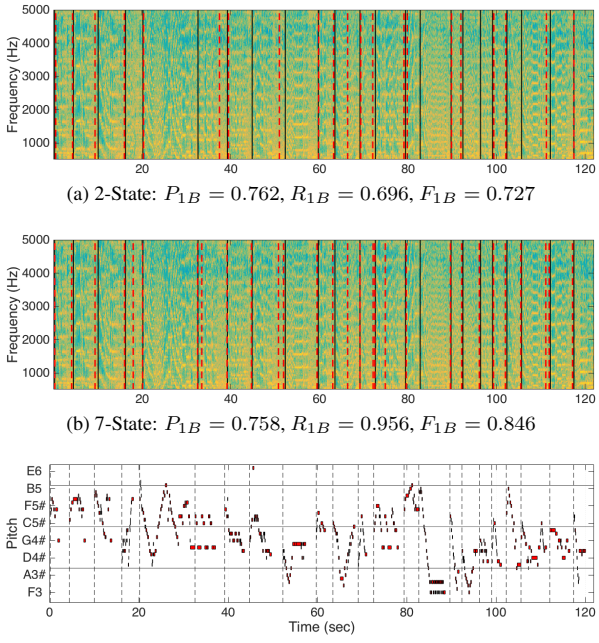


(a) 2-State

(b) 5-State

(c) 7-State

Figure 4. Audio-based HMM segmentation results by style. Significant ($p < 0.005$) increases in recall and f-score observed in Hardbop, Postbop, and Swing.



(a) 2-State: $P_{1B} = 0.762$, $R_{1B} = 0.696$, $F_{1B} = 0.727$

(b) 7-State: $P_{1B} = 0.758$, $R_{1B} = 0.956$, $F_{1B} = 0.846$

(c) MIDI Transcription

Figure 5. Segmentation of Freddie Hubbard's solo in the Eric Dolphy track "245". Black lines indicate ground-truth annotations, and red lines show predicted boundaries.

in the model topology indicates some promise, we believe there are likely better alternatives to modeling contour than arbitrary quanitzation of ground truth inter-pitch intervals. Future work should examine the potential of assembling observed contours from a smaller set of contour primitives over longer time scales than note pair transitions. Furthermore, though our approach avoided relying high-level pitch estimates derived from the audio because of strong potential for propagation of errors, we will investigate the use of mid-level pitch salience functions in future feature sets.

More generally, we believe that the availability of well-aligned audio and symbolic data can allow the use of supervised methods as a precursor to more scalable audio-based methods, and aid in the creation of mid-level features useful for a wide range of MIR problems.

10. REFERENCES

[1] Jakob Aeßler, Klaus Frieler, Martin Pfleiderer, and Wolf-Georg Zaddach. Introducing the jazzomat project - jazz solo analysis using music information retrieval methods. In *In: Proceedings of the 10th International Symposium on Computer Music Multidisciplinary Research (CMMR) Sound, Music and Motion, Marseille, Frankreich.*, 2013.

[2] Emilios Cambouropoulos. *Music, Gestalt, and Computing: Studies in Cognitive and Systematic Musicology*, chapter Musical rhythm: A formal model for determining local boundaries, accents and metre in a melodic surface, pages 277–293. Springer Berlin Heidelberg, Berlin, Heidelberg, 1997.

[3] Tuomas Eerola and Petri Toiviainen. *MIDI Toolbox: MATLAB Tools for Music Research*. University of Jyväskylä, Jyväskylä, Finland, 2004.

[4] D.P.W. Ellis. Aligning midi files to music audio, web resource, 2013. `http://www.ee.columbia.edu/~dpwe/resources/matlab/alignmidi/`. Accessed 2016-03-11.

[5] Sheng Gao, N. C. Maddage, and Chin-Hui Lee. A hidden markov model based approach to music segmentation and identification. In *Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference on*, volume 3, pages 1576–1580 vol.3, Dec 2003.

[6] M. Levy and M. Sandler. Structural segmentation of musical audio by constrained clustering. *Trans. Audio, Speech and Lang. Proc.*, 16(2):318–326, February 2008.

[7] Marcelo Rodrguez Lpez and Anja Volk. Melodic segmentation: A survey. Technical Report UU-CS-2012-015, Department of Information and Computing Sciences, Utrecht University, 2012.

[8] Jouni Paulus, Meinard Müller, and Anssi Klapuri. State of the art report: Audio-based music structure analysis. In *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010*, pages 625–636, 2010.

[9] Geoffroy Peeters, Amaury La Burthe, and Xavier Rodet. Toward automatic music audio summary generation from signal analysis. In *In Proc. International Conference on Music Information Retrieval*, pages 94–100, 2002.

[10] Mark Sandler and Jean-Julien Aucouturier. Segmentation of musical signals using hidden markov models. In *Audio Engineering Society Convention 110*, May 2001.

[11] David Temperley. *The cognition of basic musical structures*. MIT Press, 2004.

REVISITING PRIORITIES: IMPROVING MIR EVALUATION PRACTICES

Bob L. Sturm
Centre for Digital Music, Queen Mary University of London

ABSTRACT

While there is a consensus that evaluation practices in music informatics (MIR) must be improved, there is no consensus about what should be prioritised in order to do so. Priorities include: 1) improving data; 2) improving figures of merit; 3) employing formal statistical testing; 4) employing cross-validation; and/or 5) implementing transparent, central and immediate evaluation. In this position paper, I argue how these priorities treat only the symptoms of the problem and not its cause: MIR lacks a formal evaluation framework relevant to its aims. I argue that the principal priority is to adapt and integrate the formal design of experiments (DOE) into the MIR research pipeline. Since the aim of DOE is to help one produce the most reliable evidence at the least cost, it stands to reason that DOE will make a significant contribution to MIR. Accomplishing this, however, will not be easy, and will require far more effort than is currently being devoted to it.

1. CONSENSUS: WE NEED BETTER PRACTICES

I recall the aims of MIR research in Sec. 1.1, and the importance of evaluation to this pursuit. With respect to these, I describe the aims and shortcomings of MIREX in Sec. 1.2. These motivate the seven evaluation challenges of the MIR "Roadmap" [23], summarised in Sec. 1.3. In Sec. 1.4, I review a specific kind of task that is representative of a major portion of MIR research, and in Sec. 1.5 I look at one implementation of it. This leads to testing a causal model in Sec. 1.6, and the risk of committing the sharpshooter fallacy, which I describe in Sec. 1.7.

1.1 The aims of MIR research

MIR research aims to connect real-world users with music and information about music, and to help users make music and information about music [3]. One cannot overstate the importance of *relevant* and *reliable* evaluation to this pursuit: [1] we depend upon it to measure the effectiveness of our algorithms and systems, compare them against the state of the art, chart the progress of our discipline, and discriminate promising directions from dead ends. The design of any machine listening system [2] involves a series of complex decisions, and so we seek the best evidence to guide this process. This motivates the largest contribution so far to evaluation methodology in MIR research: the Music Information Retrieval Exchange (MIREX) [9].

1.2 MIREX

MIREX represents a significant advance beyond the inconsistency of evaluation practices in the early years of MIR. Its guiding precepts include [8]: test collections should be of considerable size and private, if possible; evaluations should be performed by a private centralised system; formal statistical testing should be used to detect significant differences between submissions; results should be publicly archived; and MIREX is not a competition but an opportunity to exchange knowledge. MIREX is now a decade old (evaluating nearly 3000 submissions so far) and is linked to a significant amount of research [5]. However, MIREX suffers serious problems [11, 17, 20, 22, 23, 28, 34, 35]: its tasks can lack consideration of the user; its tasks can be poorly defined and contrived; its metrics can lack relevance; and its evaluations can lack validity.

1.3 The "Roadmap" for MIR [23]

MIREX has certainly helped MIR advance, but its evaluation practices must be improved. This fact is officially acknowledged in the 2013 "Roadmap for Music Information Research" [23], authored by 17 recognised MIR researchers at seven major institutions. Section 2.6 of the Roadmap identifies seven specific challenges related to evaluation that the discipline should address to ensure its continued development. We need to:

$\mathbf{R}_{\mathrm{I}}$ "Define meaningful evaluation tasks"
$\mathbf{R}_{\mathrm{II}}$ "Define meaningful evaluation methodologies"
$\mathbf{R}_{\mathrm{III}}$ "Evaluate whole MIR systems"
$\mathbf{R}_{\mathrm{IV}}$ "Promote evaluation tasks using multimodal data"
$\mathbf{R}_{\mathrm{V}}$ "Promote best practice evaluation methodology"
$\mathbf{R}_{\mathrm{VI}}$ "Implement sustainable MIR evaluation initiatives"
$\mathbf{R}_{\mathrm{VII}}$ "[Promote reproducible] MIR."

I identify R_{I} and R_{II} as the "linchpins." It is thus essential to define, "define" and "meaningful" for both "tasks" and "methodologies". For R_{I}, the Roadmap suggests a task is "meaningful" when it is "relevant" to a well-defined

[1] An "evaluation" is a protocol for testing a hypothesis or estimating a quantity. An evaluation is "relevant" if it logically addresses the investigated hypothesis or quantity. An evaluation is "reliable" if it results in a repeatable and statistically sound conclusion.

[2] A machine listening system is a fixed map from a recording universe to a semantic universe [30]. See Sec. 3.1.

© Bob L. Sturm. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Bob L. Sturm. "Revisiting Priorities: Improving MIR Evaluation Practices", 17th International Society for Music Information Retrieval Conference, 2016.

user community, and defined (or addressed) [3] "according to some agreed criteria." For R_{II}, the Roadmap suggests an evaluation methodology is "meaningful" if it creates knowledge leading to the improvement of MIR systems, and the discipline as a whole.

1.4 The "Audio Classification (Train/Test)" task

MIREX shows MIR is replete with tasks, but I will focus on one kind: "Audio Classification (Train/Test)." This task involves building systems using feature extraction algorithms, supervised machine learning algorithms, and a training dataset, and then testing with a testing dataset. [4] At its most base, the goal is to build a system that reproduces the most ground truth of a testing dataset. This task appears in over 400 publications addressing "music genre recognition" [26, 27] (not to mention work addressing "music similarity," "music mood recognition" and "autotagging" [24]), and so typifies a major portion of MIR research. Referring to the aims of MIR and the Roadmap, I ask: how does reproducing dataset ground truth provide relevant and reliable knowledge about a system, and how to improve it, for a well-defined user-community?

1.5 Three systems designed for a specific problem

Consider three systems expressly designed to address the problem intended by the BALLROOM dataset [7]: to extract and learn "repetitive rhythmic patterns" (RRPs) from recorded audio. BALLROOM has 698, 30-second monophonic music excerpts downloaded in 2004 from a commercial web resource devoted to ballroom dancing. A system solving this problem maps 30-s music recordings to several classes, e.g., Cha cha, *according to RRPs*.

Three systems trained and tested with the same BALLROOM partitioning produce the following figures of merit: system A ($\mathscr{S}_A$) reproduces 93.6% of the ground truth; $\mathscr{S}_B$, 91.4%; and $\mathscr{S}_C$, only 28.5%. Given that a random selection of each class will reproduce about 14.3% of the ground truth, two possible hypotheses are:

H1 $\mathscr{S}_A$, $\mathscr{S}_B$ are identifying RRPs in BALLROOM, and $\mathscr{S}_C$ is not identifying RRPs in BALLROOM

H2 The features used by $\mathscr{S}_A$, $\mathscr{S}_B$ are powerful for identifying RRPs in BALLROOM, but not those of $\mathscr{S}_C$.

Let us look under the hood of each system, so to speak. Each is composed of a feature extraction algorithm (mapping the audio sample domain to a feature space), and a classification algorithm (mapping the feature domain to the semantic (label) space) [30]. For $\mathscr{S}_A$, the feature is global tempo (possibly with an octave error), and the classifier is single nearest neighbour in the training dataset [29]. For $\mathscr{S}_B$, the feature is the 800-dimensional lagged autocorrelation of an energy envelope, and the classifier is a deep neural network [18]. For $\mathscr{S}_C$, the feature is *umpapa presence*, [5] and the classifier is a decision tree.

There are a few startling things. First, while $\mathscr{S}_A$ reproduces the most ground truth, it does so by using *only* tempo. Since tempo is not rhythm, $\mathscr{S}_A$ does not address the problem for which it was designed. [6] Second, $\mathscr{S}_C$ reproduces the least ground truth of the three, but is using a feature that is relevant to the problem intended by BALLROOM [7]. Its accuracy is so low because only the Waltz-labeled recordings have high umpapa presence, while all the others are in common or duple meter and have low umpapa presence. It seems then that we must doubt H1 and H2 when it comes to $\mathscr{S}_A$ and $\mathscr{S}_C$. What about $\mathscr{S}_B$?

$\mathscr{S}_B$ is using a feature that should contain information closely related to RRPs: periodicities of acoustic stresses observed over 10 second periods [18]. In fact, it is easy to visually interpret these features in terms of tempo and meter. The deep neural network in $\mathscr{S}_B$ is not so easy to interpret. However, given the impressiveness of recent results from the deep learning revolution [15], it might seem reasonable to believe $\mathscr{S}_B$ reproduces so much ground truth because it has learned to identify the high-level RRPs that characterise the rhythms of BALLROOM labels.

1.6 An intervention into a causal model

Let us claim that $\mathscr{S}_B$ reproduces BALLROOM ground truth by detecting RRP in the music recordings. We thus propose the causal model shown in Fig. 1(a). Consider an experiment in which we perform an intervention at the exogenous factor. We take each BALLROOM testing recording and find the minimum amount of pitch-preserving time-stretching [7] (thereby producing a tempo change only) for which $\mathscr{S}_B$ produces an incorrect class.

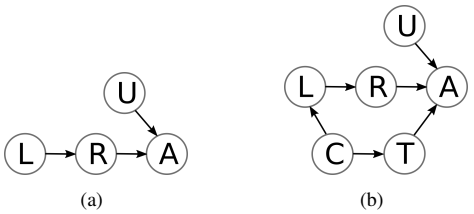


Figure 1. Two causal models relating factors: BALLROOM label (L), RRP (R), audio observation (A), competition rules (C), tempo (T), and exogenous (U).

We find [29, 31] that a mean tempo increase of 3.7 beats per minute (BPM) makes $\mathscr{S}_B$ classify all Cha-cha-labeled testing recordings as "tango." While $\mathscr{S}_B$ initially shows a very good "rumba" F-score (0.81), it no longer identifies the Rumba-labeled recordings when time-stretched by at most $\pm 3\%$. By submitting all testing recordings to a tempo change of at most $\pm 6\%$, $\mathscr{S}_B$ goes from reproducing 91.4% of the ground truth to reproducing as little as random guessing (14.3%). (By the same transformation, we can also make $\mathscr{S}_B$ reproduce all ground truth.) What is more, $\mathscr{S}_B$ assigns different labels to the same music when we change only its tempo: it classifies the same Cha cha-labeled music as "cha cha" when its tempo is 124 BPM, then "quickstep" when its tempo is 108 BPM, and so on.

[3] This is ambiguous in the Roadmap.

[4] A dataset is a sequence of observation, label pairs, $((r_i, s_i))$, where r_i is the ith observation and s_i is its ground truth.

[5] A fictional quantitative measure of the RRP, "OOM-pah-pah."

[6] In fact, Dixon et al. [7] design their systems to be tempo-invariant.

[7] We use `http://breakfastquay.com/rubberband`

These results thus cast serious doubt on H1 and H2 for $\mathscr{S}_B$. It seems that a belief in H1 and H2 is not so reasonable after all: $\mathscr{S}_A$ and $\mathscr{S}_B$ reproduce BALLROOM ground truth using a characteristic that is not rhythm. It is only $\mathscr{S}_C$ that is addressing the problem intended by BALLROOM, but it does not reproduce a large amount of ground truth simply because it is sensitive to only one kind of RRP.

1.7 On the sharpshooter fallacy

A typical response to the above is that if a system can reproduce ground truth by looking at tempo, then it should. In fact, the 2014 World Sport Dance Federation rules [8] provide strict tempo ranges for competitions featuring the dances represented by BALLROOM; and the tempi of the music in BALLROOM adhere to these ranges (with the exception of Rumba and Jive) [29, 31]. This response, however, commits the *sharpshooter fallacy*: it moves the bullseye post hoc, e.g., from "extract and learn RRPs from recorded audio" to "reproduce the ground truth." [9]

That there exists strict tempo regulations for dance competitions, and that the origin of BALLROOM comes from a commercial website selling music CDs for dance competitions, motivate the alternative causal model in Fig. 1(b). This model now shows a path from the music heard in a BALLROOM recording to its ground truth label via competition rules, which explains how $\mathscr{S}_A$ and $\mathscr{S}_B$ reproduce BALLROOM ground truth without even addressing the intended problem.

1.8 Intermediate conclusion

There are of course limitations to the above. BALLROOM is one dataset of many, and in fact could be used for a different problem than RRP. MIR tasks are broader than "Audio Classification (Train/Test)," and involve many other kinds of information than rhythm. Classification accuracy is just one measure; a confusion table could provide a more fair comparison of the three systems. I use BALLROOM and the three systems above simply because they clearly demonstrate problems that can arise even when a task and problem appear to be well-defined, and a dataset is cleanly labeled and has reputable origins. Though several systems are trained and tested in the same dataset with the express purpose of solving the same problem (as is the case for all MIREX tasks of this kind), they in fact may be solving different problems. Seeking the cause of a system's performance, e.g., through intervention experiments [25, 26, 29, 31], can then reveal a confounding of "reproduce ground truth" with, e.g., "learn to recognise rhythm." It is tempting to then move the bullseye, but doing so weakens one's contribution to the aims of MIR research. Emphasising ground truth reproduction over solving intended problems can lead to promoting non-solutions over solutions with respect to the aims of MIR research. Clearly then, MIR evaluation practices must be improved, but what should be prioritised to do so?

[8] `https://www.worlddancesport.org/Rule/Competition/General`

[9] Recall these three systems were expressly designed to address the problem intended by BALLROOM (Sec. 1.5, and described in [7]).

2. NON-CONSENSUS: OUR PRIORITIES

While problems with MIR evaluation have been known for some time, there is no consensus on what should be prioritised to solve them. I now discuss several of these.

2.1 We need to collect more data

Perhaps the most immediate answer to evaluation problems is to increase the sizes of datasets. The underlying belief is that experimental power increases with the number of observations. Along with the pursuit of model generalisation, this has motivated the creation of the Million Song Dataset [2] and AcousticBrainz [19]. The advent of crowdsourcing makes data collection seem cheap, but the actual costs can be very high. First, music recordings have many layers of intellectual property, which limit their use and distribution. This directly opposes research that is open and reproducible, and so imposes a high cost to progress. Second, and most importantly, making data bigger does not necessarily improve an experiment's power, but it *certainly* increases its cost. Even if BALLROOM had 1 billion music recordings meeting competition tempo regulations, the conclusions of Secs. 1.5-1.6 would not change. The most important question to ask then is not how to collect the most data, but how to collect and use data such that it results in the most relevant and reliable evidence possible while minimising the cost incurred.

2.2 We need to find better figures of merit

A highly discussed topic of evaluation is that of metric or measure (figure of merit, FoM). Which FoM (accuracy, F-score, AUC, etc.) gives the best indication of how well a system is addressing the intended problem? MIREX typically reports several FoM in each of its tasks since a diversity of viewpoints can inform interpretation. Still, one particular FoM can dominate a research problem, e.g., classification accuracy is the most-used FoM reported in "music genre recognition" research (appearing in over 80% of such publications [27]). This is troubling since Secs. 1.5-1.6 show that the amount of ground truth reproduced by a system could say nothing relevant or reliable about its success for some problem thought well-posed. The choice of FoM is certainly important, but the most important question to ask before selecting an FoM is how to measure its relevance and reliability, and how to compare it in meaningful ways, with respect to the intended problem. Recent work is addressing this important question, e.g., [6, 11, 16].

2.3 We need to perform more formal statistical testing

Some have argued that MIR research should adopt rigorous statistical procedures [10,34]. Null hypothesis significance testing provides a remarkable set of useful tools, despite the problems that come with their interpretation [4,14]. For instance, the probability that $\mathscr{S}_B$ in Sec. 1.5 reproduces its 91.4% of BALLROOM ground truth given it is actually selecting randomly is $p < 10^{-138}$ (by a binomial test). Though one may safely reject this hypothesis, it still gives no reason to claim $\mathscr{S}_B$ is identifying RRP. Sec. 1.6 shows $\mathscr{S}_B$ to be exploiting a third way to reproduce BALLROOM

ground truth. No statistical test will improve the relevance and reliability of the measurements of the three systems in Sec. 1.5 with respect to the hypotheses posed. A different kind of evaluation must be employed. While statistical testing is useful, the most important questions to ask first are: 1) whether the evidence produced by an evaluation will be relevant and reliable at all; and 2) what statistical test is relevant to and permitted by an experiment [1, 12].

2.4 We need to use more cross-validation

In the directions of using data in smarter ways and statistical hypothesis testing, cross-validation (CV) is a testing protocol that is standard in machine learning [13]. CV holds constant a learning algorithm but changes training and testing datasets multiple times, both of which are culled from a larger dataset. Many variants exist, but the main motivations are the same: to simulate having more data than one actually has, and to avoid overfitting models. CV produces point estimates of the expected generalisation of a learning algorithm [13], but its relevance and reliability for gauging the success of a given system for solving an intended problem can be unclear. Returning to the three systems in Sec. 1.5, CV in BALLROOM will produce more estimates of the expected generalisation of the learning algorithms, but that does not then make the results relevant to the hypotheses posed. The most important question to ask then is how to design a testing protocol that can produce relevant and reliable evidence for the hypotheses under investigation.

2.5 We need to develop central, transparent and immediate evaluation

The nature of a computerised research discipline is such that one can train and test thousands of system variants on millions of observations and produce quantitative results within a short time scale. MIREX provides one vehicle, but it occurs only once a year, and has problems of its own (Sec. 1.2). This motivates commendable efforts, such as the Networked Environment for Music Analysis [36], and `mir_eval` [20]. Their aim is to increase transparency, standardise evaluation, reduce delay, mitigate legal jeopardy, and facilitate progress in MIR. Concerns of the transparency and immediacy of an evaluation, however, are premature before designing it to be as relevant and reliable as possible at the least cost.

2.6 Intermediate conclusion

The overriding priority is not to collect more data, but to develop ways to collect and use data in provably better ways. The overriding priority is not to find better FoM, but to develop ways to judge the relevance of any FoM, and to make meaningful comparisons with it. The overriding priority is not to perform more formal statistical testing, to use CV, or facilitate transparency and immediacy in evaluation, but to develop ways of producing relevant evidence while satisfying requirements of reliability and cost. This leads me to propose a principal priority for improving evaluation practices in MIR.

3. THE PRINCIPAL PRIORITY IN EVALUATION

The *principal priority* is to develop a formal framework of evaluation that facilitates a meaningful evaluation methodology for any problem that will result in relevant and reliable evidence of the effectiveness of our algorithms and systems, facilitate comparisons with the state of the art, chart the progress of the discipline, and discriminate promising directions from dead ends, all with respect to the aims of MIR research. I propose that this can be accomplished by leveraging the established design and analysis of experiments (DOE) in tandem with an effort to reign in the ambiguity of MIR problems and tasks. This will then address the shortcomings of MIREX (Sec. 1.2), position MIR to meet the Roadmap challenges (Sec. 1.3), and enable a new and progressive research pipeline.

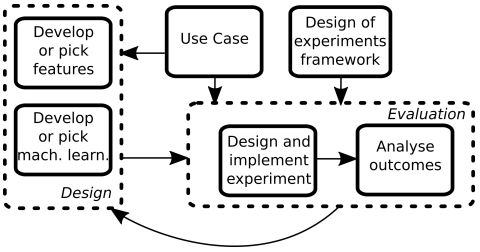


Figure 2. This new research pipeline involves a use case, a DOE framework, and feedback to improve system design.

Figure 2 illustrates a new way to engineer MIR systems and their component technologies. Three central components are the use case, a formal design of experiments (DOE) framework, and feedback from the evaluation to system design. A *use case* is a formal expression of the problem a system is to address. The *DOE framework* provides the theoretical underpinning for designing, implementing and analysing the most relevant and reliable evaluation of a system with respect to a use case at the least possible cost. These two components feed into realising an evaluation of a specific MIR system, itself built with reference to the use case. The evidence produced by the evaluation thus leads to improving the design of the MIR system. The use case together with the DOE framework temper linchpins R_I and R_{II}. With these firmly established, the rest of the Roadmap evaluation challenges can be accomplished.

3.1 On the use case

I define a use case in [30] as a means for mitigating ambiguity in research, and thus tempering linchpin R_I. For instance, nearly all published work on the problem of "music genre recognition" does not explicitly define the problem, instead posing it as reproducing the ground truth of a given dataset. As a result, many hundreds of publications have unknown relevance to the aims of MIR (Sec. 1.1), even when they use the same dataset [26].

A *use case* is defined as a tuple of four formal elements: the music universe (Ω), the music recording universe ($\mathcal{R}_\Omega$), the description universe ($\mathcal{S}_{\mathcal{V},A}$), and a set of success criteria. This retains a distinction between the intangible Ω and

ID	*Treatments* ($\mathcal{T}$)	*Experimental unit*	*Observational unit* ($\omega \in \Omega$)	*Treatment structure*	*Plot structure*	*Response*	*Response model*
a	Amounts of compost & water	tomato plant in a greenhouse pot	tomato plant in a greenhouse pot	all combinations of two factors	blocks	tomato yield (grams)	simple textbook
b	New feed, old feed	pen	calf	new treatment and control	unstructured	weight (kg)	simple textbook
c	Local or remote learning	students in DOE 101 classroom-year	student	unstructured	blocks	test score (%)	fixed effects
d	Four wines	judge	judge-tasting	unstructured	unstructured	score $\{1, ..., 5\}$	simple textbook

Table 1. Examples of the various components of experiments. (a): estimating the relationship between tomato yield, and the amount of water and compost applied to a tomato plant in a greenhouse pot. (b): testing for a significant difference between new and old feed in the weight gain of a calf (several calves to a pen, feed applied to whole pen). (c): testing for a significant difference between students learning DOE locally or remotely (students are or are not math majors, thus defining two blocks, motivating a fixed effects response model). (d): testing for a significant difference in wine quality.

the tangible $\mathcal{R}_\Omega$ — elements of which are fed to a recorded music description system (a map, $\mathscr{S} : \mathcal{R}_\Omega \to \mathcal{S}_{\mathcal{V},A}$). $\mathcal{S}_{\mathcal{V},A}$ is a set of elements assembled in a meaningful way for a user. The success criteria embody the requirements of a user for mapping from Ω and/or $\mathcal{R}_\Omega$ to $\mathcal{S}_{\mathcal{V},A}$.

To provide illustration, let us define two use cases. Define Ω as all music meeting specific tempo and stylistic regulations with respect to the labels in BALLROOM. Define $\mathcal{R}_\Omega$ as the set of all possible 30-s, monophonic recording excerpts of the music in Ω sampled at 22050 Hz. Define $\mathcal{S}_{\mathcal{V},A}$ as the set of tokens, {"Cha cha", "Jive", "Quickstep", "Rumba", "Tango", "Waltz"}. Define the success criteria as: "the amount of ground truth reproduced is inconsistent with random selection." Provided BALLROOM is a sample of $\mathcal{R}_\Omega \times \mathcal{S}_{\mathcal{V},A}$, it is relevant to this use case; and depending on its size relative to the variability of the population (which is predicted, e.g., using expert elicitation), a measurement of the ground truth reproduced by a system tested in BALLROOM could then provide reliable evidence of its success.

A different use case is possible. Define Ω, $\mathcal{R}_\Omega$ and $\mathcal{S}_{\mathcal{V},A}$ as above, but define the success criteria as: "the amount of ground truth reproduced is inconsistent with random selection, and *independent of tempo*." Again, provided BALLROOM is a sample of $\mathcal{R}_\Omega \times \mathcal{S}_{\mathcal{V},A}$, it is relevant to this use case. However, a measurement of the amount of ground truth reproduced by a system tested in BALLROOM is not relevant to the use case *because* it does not control for the restriction imposed in the success criteria. A different evaluation must be designed.

The use case proposed in [30] is not the only way or the best way to mitigate ambiguity in MIR research, but I claim that it is one way by which a research problem and task can be defined with clarity, and which thereby can aid with the design and evaluation of MIR systems.

3.2 On the formal design of experiments

The aim of DOE is to help one produce the most reliable evidence at the least cost. DOE is an area of statistics that has become essential to progressive science and profitable industry, from biology and genetics to medicine and agriculture [1]. (In fact, it arose from agriculture in the early 20th century.) Hence, I claim that it is reasonable to argue that DOE can help build a formal framework of evaluation that can reliably guide the engineering of systems to address the aims of MIR (Sec. 1.1).

The design of an experiment entails performing several essential and non-trivial steps. In the terminology of DOE [1], this includes identifying treatments, experimental and observational units, identifying structures in the treatments and plots, creating the design, and specifying the response model, all with respect to the hypothesis or quantity under investigation. Below are some definitions of these components. Table 1 provides examples.

Treatments Descriptions of what is applied to an experimental unit, indexed by $\mathcal{T} = \{1, \ldots, t\}$.

Experimental unit The smallest unit to which a treatment is applied.

Observational unit (plot) The smallest unit on which a response is measured. The set of N plots is indexed by $\Omega := \{\omega : \omega \in \{1, \ldots, N\}\}$.

Experimental design A map $T : \Omega \to \mathcal{T}$.

Plot/Treatment structure Meaningful ways of dividing up the plots/treatments.

Response model An assumed mathematical relation between the response and the treatment parameter.

Treatment Parameter The (latent) contribution of the treatment to the measured response.

Fundamental questions that DOE answers are: for my t treatments, how large should N be to reach my required experimental power and not exceed my resources? How should I collect the plots? How should I map the plots to the treatments? An essential part of answering these is "expert elicitation," whereby knowledge about the plots and treatments is collected from someone familiar with them. This informs the design, response model, and subsequent analysis. For example, it is important to know in experiments (a) and (c) in Table 1 if the plots have structure, e.g., positions of pots in greenhouse get variable sunlight; and students in the course are math majors or non-math majors. Otherwise, if the amount of sunlight correlates with the amount of water and compost applied to a plant, or if most students that take the course remotely are math majors, then one might conclude that compost and water have a negative effect on yield (confounding of sunlight and treatment), or remote learning is better than local learning (confounding of student background and learning method).

At first it seems standard MIR tasks and evaluation need only be "translated" into the language of DOE, and then existing statistical machinery be deployed [32]. This translation is not so immediate, however. Consider the evaluation performed in [33]: 100 repetitions of stratified 10-fold CV (10fCV) in a specific dataset sampled from some $\mathcal{R}_\Omega \times \mathcal{S}_{\mathcal{V},A}$. In each repetition, several systems are trained and tested, the mean amount of ground truth reproduced by the resulting systems is measured, the mean and standard deviation of the 100 repetitions are reported, and conclusions are made. This appears to be a factorial design, crossing F (feature extraction method) and M (supervised learning method). The treatments then appear to be all levels of $F \wedge M$. The experimental and observational unit then is a complete 10fCV, of which there are $N = 100|F||M|$, i.e., each level in $F \wedge M$ treats 100 10fCV plots. Finally, the response is the proportion of ground truth reproduced.

This scenario appears similar to testing for a significant difference between local and remote learning in Table 1(c), except there are some important differences. First, any pair of systems in each level of $F \wedge M$ in any repetition of 10fCV share 80% of the same training data. Hence, the 10 systems produced in each level of $F \wedge M$ in any repetition of 10fCV are not independent. Second, all repetitions of 10fCV at a level of $F \wedge M$ produce measurements that come from the same data. Hence, the 100 plots are not independent. Third, the systems themselves are generated from training data used to test other systems produced at the same level. Considering the scenario in Table 1(c), this would be like tailoring the implementations of local and remote classes to some of the students in them. This means the realisations of the treatments come from material that they in turn treat, which introduces a non-trivial dependency between treatments and plots. Finally, there is unacknowledged structure in the particular dataset used in [33], which can bias the response significantly [26].

All of the above and more [10] means that the responses measured in this experiment should be modelled in a more complex way than the simple textbook model [1] — which assumes a normal distribution having a variance that decreases with the number of measurements, and a mean that is the treatment parameter (in this case, the expected generalisation of a level in $F \wedge M$ in $\mathcal{R}_\Omega \times \mathcal{S}_{\mathcal{V},A}$). What can be concluded from the kind of experiments in [33] remains to be seen, but it is clear that not much weight should be given to conclusions drawn from the simple textbook model [32].

3.3 On the feedback

The initial evaluation of the three systems in Sec. 1.5 is of little use for improving them with respect to the problem they are designed to address. It even provides misleading information about which is best or worst at addressing that problem. The knowledge produced by the evaluation is thus suspicious. Instead, my system analysis, along with the intervention experiment in Sec. 1.6, provide useful knowledge for improving the systems, as well as evaluation using BALLROOM. Our work in [21] provides another example of this. According to the linchpin challenge $\mathrm{R_{II}}$ in Sec. 1.3, and its discussion in the Roadmap, I claim that system analysis along with the intervention experiment — designed to explain why a system reproduces an amount of ground truth inconsistent with random selection — can lead to real and useful knowledge for improving MIR systems and evaluation practices.

[10] There are other major issues that contribute complications, e.g., the meaning of $\mathcal{R}_\Omega \times \mathcal{S}_{\mathcal{V},A}$, whether "ground truth" is a rational concept, and the problem of "curation" in assembling of a music dataset.

4. CONCLUSION

The MIR discipline has reached a level of maturity such that its impact is undeniable [3,5], and its leaders recognise contemporary needs for targeted development in specific directions [23]. In this position paper, I address the direction of improving MIR evaluation — specifically the linchpin challenges $\mathrm{R_I}$ and $\mathrm{R_{II}}$ (Sec. 1.3) — and revisit several priorities for improving evaluation. I argue that these only treat the symptoms of the problem and not its cause: MIR lacks a formal evaluation framework relevant to its aims. I argue that addressing this cause is the principal priority. I propose that this can be addressed by leveraging established DOE along with an effort to mitigate ambiguity in research. However, this is not as straight-forward as I initially envisioned [26, 32]. To develop and integrate such a framework into the MIR research pipeline will require far more effort than is currently being devoted to it. It will require the focus of a multidisciplinary team of specialists for many years. Toward this end, I hope this position paper persuades some to participate in solving the core problem.

4.1 Acknowledgments

This paper comes from feedback to many grant proposals, paper submissions, poster discussions, public seminars, and private conversations. I give considerable thanks to all who have taken the time to talk, and to: H. Maruri-Aguilar, B. Parker, F. Rodríguez-Algarra and H. Grossmann.

5. REFERENCES

[1] R. A. Bailey. *Design of comparative experiments*. Cambridge University Press, 2008.

[2] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proc. ISMIR*, pages 591–596, 2011.

[3] M. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proc. IEEE*, 96(4):668–696, Apr. 2008.

[4] D. Colquhoun. An investigation of the false discovery rate and the misinterpretation of p-values. *Royal Society Open Science*, 1(3), 11 2014.

[5] S. J. Cunningham, D. Bainbridge, and J. S. Downie. The impact of MIREX on scholarly research. In *Proc. ISMIR*, pages 259–264, 2012.

[6] M. Davies and S. Böck. Evaluating the evaluation measures for beat tracking. In *Proc. ISMIR*, pages 637–642, 2014.

[7] S. Dixon, F. Gouyon, and G. Widmer. Towards characterisation of music via rhythmic patterns. In *Proc. ISMIR*, pages 509–517, 2004.

[8] J. Downie, A. Ehmann, M. Bay, and M. Jones. The music information retrieval evaluation exchange: Some observations and insights. In *Advances in Music Information Retrieval*, pages 93–115. Springer, 2010.

[9] J. S. Downie. The scientific evaluation of music information retrieval systems: Foundations and future. *Computer Music Journal*, 28(2):12–23, 2004.

[10] A. Flexer. Statistical evaluation of music information retrieval experiments. *J. New Music Research*, 35(2):113–120, 2006.

[11] A. Flexer. On inter-rater agreement in audio music similarity. In *Proc. ISMIR*, pages 245–250, 2014.

[12] D. J. Hand. Deconstructing statistical questions. *J. Royal Statist. Soc. A (Statistics in Society)*, 157(3):317–356, 1994.

[13] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, 2 edition, 2009.

[14] John P. A. Ioannidis. Why most published research findings are false. *PLOS Medicine*, 2007.

[15] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[16] O. Nieto, M. M. Farbood, T. Jehan, and J. P. Bello. Perceptual analysis of the f-measure for evaluating section boundaries in music. In *Proc. ISMIR*, pages 265–270, 2014.

[17] G. Peeters, J. Urbano, and G. J. F. Jones. Notes from the ISMIR 2012 late-breaking session on evaluation in music information retrieval. In *Proc. ISMIR*, 2012.

[18] A. Pikrakis. A deep learning approach to rhythm modeling with applications. In *Proc. Int. Workshop Machine Learning and Music*, 2013.

[19] A. Porter, D. Bogdanov, R. Kaye, R. Tsukanov, and X. Serra. Acousticbrainz: A community platform for gathering music information obtained from audio. In *Proc. ISMIR*, pages 786–792, 2015.

[20] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis. mir eval: A transparent implementation of common MIR metrics. In *Proc. ISMIR*, pages 367–372, 2014.

[21] F. Rodríguez-Algarra, B. L. Sturm, and H. Maruri-Aguilar. Analysing scattering-based music content analysis systems: Where's the music? In *Proc. ISMIR*, 2016.

[22] M. Schedl, A. Flexer, and J. Urbano. The neglected user in music information retrieval research. *J. Intell. Info. Systems*, 41(3):523–539, 2013.

[23] X. Serra, M. Magas, E. Benetos, M. Chudy, S. Dixon, A. Flexer, E. Gómez, F. Gouyon, P. Herrera, S. Jordà, O. Paytuvi, G. Peeters, J. Schlüter, H. Vinet, and G. Widmer. *Roadmap for Music Information ReSearch*. Creative Commons, 2013.

[24] B. L. Sturm. Evaluating music emotion recognition: Lessons from music genre recognition? In *Proc. ICME*, 2013.

[25] B. L. Sturm. A simple method to determine if a music information retrieval system is a "horse". *IEEE Trans. Multimedia*, 16(6):1636–1644, 2014.

[26] B. L. Sturm. The state of the art ten years after a state of the art: Future research in music information retrieval. *J. New Music Research*, 43(2):147–172, 2014.

[27] B. L. Sturm. A survey of evaluation in music genre recognition. In A. Nürnberger, S. Stober, B. Larsen, and M. Detyniecki, editors, *Adaptive Multimedia Retrieval: Semantics, Context, and Adaptation*, volume LNCS 8382, pages 29–66, Oct. 2014.

[28] B. L. Sturm. Faults in the latin music database and with its use. In *Proc. ISMIR (Late breaking demo)*, 2015.

[29] B. L. Sturm. The "horse" inside: Seeking causes of the behaviours of music content analysis systems. *ACM Computers in Entertainment (accepted)*, 2015.

[30] B. L. Sturm, R. Bardeli, T. Langlois, and V. Emiya. Formalizing the problem of music description. In *Proc. ISMIR*, pages 89–94, 2014.

[31] B. L. Sturm, C. Kereliuk, and A. Pikrakis. A closer look at deep learning neural networks with low-level spectral periodicity features. In *Proc. Int. Workshop on Cognitive Info. Process.*, pages 1–6, 2014.

[32] B. L. Sturm, H. Maruri-Aguilar, B. Parker, and H. Grossmann. The scientific evaluation of music content analysis systems: Valid empirical foundations for future real-world impact. In *Proc. ICML Machine Learning for Music Discovery Workshop*, 2015.

[33] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. Speech Audio Process.*, 10(5):293–302, July 2002.

[34] J. Urbano, B. McFee, J. S. Downie, and M. Schedl. How significant is statistically significant? the case of audio music similarity and retrieval. In *Proc. ISMIR*, pages 181–186, 2012.

[35] J. Urbano, M. Schedl, and X. Serra. Evaluation in music information retrieval. *J. Intell. Info. Systems*, 41(3):345–369, Dec. 2013.

[36] K. West, A. Kumar, A. Shirk, G. Zhu, J. S. Downie, A. Ehmann, and M. Bay. The networked environment for music analysis (nema). In *World Congress on Services*, pages 314–317, July 2010.

SIMULTANEOUS SEPARATION AND SEGMENTATION IN LAYERED MUSIC

Prem Seetharaman
Northwestern University
prem@u.northwestern.edu

Bryan Pardo
Northwestern University
pardo@northwestern.edu

ABSTRACT

In many pieces of music, the composer signals how individual sonic elements (samples, loops, the trumpet section) should be grouped by introducing sources or groups in a layered manner. We propose to discover and leverage the layering structure and use it for both structural segmentation and source separation. We use reconstruction error from non-negative matrix factorization (NMF) to guide structure discovery. Reconstruction error spikes at moments of significant sonic change. This guides segmentation and also lets us group basis sets for NMF. The number of sources, the types of sources, and when the sources are active are not known in advance. The only information is a specific type of layering structure. There is no separate training phase to learn a good basis set. No prior seeding of the NMF matrices is required. Unlike standard approaches to NMF there is no need for a post-processor to partition the learned basis functions by group. Source groups are learned automatically from the data. We evaluate our method on mixtures consisting of looping source groups. This separation approach outperforms a standard clustering NMF source separation approach on such mixtures. We find our segmentation approach is competitive with state-of-the-art segmentation methods on this dataset.

1. INTRODUCTION

Audio source separation, an open problem in signal processing, is the act of isolating sound producing sources (or groups of sources) in an audio scene. Examples include isolating a single person's voice from a crowd of speakers, the saxophone section from a recording of a jazz big band, or the drums from a musical recording [13].

A system that can understand and separate musical signals into meaningful constituent parts (e.g. melody, backing chords, percussion) would have many useful applications in music information retrieval and signal processing. These include melody transcription [18], audio remixing [28], karaoke [21], and instrument identification [8].

Many approaches have been taken to audio source separation, some of which take into account salient aspects of musical structure, such as musical scores, or pitch (see Section 2). Few algorithms have explicitly learned musical structure from the audio recording (using no prior learning and no musical score) and used it to guide source discovery and separation. Our approach is designed to leverage compositional structures that introduce important musical elements one by one in *layers*. In our approach, separation alternates with segmentation, simultaneously discovering the layering structure and the functional groupings of sounds.

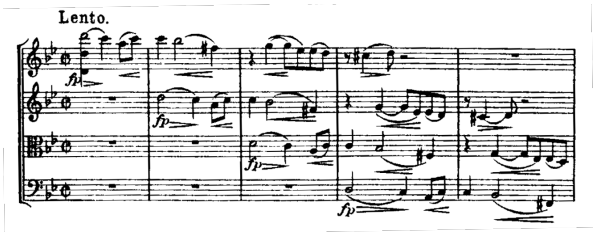


Figure 1. An exemplar layering structure in classical music - *String Quartet No. 1, Op. 27, Mvt IV, Measures 1-5.* by *Edvard Grieg*. The instruments enter one at a time in a layering structure, guiding the ear to both the content and the different sources.

In a layered composition, the composer signals how individual sound sources (clarinet, cello) or sonic elements (samples, loops, sets of instruments) should be grouped. For example, often a song will start by introducing sources individually (e.g. drums, then guitar, then vocals, etc) or in groups (the trumpet section). Similarly, in many songs, there will be a "breakdown", where most of the mixture is stripped away, and built back up one element at a time. In this way, the composer communicates to the listener the functional musical groups (where each group may consist of more than one source) in the mixture. This layering structure is widely found in modern music, especially in the pop and electronic genres (Figure 2), as well as classical works (Figure 1).

We propose a separation approach that engages with the composer's intent, as expressed in a layered musical structure, and separates the audio scene using discovered functional elements. This approach links the learning of the segmentation of music to source separation.

We identify the layering structure in an unsupervised manner. We use reconstruction error from non-negative matrix factorization (NMF) to guide structure discovery.

© Prem Seetharaman, Bryan Pardo. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Prem Seetharaman, Bryan Pardo. "Simultaneous separation and segmentation in layered music", 17th International Society for Music Information Retrieval Conference, 2016.

Reconstruction error spikes at moments of significant sonic change. This guides segmentation and also lets us know where to learn a new basis set.

Our approach assumes nothing beyond a layering structure. The number of sources, the types of sources, and when the sources are active are not known *a priori*. In parallel with discovering the musical elements, the algorithm temporally segments the original music mixture at moments of significant change. There is no separate training phase to learn a good basis set. No prior seeding of the NMF matrices is required. Unlike standard NMF there is no need for a post-processor that groups the learned basis functions by source or element [9] [25]. Groupings are learned automatically from the data by leveraging information the composer put there for a listener to find.

Our system produces two kinds of output: a temporal segmentation of the original audio at points of significant change, and a separation of the audio into the constituent sonic elements that were introduced at these points of change. These elements may be individual sources, or may be groups (eg. stems, orchestra sections).

We test our method on a dataset of music built from commercial musical loops, which are placed in a layering structure. We evaluate the algorithm based on separation quality, as well as segmentation accuracy. We compare our source separation method to standard NMF, paired with a post processer that clusters the learned basis set into groups in a standard way. We compare our segmentation method to the algorithms included in the Musical Structure Analysis Framework (MSAF) [16].

The structure of this paper is as follows. First, we describe related work in audio source separation and music segmentation. Then, we give an overview of our proposed separation/segmentation method, illustrated with a real-world example. We then evaluate our method on our dataset. Finally, we consider future work and conclude.

2. RELATED WORK

2.1 Music segmentation

A good music segmentation reports perceptually relevant structural temporal boundaries in a piece of music (e.g. verse, chorus, bridge, an instrument change, a new source entering the mixture).

A standard approach for music segmentation is to leverage the self-similarity matrix [7]. A novelty curve is extracted along the diagonal of the matrix using a checkerboard kernel. Peak picking on this novelty curve results in a music segmentation. The relevance of this segmentation is tied to the relevance of the similarity measure.

[12] describes a method of segmenting music where frames of audio are labeled as belonging to different states in a hidden Markov model, according to a hierarchical labeling of spectral features. [10] takes the self-similarity matrix and uses NMF to find repeating patterns/clusters within it. These patterns are then used to segment the audio. [17] expands on this work by adding a convex constraint to NMF.

[22] infers the structural properties of music based on structure features that capture both local and global properties of a time series, with similarity features.

The most similar work to ours is [27], which uses shift invariant probabilistic latent component analysis to extract musical riffs and repeated patterns from a piece of music. The activation of these recurring temporal patterns is used to then segment the audio. Our approach takes into account temporal groupings when finding these patterns, whereas their approach does not.

Our proposed method uses the reconstruction error of a source model over time in a musical signal in order to find structural boundaries. We explicitly connect the problem of music segmentation with the problem of audio source separation and provide an alternative to existing approaches to finding points of significant change from the audio.

2.2 Source separation

There are several source separation methods that leverage high-level musical information in order to perform audio source separation.

Separation from repeating patterns: REPET [21] separates the repeating background structure (e.g. bass, backing chords, rhythm from the guitar, and drums in a band) from a non-repeating foreground (e.g. lead singer with a melody) by detecting a periodic pattern (e.g. a sequence of chords that repeats every four bars). While REPET models the repeating background structure as a whole, our proposed method models the individual musical elements implicit in the composer's presentation of the material and does not require a fixed periodic pattern.

In [20], source models are built using a similarity matrix. This work looks for similar time frames anywhere in the signal, using a similarity measure (cosine similarity) that does not take musical grouping or temporal structure into account. Our method leverages the temporal groupings created by the composer's layering of sonic elements.

Informed separation: [4] incorporates outside information about the musical signal. A musical score gives information about the pitch and timing of events in the audio and is commonly used for informed musical separation. First, [4] finds an alignment between the low-level audio signal and the high-level musical score (in MIDI form). The pitch and timing of the event are then used to perform audio separation. These score-informed approaches are elaborated on in [6]. Our approach, does not require a score. Musical elements are discovered, modelled, and separated from the mixture using only the mixture, itself.

Non-negative matrix factorization (NMF). Our work uses NMF which was first proposed for audio source separation in [24]. Probabilistic latent component analysis (PLCA) can be seen as a probabilistic formulation of NMF, and is also used for source separation [23].

NMF finds a factorization of an input matrix $\mathbf{X}$ (the spectrogram) into two matrices, often referred to as spectral templates $\mathbf{W}$ and activations $\mathbf{H}$. Straightforward NMF has two weaknesses when used for source separation that will be elaborated on in Section 3: (1) there is no guar-

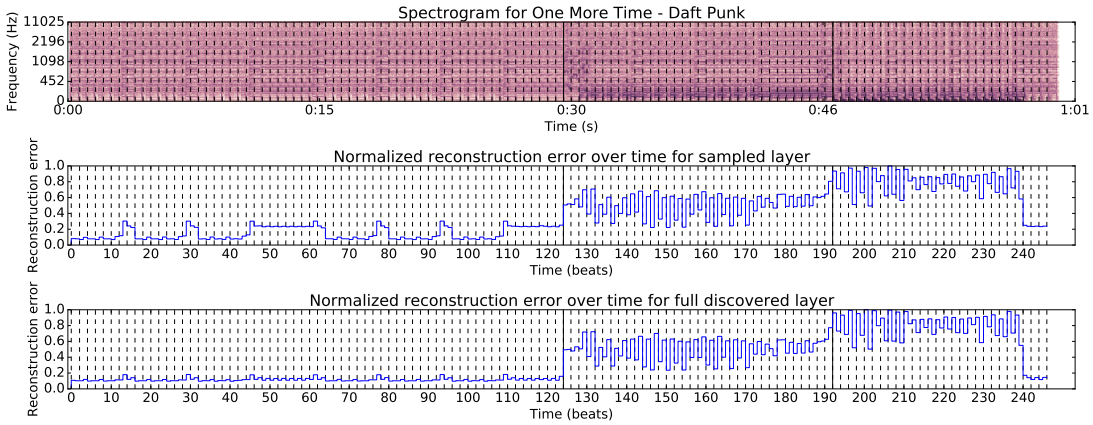


Figure 2. The top graph shows the spectrogram of 0:00 to 1:01 of *One More Time*, by *Daft Punk*. Ground truth segmentation is shown by the solid vertical black lines, where each line signals a new source starting. The middle graph shows the behavior of the reconstruction error of a sampled source layer over time ($\mathbf{e}$). When new layers begin, reconstruction error noticeably spikes and changes behavior. The bottom graph shows the reconstruction error over time for a full model of the first layer. Beats are shown by the vertical dashed black lines.

antee that an individual template (a column of $\mathbf{W}$) corresponds to only one source and (2) spectral templates are not grouped by source. Until one knows which templates correspond to a particular source or element of interest, one cannot separate out that element from the audio.

One may solve these problems by using prior training data to learn templates, or meta-data, such as musical scores [6], to seed matrices with approximately-correct templates and activations. User guidance to select the portions of the audio to learn from has also been used [2].

To group spectral templates by source without user guidance, researchers typically apply timbre-based clustering [9] [25]. This does not consider temporal grouping of sources. There are many cases where sound sources with dissimilar spectra (e.g. a high squeak and a tom drum, as in *Working in a Coal Mine* by DEVO) are temporally grouped as a single functional element by the composer. Such elements will not be grouped together with timbre-based clustering.

A non-negative Hidden Markov model (NHMM) [15] has been used to separate individual spoken voices from mixtures. Here, multiple sets of spectral templates are learned from prior training data and the system dynamically switches between template sets based on the estimated current state in the NHMM Markov model. A similar idea is exploited in [3], where a classification system is employed to determine whether a spectral frame is described by a learned dictionary for speech.

Our approach leverages temporal grouping created by composers in layered music. This lets us appropriately learn and group spectral templates without the need for prior training, user input or extra information from a musical score, or post-processing.

3. NON-NEGATIVE MATRIX FACTORIZATION

We now provide a brief overview of non-negative matrix factorization (NMF). NMF is a method to factorize a non-negative matrix $\mathbf{X}$ as the product of two matrices $\mathbf{W}$ and $\mathbf{H}$. In audio source separation, $\mathbf{X}$ is the power spectrogram of the audio signal, which is given as input. $\mathbf{W}$ is interpreted as a set of spectral templates (e.g. individual notes, the spectrum of a snare hit, etc.). $\mathbf{H}$ is interpreted as an activation matrix indicating when the spectral templates of $\mathbf{W}$ are active in the mixture. The goal is to learn this dictionary of spectral templates and activation functions. To find $\mathbf{W}$ and $\mathbf{H}$, some initial pair of $\mathbf{W}$ and $\mathbf{H}$ are created with (possibly random) initial values. Then, a gradient descent algorithm is employed [11] to update $\mathbf{W}$ and $\mathbf{H}$ at each step, using an objective function such as:

$$argmin_{\mathbf{W},\mathbf{H}}||\mathbf{X} - \mathbf{WH}||_F^2 \quad (1)$$

where $||\cdot||_F^2$ refers to the Frobenius norm. Once the difference between $\mathbf{WH}$ and $\mathbf{X}$ falls below an error tolerance, the factorization is complete.

There are typically many approximate solutions that fall below any given error bound. If one varies the initial $\mathbf{W}$ and $\mathbf{H}$ and restarts, a different decomposition is likely to occur. Many of these will not have the property that each spectral template (each column of $\mathbf{W}$) represents exactly one element of interest. For example, it is common for a single spectral template to contain audio from two or more elements of interest (e.g. a mixture of piano and voice in one template). Since these templates are the atomic units of separation with NMF, mixed templates preclude successful source separation. Therefore, something must be done to ensure that, after gradient descent is complete, each spectral template belongs to precisely one group or source of interest. An additional issue is that, to perform meaningful source separation, one must partition these spectral templates into groups of interest for separation. For example, if the goal is to separate piano from drums in a mixture of piano and drums, all the templates modeling the drums should be grouped together.

One can solve these problems by using prior training data, running the algorithm on audio containing an isolated element of interest to learn a restricted set of $\mathbf{W}$. One can repeat this for multiple elements of interest to separate audio from a mixture using these prior learned templates. This avoids the issues caused from learning the spectral templates directly from the mixture: one template having portions of two sources, and not knowing which templates belong to the same musical element. One may also use prior knowledge (e.g. a musical score) to seed the $\mathbf{W}$ and $\mathbf{H}$ matrices with values close to the desired final goal. We propose an alternative way of grouping spectral templates that does not require prior seeding of matrices [6] or user segmentation of audio to learn the basis set for each desired group [2], nor post-processing to cluster templates.

4. PROPOSED APPROACH

Our approach has four stages: estimation, segmentation, modeling, and separation. We cycle through these four stages in that order until all elements and structure have been found.

Estimation: We assume the composer is applying the compositional technique of layering. This means that estimating the source model from the first few audio frames will give us an initial model of the first layer present in the recording. Note that in our implementation, we beat track the audio [14] [5]. Beat tracking reduces the search space for a plausible segmentation, but is not integral to our approach.

We use the frames from the first four beats to learn the initial spectral dictionary. Consider two time segments in the audio, with i, j and k as temporal boundaries: $\mathbf{X} = [\mathbf{X}_{i:j-1}, \mathbf{X}_{j:k}]$. To build this model, we use NMF on a segment of $\mathbf{X_{i:j-1}}$ to find spectral templates $\mathbf{W_{est}}$.

Segmentation: Once an estimated dictionary $\mathbf{W_{est}}$ is found, we measure how well it models the mixture over time. Keeping $\mathbf{W_{est}}$ fixed, learn the activation matrix $\mathbf{H}$ for the second portion. The reconstruction error for this is:

$$error(\mathbf{W_{est}H}, \mathbf{X_{j:k}}) = ||\mathbf{X_{j:k}} - \mathbf{W_{est}H}||_F^2 \quad (2)$$

Equation 2 measures how well the templates in $\mathbf{W_{est}}$ model the input $\mathbf{X}$. For example, assume $\mathbf{W_{est}}$ was constructed a spectrogram of snare drum hits in segment $\mathbf{X_{i:j-1}}$. If a guitar and bass are added to the mixture somewhere in the range $j:k$,then the reconstruction error on $\mathbf{X_{j:k}}$ will be greater than the reconstruction error on $\mathbf{X_{i:j-1}}$. We use reconstruction error as a signal for segmentation. We slide the boundaries j, k over the the mixture, and calculate error for each of these time segments, as shown in Figure 2. This gives us $\mathbf{e}$, a vector of reconstruction errors for each time segment.

In the middle graph in Figure 2, we show reconstruction error over time, quantized at the beat level. Reconstruction error spikes on beats where the audio contains new sounds not modeled in $\mathbf{W_{est}}$. As layers are introduced by the artist, reconstruction error of the initial model rises considerably. Identifying sections of significant change in reconstruction error gives a segmentation on the music. In Figure 2, the segmentation is shown by solid vertical lines. At each solid line, a new layer is introduced into the mixture. Our method for identifying these sections uses a moving median absolute deviation, and is detailed in Algorithm 1.

Algorithm 1 Method for finding level changes in reconstruction error over time, where $\odot$ is the element-wise product. $\mathbf{e}$ is a vector where $\mathbf{e}(t)$ is the reconstruction error for $\mathbf{W_{est}}$ at time step t. lag is the size of a smoothing window for $\mathbf{e}$. p and q affect how sensitive the algorithm is when finding boundary points. We use $lag = 16$, $p = 5.5$ and $q = .25$ in our implementation. These values were found when training on a different dataset, containing 5 mixtures, than the one in Section 5.

```
lag, p, q ← initialize, tunable parameters
e ← reconstruction error over time for W_est
e ← (e ⊙ e)                          ▷ Element-wise product
d ← max(Δe/Δt)
for i from lag to length(e) do       ▷ Indexing e
    window ← e_{i−lag:i−1}
    m ← median(abs(window - median(window))
    if abs(e_i − median(window)) ≥ p ∗ m then
        if abs(e_i − e_{i−1}) ≥ q ∗ d then
            return i                 ▷ Boundary frame in X
        end if
    end if
end for
return length(e)                     ▷ Last frame in X
```

Modeling: once the segmentation is found, we learn a model using NMF on the entire first segment. This gives us $\mathbf{W_{full}}$, which is different from $\mathbf{W_{est}}$, which was learned from just first four beats of the signal. In Figure 2, the first segment is the first half of the audio. $\mathbf{W_{full}}$ is the final model used for separating the first layer from the mixture. As can be seen in the bottom graph of Figure 2, once the full model is learned, the reconstruction error of the first layer drops.

Separation: once the full model $\mathbf{W_{full}}$ is learned, we use it for separation. To perform separation, we construct a binary mask using NMF. $\mathbf{W_{full}}$ is kept fixed, and $\mathbf{H}$ is initialized randomly for the entire mixture. The objective function described in Section 3 is minimized only over $\mathbf{H}$. Once $\mathbf{H}$ is found, $\mathbf{W_{full}H}$ tells us when the elements of $\mathbf{W_{full}}$ are active. We use a binary mask for separation, obtained via:

$$\mathbf{M} = round(\mathbf{W_{full}H} \oslash max(\mathbf{W_{full}H}, abs(\mathbf{X})))$$

where $\oslash$ indicates element-wise division and $\odot$ is element-wise multiplication. We reconstruct the layer using:

$$\mathbf{X}_{layer} = \mathbf{M} \odot \mathbf{X} \quad (3)$$

$$\mathbf{X}_{residual} = (\mathbf{1} - \mathbf{M}) \odot \mathbf{X} \quad (4)$$

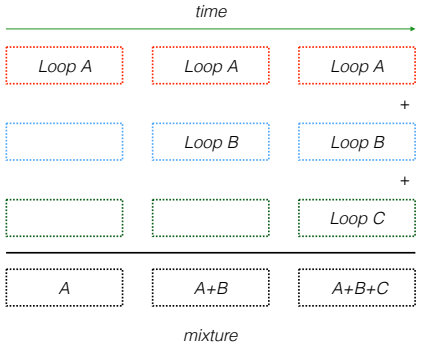


Figure 3. Construction of a single mixture using a layering structure in our dataset, from 3 randomly selected loops each from 3 sets A, B, and C.

where $\odot$ indicates element-wise product. $\mathbf{X}_{residual}$ is the mixture without the layer. We restart at the **estimation** stage above, this time using $\mathbf{X}_{residual}$ as the input, and setting the start point to the segmentation boundary found in the **segmentation** stage above. Taking the inverse Fourier transform of $\mathbf{X}_{layer}$ gives us the audio signal of the separated layer.

Termination: if $\mathbf{X}_{residual}$ is empty (no source groups remain in the mixture), we terminate.

5. EVALUATION

5.1 Dataset

We evaluate our approach in two ways: separation quality, and segmentation accuracy. To do this, we construct a dataset where ground truth is known for separation and segmentation. As our approach looks for a layering structure, we devise mixtures where this layering occurs. We obtain source audio from *Looperman* [1], an online resource for musicians and composers looking for loops and samples to use in their creative work. Each loop from *Looperman* is intended by its contributor to represent a single source. Each loop can consist of a single sound producing source (e.g. solo piano) or a complex group of sources working together (e.g. a highly varied drumkit).

From *Looperman*, we downloaded 15 of these loops, each 8 seconds long at 120 beats per minute. These loops are divided into three sets of 5 loops each. Set A contained 5 loops of rhythmic material (drum-kit based loops mixed with electronics), set B contained 5 loops of harmonic and rhythmic material performed on guitars, and set C contained 5 loops of piano. We arranged these loops to create mixtures that had layering structure, as seen in Figure 3.

We start with a random loop from set A, then add a random loop from B, then add a random loop from C, for a total length of 24 seconds. We produce 125 of these mixtures.

Ground truth segmentation boundaries are at 8 seconds (when the second loop comes in), and at 16 seconds (when the third loop comes in). In Figure 3, each row is ground truth for separation.

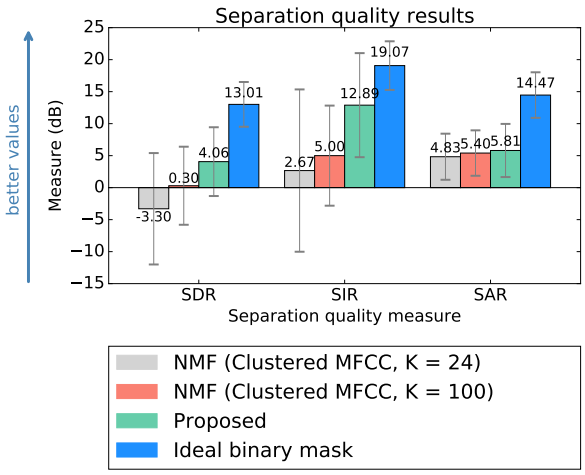


Figure 4. Separation performance of current and proposed algorithms, and an ideal binary mask. Higher numbers are better. The ideal binary mask is an upper bound on separation performance. The error bars indicate standard deviations above and below the mean.

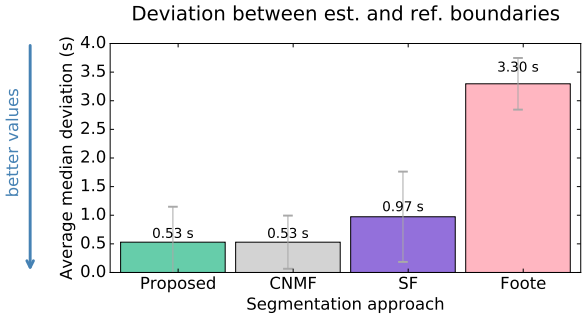


Figure 5. Segmentation performance of current and proposed algorithms. *Est. and ref.* refers to the median deviation in seconds between a ground truth boundary and a boundary estimated by the algorithm. Lower numbers are better. The error bars indicate standard deviations above and below the mean.

5.2 Methods for comparison

For separation, we compare our approach to a separation method in [25]. In this method, they use NMF on the entire mixture spectrogram, and then cluster the components into sources using MFCCs. Each cluster of components is then used to reconstruct a single source in the mixture. In our approach, the number of components (K) was fixed at $K = 8$, giving a total of $K = 24$ components for the entire mixture. For direct comparison, we give the method in [25] $K = 24$ components. We also look at the case where [25] is given $K = 100$ components.

For segmentation, we compare our approach with [22], [17], and [7].

Approach	Median deviation (s)	Avg # of segments
CNMF [17]	.53	6.152
SF [22]	.97	4.024
Foote [7]	3.3	3.048
Proposed	.53	3.216

Table 1. Segmentation results for various approaches. In the dataset, an accurate segmentation reports 3 segments. While CNMF reports similar average median deviation from estimated to reference boundaries to the proposed method, it finds almost twice the number of boundaries. Foote finds a number of segments closer to ground truth, but the boundaries are in the wrong place.

5.3 Results

5.3.1 Separation

To measure separation quality, we use the BSS Eval toolbox [26] as implemented in [19], which reports Source-to-Distortion (SDR), Source-to-Interference (SIR), and Source-to-Artifact (SAR) ratios. For all of these, we compare our proposed approach to an NMF clustering approach based on MFCCs in [25]. This clustering approach was given the number of sources to find in the mixture. This is in contrast to our algorithm, where the number of sources is unknown, and instead is discovered. We also compare to an ideal binary mask. Results are in Figure 4, which shows mean SDR, SIR, and SAR for different source separation methods.

As seen in Figure 4, our approach found sources that correlated with the target sources, giving SDR and SIR more comparable to the ideal binary mask. This is in contrast to the clustering approach, which found sources that poorly correlated with the actual target sources, resulting in low values for SDR and SIR, even when using more components than our approach ($K = 100$ vs. $K = 24$). The clustering mechanism in [25] leverages MFCCs, and finds sources that are related in terms of resonant characteristics (e.g. instrument types) but fails to model sources that have multiple distinct timbres working together.

Our results indicate that separation based on NMF reconstruction error is a useful signal to guide the grouping of spectral templates for NMF, and boost separation quality on layered mixtures.

5.3.2 Segmentation

To measure segmentation accuracy, we use the median absolute time difference from a reference boundary to its nearest estimated boundary, and vice versa. For both of these measures, we compare our proposed approach with [22], [17], and [7], implemented in MSAF [16], as shown in Figure 5.

We find that our approach is as accurate as existing state-of-the-art, as can be seen in Figure 5 and Table 1. Our results indicate that, when finding a segmentation of a mixture, in which segment boundaries are dictated by sources entering the mixture, current approaches are not sufficient. Our approach, because it uses reconstruction error of source models to drive the segmentation, finds more accurate segment boundaries.

6. CONCLUSIONS

We have presented a method for source separation and music segmentation which uses reconstruction error in non-negative matrix factorization to find and model groups of sources according to discovered layered structure. Our method does not require pre-processing of the mixture or post-processing of the basis sets. It requires no user input, or pre-trained external data. It bootstraps an understanding of both the segmentation and the separation from the mixture alone. It is a step towards a framework in which separation and segmentation algorithms can inform one another, for mutual benefit. It makes no assumptions on what a source actually is, but rather finds functional sources implied by a specific type of musical structure.

We showed that tracking reconstruction error of a source model over time in a mixture is a helpful approach to finding structural boundary points in the mixture. These structural boundary points can be used to guide NMF. This separation approach outperforms NMF that clusters spectral templates via heuristics. This work demonstrates a clear, novel, and useful relationship between the problems of separation and segmentation.

The principles behind this approach can be expanded to other source separation approaches. Since source separation algorithms rely on specific cues (e.g. repetition like in REPET, or a spectral model like in NMF), the temporal failure points of source separation algorithms (e.g. the repeating period has failed, or the model found by NMF has failed to reconstruct the mixture) may be a useful cue for music segmentation.

The approach presented here exploits the compositional technique of layering employed in many musical works. For future approaches, we would like to build separation techniques which leverage other compositional techniques and musical structures, perhaps integrating our work with existing work in segmentation.

7. ACKNOWLEDGEMENTS

This work is supported by National Science Foundation Grant 1420971.

8. REFERENCES

[1] Looperman. http://www.looperman.com.

[2] Nicholas J Bryan, Gautham J Mysore, and Ge Wang. Isse: An interactive source separation editor. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 257–266. ACM, 2014.

[3] Zhiyao Duan, Gautham J Mysore, and Paris Smaragdis. Online plca for real-time semi-supervised source separation. In *Latent Variable Analysis and Signal Separation*, pages 34–41. Springer, 2012.

[4] Zhiyao Duan and Bryan Pardo. Soundprism: An online system for score-informed source separation of music audio. *Selected Topics in Signal Processing, IEEE Journal of*, 5(6):1205–1215, 2011.

[5] Daniel PW Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[6] Sebastian Ewert, Bryan Pardo, Mathias Muller, and Mark D Plumbley. Score-informed source separation for musical audio recordings: An overview. *Signal Processing Magazine, IEEE*, 31(3):116–124, 2014.

[7] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, volume 1, pages 452–455. IEEE, 2000.

[8] Toni Heittola, Anssi Klapuri, and Tuomas Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. In *ISMIR*, pages 327–332, 2009.

[9] Rajesh Jaiswal, Derry FitzGerald, Dan Barry, Eugene Coyle, and Scott Rickard. Clustering nmf basis functions using shifted nmf for monaural sound source separation. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 245–248. IEEE, 2011.

[10] Florian Kaiser and Thomas Sikora. Music structure discovery in popular music using non-negative matrix factorization. In *ISMIR*, pages 429–434, 2010.

[11] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[12] Mark Levy and Mark Sandler. Structural segmentation of musical audio by constrained clustering. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):318–326, 2008.

[13] Josh H McDermott. The cocktail party problem. *Current Biology*, 19(22):R1024–R1027, 2009.

[14] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th Python in Science Conference*, 2015.

[15] Gautham J Mysore, Paris Smaragdis, and Bhiksha Raj. Non-negative hidden markov modeling of audio with application to source separation. In *Latent variable analysis and signal separation*, pages 140–148. Springer, 2010.

[16] O. Nieto and J. P. Bello. Msaf: Music structure analytis framework. In *The 16th International Society for Music Information Retrieval Conference*, 2015.

[17] Oriol Nieto and Tristan Jehan. Convex non-negative matrix factorization for automatic music structure identification. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 236–240. IEEE, 2013.

[18] Mark D Plumbley, Samer A Abdallah, Juan Pablo Bello, Mike E Davies, Giuliano Monti, and Mark B Sandler. Automatic music transcription and audio source separation. *Cybernetics &Systems*, 33(6):603–627, 2002.

[19] Colin Raffel, Brian McFee, Eric J Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, Daniel PW Ellis, and C Colin Raffel. mir_eval: A transparent implementation of common mir metrics. *Proc. of the 15th International Society for Music Information Retrieval Conference*, 2014.

[20] Zafar Rafii and Bryan Pardo. Music/voice separation using the similarity matrix. In *ISMIR*, pages 583–588, 2012.

[21] Zafar Rafii and Bryan Pardo. Repeating pattern extraction technique (repet): A simple method for music/voice separation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 21(1):73–84, 2013.

[22] Jean Serra, Mathias Muller, Peter Grosche, and Josep Ll Arcos. Unsupervised music structure annotation by time series structure features and segment similarity. *Multimedia, IEEE Transactions on*, 16(5):1229–1240, 2014.

[23] Madhusudana Shashanka, Bhiksha Raj, and Paris Smaragdis. Probabilistic latent variable models as nonnegative factorizations. *Computational intelligence and neuroscience*, 2008, 2008.

[24] Paris Smaragdis. Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs. In *Independent Component Analysis and Blind Signal Separation*, pages 494–499. Springer, 2004.

[25] Martin Spiertz and Volker Gnann. Source-filter based clustering for monaural blind source separation. In *Proceedings of International Conference on Digital Audio Effects DAFx09*, 2009.

[26] Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte. Performance measurement in blind audio source separation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(4):1462–1469, 2006.

[27] Ron J Weiss and Juan P Bello. Unsupervised discovery of temporal structure in music. *Selected Topics in Signal Processing, IEEE Journal of*, 5(6):1240–1251, 2011.

[28] John F Woodruff, Bryan Pardo, and Roger B Dannenberg. Remixing stereo music with score-informed source separation. In *ISMIR*, pages 314–319, 2006.

TOWARDS MODELING AND DECOMPOSING LOOP-BASED ELECTRONIC MUSIC

Patricio López-Serrano **Christian Dittmar** **Jonathan Driedger** **Meinard Müller**
International Audio Laboratories Erlangen, Germany
patricio.lopez.serrano@audiolabs-erlangen.de

ABSTRACT

Electronic Music (EM) is a popular family of genres which has increasingly received attention as a research subject in the field of MIR. A fundamental structural unit in EM are loops—audio fragments whose length can span several seconds. The devices commonly used to produce EM, such as sequencers and digital audio workstations, impose a musical structure in which loops are repeatedly triggered and overlaid. This particular structure allows new perspectives on well-known MIR tasks. In this paper we first review a prototypical production technique for EM from which we derive a simplified model. We then use our model to illustrate approaches for the following task: given a set of loops that were used to produce a track, decompose the track by finding the points in time at which each loop was activated. To this end, we repurpose established MIR techniques such as fingerprinting and non-negative matrix factor deconvolution.

1. INTRODUCTION

With the advent of affordable electronic music production technology, various loop-based genres emerged: techno, house, drum'n'bass and some forms of hip hop; this family of genres is subsumed under the umbrella term *Electronic Music* (EM). EM has garnered mainstream attention within the past two decades and has recently become a popular subject in MIR: standard tasks have been applied to EM (structure analysis [17]); new tasks have been developed (breakbeat analysis and resequencing [7, 8]); and specialized datasets have been compiled [9].

A central characteristic of EM that has not been extensively considered is its sequencer-centric composition. As noted by Collins [4], *loops* are an essential element of EM: loops are short audio fragments that are "generally associated with a single instrumental sound" [3]. Figure 1 illustrates a simplified EM track structure similar to that encouraged by digital audio workstations (DAWs) such as *Ableton Live* [1]. The track starts with the activation of a drum loop (blue, bottom row). After one cycle, a melody loop (yellow, middle row) is added, while the drum loop continues to play. A third layer—the bass (red, top row)—is activated in the third cycle. Over the course of the track, these loops are activated and deactivated. An important observation is that all appearances of a loop are identical; a property that can be modeled and exploited in MIR tasks. In particular, we consider the task of *decomposing* an EM track: given the set of loops that were used to produce a track and the final, *downmixed* version of the track itself, we wish to retrieve the set of timepoints at which each loop was activated.

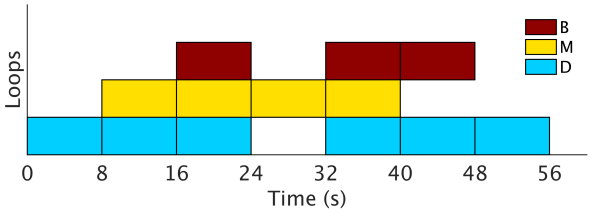


Figure 1. A condensed EM track built with three loop layers: drums (D), melody (M) and bass (B). Each block denotes the activation of the corresponding pattern during the time it spans.

This work offers three main contributions. First, we review the production process of EM and how it leads to the prototypical structure outlined previously (Section 2). Second, we propose a simplified formal model that captures these structural characteristics (Section 3). Third, we use our model to approach the EM decomposition task from two angles: first, we interpret it within a standard retrieval scenario by using fingerprinting and diagonal matching (Section 4). Our second approach is based on non-negative matrix factor deconvolution (NMFD), a technique commonly used for audio source separation (Section 5). We summarize our findings and discuss open issues in Section 6.

2. STRUCTURE AND PRODUCTION PROCESS

Unlike other genres, EM is often produced by starting with a single distinct musical pattern [19] (also called *loop*) and then adding and subtracting further musical material to shape the tension and listener's expectation. An EM track is built by combining layers (with potentially differ-

© Patricio López-Serrano, Christian Dittmar, Jonathan Driedger, Meinard Müller. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Patricio López-Serrano, Christian Dittmar, Jonathan Driedger, Meinard Müller. "Towards Modeling and Decomposing Loop-Based Electronic Music", 17th International Society for Music Information Retrieval Conference, 2016.

ent lengths) in looping cyclical time—where the overall form corresponds to the multitrack layout of sequencers and digital audio workstations (DAWs) [4]. Figure 1 provides a simple example of such a track (total duration 56 s), consisting of three layers or loops: drums (D), bass (B) and melody (M), each with a duration of 8 s. We will be using this track as a running example to clarify the points made throughout this paper.

A common characteristic of EM tracks is their relative *sparseness* or low timbral complexity during the intro and outro—in other words, a single loop is active. This practice is rooted in two facts: Firstly, EM tracks are conceived not as isolated units, but rather as part of a seamless mix (performed by a DJ), where two or more tracks are overlaid together. Thus, in what could be termed *DJ-friendly* tracks [4], a single, clearly percussive element at the beginning and end facilitates the task of beat matching [3] and helps avoid unpleasantly dense transitions. We have constructed our running example following this principle: in Figure 1, the only active layer during the intro and outro is the drum loop (bottom row, blue).

The second reason for having a single-layer intro is that this section presents the track's main elements, making the listener aware of the sounds [3]. Once the listener has become familiar with the main musical idea expressed in the intro, more layers are progressively brought in to increase the tension (also known as a *buildup*), culminating in what Butler [3] designates as the track's *core*: the "thicker middle sections" where all loop layers are simultaneously active. This is reflected in Figure 1, where the melody is brought in at 8 s and the bass at 16 s, overlapping with uninterrupted drums. After the core has been reached, the majority of layers are muted or removed—once again, to create musical anticipation—in a section usually known as *break* or *breakdown* (see the region between 24–32 s in Figure 1, where only the melody is active). To release the musical tension, previous loops are reintroduced after the *breakdown*, (seconds 32–40, Figure 1) only to be gradually removed again, arriving at the outro. In the following sections we will develop a model that captures these structural characteristics and provides a foundation for analyzing EM tracks.

3. SIMPLIFIED MODEL FOR EM

In Section 2 we illustrated the typical form of loop-based electronic music. With this in mind, our goal is to analyze an EM track's structure. More specifically, our method takes as input the set of loops or patterns that were used to produce a track, as well as the final, *downmixed* version of the track itself. From these, we wish to retrieve the set of timepoints at which each loop was activated within the track. We begin by formalizing the necessary input elements.

Let $V \in \mathbb{R}^{K \times M}$ be the feature representation of an EM track, where $K \in \mathbb{N}$ is the feature dimensionality and $M \in \mathbb{N}$ represents the number of elements or frames along the time axis. We assume that the track was constructed from a set of R patterns $P^r \in \mathbb{R}^{K \times T^r}$,

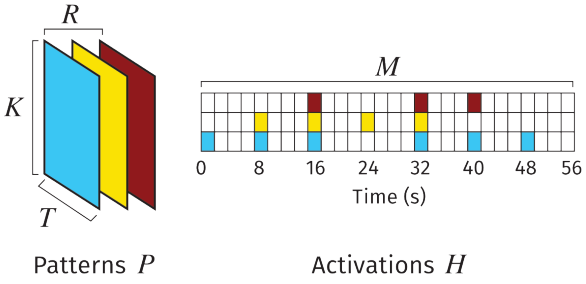


Figure 2. **(Left)**: Tensor P with three patterns (drums, bass, melody). **(Right)**: Activation matrix H with three rows; the colored cells denote an activation of the corresponding pattern.

$r \in [0 : R-1] := \{0, \ldots, R-1\}$. The parameter $T^r \in \mathbb{N}$ is the number of feature frames or observations for pattern P^r. In practice, the patterns can have different lengths—however, without loss of generality, we define their lengths to be the same $T := T^0 = \ldots = T^{R-1}$, which could be achieved by adequately zero-padding shorter patterns until they reach the length of the longest. Based on this assumption, the patterns can be grouped into a pattern tensor $P \in \mathbb{R}^{K \times R \times T}$. In the case of our running example, seen in Figure 1, $T \mathrel{\hat{=}} 8$ s and the number of patterns is $R = 3$. Consequently, the subdimension of the tensor which refers to a specific pattern with index r is $P^r := P(\cdot,\ r,\ \cdot)$ (i. e., the feature matrix for either (D), (M), or (B) in our example); whereas $P_t := P(\cdot,\ \cdot,\ t)$ refers to frame index t simultaneously in all patterns.

In order to construct the feature representation V from the pattern tensor P, we require an activation matrix $H \in \mathbb{B}^{R \times M}$ with $\mathbb{B} := \{0, 1\}$, such that

$$V \mathrel{\hat{=}} \sum_{t=0}^{T-1} P_t \cdot \overset{t\rightarrow}{H}, \tag{1}$$

where $\overset{t\rightarrow}{(\cdot)}$ denotes a frame shift operator [18]. Figure 2 depicts P and H as constructed for our running example. The model assumes that the sum of pattern signals and their respective transformations to a feature representation are linear, which may not always be the case. The additive assumption of Eq. 1 implies that no time-varying and/or non-linear effects were added to the mixture (such as compression, distiortion, or filtering), which are often present in real-world EM. Aside from this, we specify a number of further constraints below.

The devices used to produce early EM imposed a series of technical constraints which we formalize here. Although many of these constraints were subsequently eliminated in more modern equipment and DAWs, they have been ingrained into the music's aesthetic and remain in use up to the present day.

Non-overlap constraint: A pattern is never superimposed with itself, i. e., the distance between two activations of any given pattern is always equal to or greater than the pattern's length. Patterns are loaded into a device's mem-

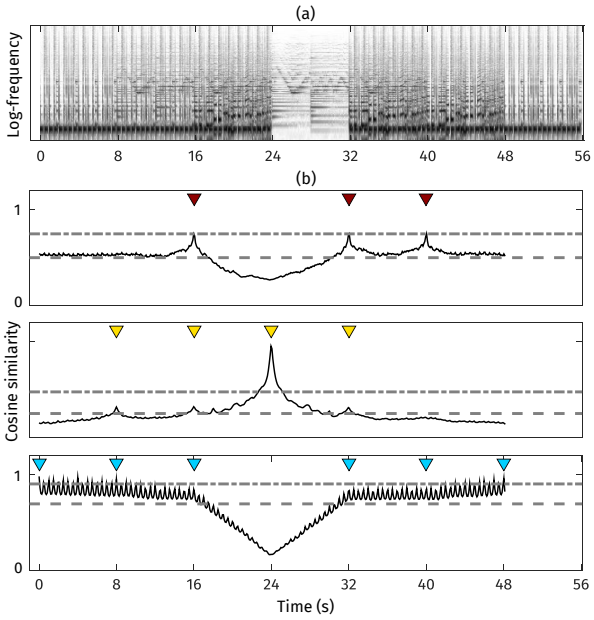


Figure 3. **(a)**: Log-frequency spectrogram for entire track. **(b)**: Matching curves computed using cosine similarity for drums, melody, and bass (bottom to top). The dashed line represents the curve's global mean; the dash-dotted line is the GT mean (see Section 4.4 for definition of *gain*). Colored triangles indicate GT loop activation positions.

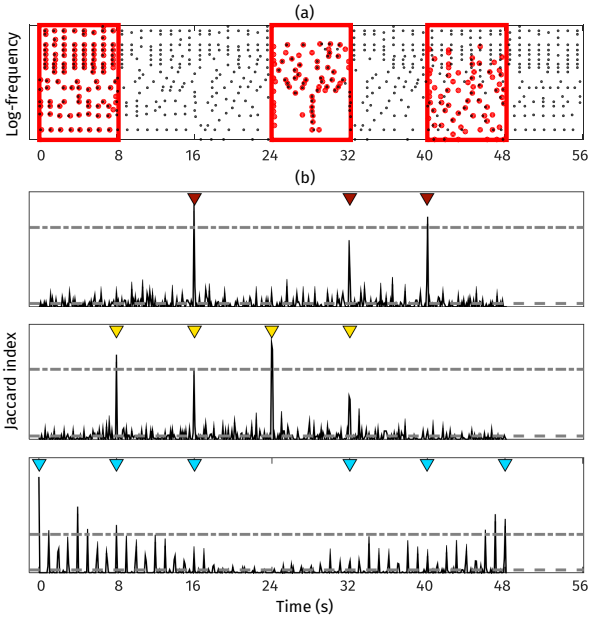


Figure 4. **(a)**: Log-frequency spectral peak map for the entire track (black dots) and for each query (red dots enclosed in red, from left to right: drums, melody, and bass). **(b)**: Matching curves computed with the Jaccard index and each pattern as a query for drums, melody, and bass (bottom to top).

ory and triggered by a sequencer—usually without further activation signals before it has played through to the end. If a pattern P^r is activated at time $m \in [0 : M-1]$, then $H^r(m) \neq 0 \Rightarrow H^r(m+1) = \ldots = H^r(m+T-1) = 0$.

Length constraint: As noted by [4], multiple layers in EM are complementary, creating aggregate effects and capable of being independently inserted and removed. For this reason, we make the simplifying assumption that $T := T^0 = T^1 = \ldots = T^{R-1}$, i. e., that all patterns have the same length.

T-grid constraint: Motivated by the use of centralized MIDI clocks and the fixed amount of musical time available on prevalent devices such as drum machines (which typically allow programming one musical measure at a time, in 16 steps), we enforce a timing grid which restricts the possible activation points in H. In Figure 1, patterns are always introduced and removed at multiples of 8 s.

Amplitude constraint: We assume that a pattern is always activated with the same *intensity* throughout a track, and therefore each row r in the activation matrix H fulfills $H^r := H(r, \cdot) \in \mathbb{B}^{1\times M}$.

4. FINGERPRINT-BASED EM DECOMPOSITION

In the running example, multiple patterns are overlaid in different configurations to form the track. If we know *a priori* which patterns are included and wish to find their respective activation positions, we need a technique capable of identifying an audio query within a database where further musical material is superimposed. We first examine log-frequency spectrograms and diagonal matching as a baseline approach, and continue with audio fingerprinting techniques based on spectral peaks in combination with various similarity measures. In Section 5 we discuss an alternative approach based on NMFD. The running example is constructed with one audio file for each pattern and a generic EM track arrangement seen in Figure 1. The complete track is generated in the time domain by summing the individual patterns that are active at a given point in time. All audio files have been downmixed to mono with a sampling rate $F_s = 22050$ Hz.

4.1 Diagonal Matching

We implement the diagonal matching procedure outlined in [13, pp. 376–378] to measure the *similarity* between each query pattern P^r and the track feature matrix V. In simple terms, to test if and where the query $P^r = (P^r_0, \ldots, P^r_{T-1})$ is contained in $V = (V_0, \ldots, V_{M-1})$, we shift the sequence P^r over the sequence V and locally compare P^r with suitable subsequences of V. In general, let $\mathcal{F}$ be the feature space (for example, $\mathcal{F} = \mathbb{R}^K$ in the case of log-frequency spectrograms). A similarity measure $s : \mathcal{F} \times \mathcal{F} \to \mathbb{R} \cap [0, 1]$ between two feature frames will yield a value of 1 if the query is identical to a certain region of the database, and 0 if there is no resemblance at all.

4.2 Baseline Procedure

For each individual pattern, as well as the entire track, we compute an STFT $\mathcal{X}$ with the following parameters: block size $N = 4096$, hop size $H = N/2$ and a Hann window. From these, magnitude spectrograms (abbreviated as MS) are computed and mapped to a logarithmically spaced frequency axis with a lower cutoff frequency of 32 Hz, an upper cutoff frequency of 8000 Hz and spectral selectivity of 36 bins per octave (abbreviated LS). Under these STFT settings, the 36-bin spectral selectivity does not hold in the two lowest octaves; however, their spectral peak contribution is negligible. Preliminary experiments have shown that this musically meaningful feature representation is beneficial for the matching procedure both in terms of efficiency and accuracy. We begin with a baseline experiment (Figure 3), using LS and cosine similarity:

$$\textit{Cosine} : s_{\cos}(u,v) := 1 - \frac{\langle u|v\rangle}{||u|| \cdot ||v||},\ u,v \in \mathbb{R}^K \setminus \{0\}. \tag{2}$$

Notice that the clearest peak is produced by the melody activation at 24 s (Figure 3b, middle row), which occurs without any other patterns being overlaid. The three remaining activation points for the melody have a very low gain relative to their neighboring values. The matching curve for the drums (Figure 3b, bottom row) displays a coarse downwards trend starting at 0 s and reaching a global minimum at 24 s (the point at which the drum pattern is not activated in our example); this trend is reversed as the drums are added again at 32 s. The internal repetitivity (or self-similarity) of the drum pattern causes the periodic peaks seen throughout the matching curve. Overall, it is evident from all three curves that the combination of LS with cosine similarity is insufficient to capture the activations when multiple patterns are superimposed—motivating our next experimental configuration which uses spectral peak maps.

4.3 Fingerprinting with Peak Maps

Although our scenario is slightly different to that of audio fingerprinting and identification, both require a feature representation which captures an individual pattern's characteristics despite the superposition of further sound sources. To this end, we use spectral peak maps as described in [13]. Conceptually, we are following an early approach for loop retrieval inside hip hop recordings which was presented in [20] and later refined in [2]. Their method is based on a modification of the fingerprinting procedure originally described in [21].

For each time-frequency bin in the respective LS, a rectangular analysis window is constructed. The maximum value within each window is kept (with the value 1 on the output) and all neighbors are set to 0 on the output. In Figure 4a we show the spectral peak map for the entire track (black dots) and the query peak map for each query pattern (red dots in red rectangles). These log-frequency peak maps populate a pattern tensor $P \in \mathbb{B}^{K\times R\times T}$, where $K = 286$. Thus P^r corresponds to the peak map for the r^{th} pattern, while V corresponds to the entire track.

	Gain		*Pearson*	
	μ	σ	μ	σ
MS/cos	1.72	0.31	0.13	0.05
LS/cos	1.57	0.29	0.11	0.05
PLS/cos	19.46	10.45	0.52	0.18
PLS/inc	21.69	11.90	0.51	0.19
PLS/Jac	19.54	9.76	0.53	0.18

Table 1. Results for diagonal matching experiments with magnitude spectrograms (MS), log-frequency spectrograms (LS), and log-frequency peak maps (PLS) using the cosine, inclusion and Jaccard similarity measures. Each column shows the mean and variance for peak gain and Pearson correlation.

In addition to the cosine measure defined in Eq. 2, we test different similarity measures s:

$$\textit{Jaccard} : s_{\text{Jac}}(u,v) := 1 - \frac{||u \wedge v||}{||u \vee v||},\ u,v \in \mathbb{B}^K, \tag{3}$$

$$\textit{Inclusion} : s_{\text{inc}}(u,v) := 1 - \frac{||u \wedge v||}{||u||},\ u,v \in \mathbb{B}^K, \tag{4}$$

where we set $\frac{0}{0} := 1$. The inclusion metric aims to quantify the extent to which the query is contained or *included* in the database and has a similar definition to the Jaccard index.

4.4 Evaluation

We use two measures to quantify how well the matching curves capture the pattern activations. For the first measure, termed *gain*, we compute the average of the activation values at the ground truth (GT) activation points: in Figures 3b and 4b, these locations are marked by colored triangles, corresponding to each loop in the running example; their mean value is shown as a dash-dotted line. We also compute the mean value for the entire curve (dashed line) and use the ratio between these two means in order to assess the quality of the matching curve. Ideally, the curve assumes large values at the GT activation points and small values elsewhere, resulting in a larger gain. As a second measure we take the *Pearson correlation* between a computed matching curve and its corresponding row H^r in the GT activation matrix, where the activation points have a value of 1, and 0 elsewhere. Again, a high Pearson correlation reflects high matching curve quality.

We generated a set of patterns used to build prototypical EM tracks. To foster reproducible research, we produced them ourselves, avoiding potential copyright issues—they are available under a Creative Commons Attribution-ShareAlike 4.0 International license and can be obtained at the companion website [1]. We chose seven prominent EM subgenres such as *big beat*, *garage* and *drum'n'bass* (in a tempo range between 120–160 BPM). For each subgenre, we generated four patterns in the categories of drums, bass, melody and additional effects.

[1] https://www.audiolabs-erlangen.de/resources/MIR/2016-ISMIR-EMLoop

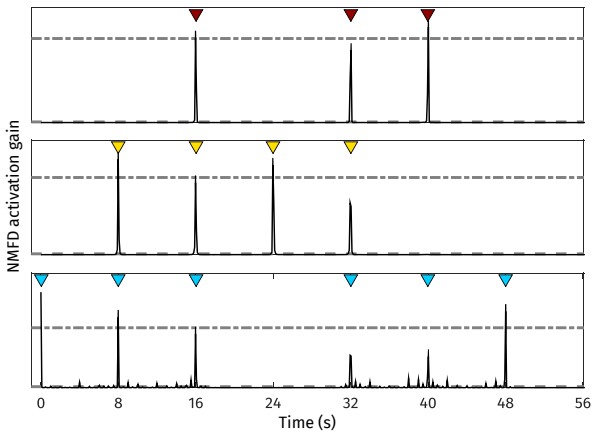


Figure 5. Activation curves learned by NMFD applied to the magnitude spectrogram of the running example. The dashed lines represent the mean value for the complete curve, but are very close to the x-axis.

As stated by Wang [21], spectral peak maps are robust to superposition of multiple sources; a fact which becomes clear when comparing Figures 3b and 4b. In Figure 4b, the *peak gain* has greatly increased compared to the baseline approach with LS. In Table 1 we list the mean peak gain and Pearson correlation for all seven tracks, along with standard deviations for each value. The first two rows, MS/cos and LS/cos, correspond to the baseline approach—the last three rows summarize the experiments with spectral peak maps. Note that spectral peak maps have approximately ten times the peak gain of MS/LS, whereas the Pearson correlation increases by a factor of four. Figures 3 and 4 illustrate the results in Table 1 at an intuitive level. With MS, the spectral content shared among different types of patterns impedes distinct peaks from emerging. By discarding this irrelevant information, LS better represent the characteristics of each pattern. From the perspective of peak quality, only the self-similarity of the drum pattern continues to pose a challenge.

5. NMFD-BASED EM DECOMPOSITION

By design, our model for EM is very close to the formulation of NMFD; in this section we explore the performance of NMFD and compare it with our fingerprinting methods.

5.1 Related Work

In this section, we briefly review the NMFD method that we employ for decomposing the feature representation V. Weiss and Bello [22] used non-negative matrix factorization (NMF) to identify repeating patterns in music. By adding sparsity constraints and shift-invariant probabilistic latent component analysis (SI-PLCA), they automatically identify the number of patterns and their lengths—applied to beat-synchronous chromagrams in popular music. Masuda et al. [12] propose a query-by-audio system based on NMF to identify the locations where a query musical phrase is present in a musical piece. Among more general techniques for investigating alleged music plagiarism, Dittmar et al. [5] proposed a method for retrieval of sampling. Their approach, based on NMF, was not supplemented with systematic evaluation, but was further investigated in [23]. Previous works [6, 11, 16, 18] successfully applied NMFD—a convolutive version of NMF—for drum transcription and separation. Hockman et al. [7, 8] specifically focused on analyzing breakbeats, i. e., drum-only loops as used in hip hop and drum'n'bass . Detecting sample occurrences throughout a track is a secondary aspect, as they address the more challenging scenario of estimating the loop resequencing [8]. All these previous works have in common that they attempt to retrieve one loop inside a song, whereas we pursue a more holistic approach that allows to deconstruct the whole track into loops.

5.2 NMFD Model

Our objective is to decompose V into component magnitude spectrograms that correspond to the distinct musical elements. Conventional NMF can be used to compute a factorization $\mathrm{V} \approx \mathrm{W} \cdot \mathrm{H}$, where the columns of $\mathrm{W} \in \mathbb{R}_{\geq 0}^{K \times R}$ represent spectral basis functions (also called templates) and the rows of $\mathrm{H} \in \mathbb{R}_{\geq 0}^{R \times M}$ contain time-varying gains (also called activations). The rank $R \in \mathbb{N}$ of the approximation (i. e., number of components) is an important but generally unknown parameter. NMFD extends NMF to the convolutive case by using two-dimensional templates so that each of the R spectral bases can be interpreted as a magnitude spectrogram snippet consisting of $T \ll M$ spectral frames. The convolutive spectrogram approximation $\mathrm{V} \approx \Lambda$ is modeled as

$$\Lambda := \sum_{t=0}^{T-1} \mathrm{W}_t \cdot \overset{t\rightarrow}{\mathrm{H}}, \tag{5}$$

where $\overset{t\rightarrow}{(\cdot)}$ denotes a frame shift operator (see also Eq. 1). As before, each column in $\mathrm{W}_t \in \mathbb{R}_{\geq 0}^{K \times R}$ represents the spectral basis of a particular component, but this time we have T different versions W_t, with $t \in [0 : T-1]$ available. If we take lateral slices along the columns of W_t, we can obtain R prototype magnitude spectrograms $\mathrm{U}^r \in \mathbb{R}_{\geq 0}^{K \times T}$. NMFD typically starts with a suitable initialization (with random values or constant values) of matrices $\mathrm{W}_t^{(0)}$ and $\mathrm{H}^{(0)}$. These matrices are iteratively updated to minimize a suitable distance measure between the convolutive approximation Λ and V. In this work, we use the update rules detailed in [18], which extend the well-known update rules for minimizing the Kullback-Leibler Divergence (KLD) [10] to the convolutive case.

5.3 Evaluation

For our experiments with NMFD we used MS and LS to conduct the procedure in two variants. For the first variant (referred to as R in Table 2), the only *a priori* information used is the number of patterns (or templates) R and their length T. The templates are initialized randomly and

	Gain		*Pearson*	
	μ	σ	μ	σ
NMFD@MS, R	55.20	29.80	0.65	0.25
NMFD@MS, RP	89.62	39.67	0.88	0.11
NMFD@LS, R	52.39	35.95	0.64	0.22
NMFD@LS, RP	79.59	34.99	0.87	0.12

Table 2. Results for NMFD with magnitude spectrograms (MS) and log-frequency spectrograms (LS), initializing the number of templates (R) and also the loop templates (RP). Each column shows the mean and variance for peak gain and Pearson correlation.

fifty iterative updates are used to minimize the KLD. To account for the effects of random initialization, we carry out ten initialization passes per track. The results in Table 2 reflect the mean and standard deviation across all passes. For the second variant (RP), we supply the pattern templates themselves at initialization (i. e., R, T and P are known). We also disallow template updates and only allow activation updates. Since the templates in variant R are initialized randomly, there is no direct relationship between the learned activation curves and the corresponding ground truth curves. We deal with this permutation indeterminacy by comparing all computed activation curves with all ground truth curves and taking the results which maximize the overall score. For all configurations in Table 2, we observe a peak gain at least twice as high as that obtained through diagonal matching; the Pearson correlation increases by a factor of 1.2–1.7, depending on the NMFD configuration taken for comparison. Focusing on the differences among NMFD configurations, RP brings peak gain improvements by a factor slightly greater than 1.5; the Pearson correlation increases by about 1.35. The feature choice (MS or LS) does not play a significant role in result quality. Due to the amount of prior knowledge used to initialize the RP configuration, we consider it as an upper bound for less informed approaches.

6. CONCLUSIONS AND FUTURE WORK

In the preceding sections we developed a better understanding of the feature representations and matching techniques that are commonly used for pattern activation discovery. In this section, we reflect on some of the limitations of our work, further research topics, and computational performance issues.

Clearly, our work only provides a baseline for further work towards more realistic scenarios. As to our model's inherent shortcomings, real-world EM tracks usually contain more than four individual patterns, which are rarely available. Moreover, activations of a given pattern are often (spectrally) different from one another due to the use of effects such as delay and reverb, filter sweeps or resequencing. Thus, we consider this study as a stepping stone towards a fully-developed pipeline for EM structure analysis and decomposition. One potential research direction would be the automatic identification of suitable pattern candidates. A repetition-based analysis technique as described in [14] could be used in conjunction with spectral peak maps to compute self-similarity matrices (SSMs) that saliently encode inclusion relationships. Furthermore, semi-informed variants of NMFD might be helpful in discovering additional patterns that are not explicitly given, where the use of rhythmic structure can serve as prior knowledge to initialize the activations. Although diagonal matching curves can be computed efficiently with a straightforward implementation, we have seen they have certain shortcomings; we wish to investigate the feasibility of using them as rough initial guesses and leaving the refinement up to NMFD. Beyond each method's capabilities, as seen in Tables 1 and 2, there is also the issue of their running time and memory requirements. For the running example, we tested our `MATLAB` implementation on a 3.2 GHz Intel Core i5 CPU with 16 GB RAM, yielding the mean execution times in Table 3. From Tables 3 and 2 we can conclude that NMFD@LS offers the best balance between quality and resource intensity. NMFD@MS takes approximately 14 times longer to compute than NMFD@LS and only produces marginally better results. Indeed, recall that the feature dimensionality $K = 286$ for LS and $K = 2049$ for MS, which explains the large difference in execution times.

Method	*Time (s)*
PLS	0.2
NMFD@LS,(R/RP)	2.5
NMFD@MS,(R/RP)	36.0

Table 3. Computation times for diagonal matching with log-spectral peak maps (PLS), NMFD with magnitude spectrograms (MS), and NMFD with log-frequency spectrograms (LS). The choice of initialization R or RP for NMFD does not impact execution time.

As a final remark, musical structure analysis is an ill-defined problem, primarily because of ambiguity; a segmentation may be based on different principles (homogeneity, repetition, novelty) that can conflict with each other [15]. The main advantage of our method is that we avoid the philosophical issue of how a track's structure is *perceived*, and rather attempt to determine how it was *produced*—a univocal problem. It can then be argued that the listeners' perception is influenced by the cues inherent to EM's compositional style.

7. ACKNOWLEDGMENTS

Patricio López-Serrano is supported in part by CONACYT-DAAD. The authors are supported by the German Research Foundation (DFG MU 2686/6-1, DFG MU 2686/7-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institute for Integrated Circuits IIS. This work's core ideas were born at HAMR@ISMIR 2015: we thank our teammate Hendrik Schreiber, Colin Raffel, the organizers, and the reviewers for their valuable input.

8. REFERENCES

[1] Ableton. Live. `https://www.ableton.com/en/live/` (web source, retrieved March 2016), 2016.

[2] Jan Balen, Joan Serrà, and Martín Haro. *From Sounds to Music and Emotions: 9th Int. Symposium, CMMR 2012, London, UK, June 19-22, 2012, Revised Selected Papers*, chapter Sample Identification in Hip Hop Music, pages 301–312. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[3] Mark J. Butler. *Unlocking the Groove: Rhythm, Meter, and Musical Design in Electronic Dance Music*. Profiles in popular music. Indiana University Press, 2006.

[4] Nick Collins, Margaret Schedel, and Scott Wilson. *Electronic Music*. Cambridge Introductions to Music. Cambridge University Press, Cambridge, United Kingdom, 2013.

[5] Christian Dittmar, Kay Hildebrand, Daniel Gärtner, Manuel Winges, Florian Müller, and Patrick Aichroth. Audio forensics meets music information retrieval - a toolbox for inspection of music plagiarism. In *European Sig. Proc. Conf. (EUSIPCO)*, pages 1249–1253, Bucharest, Romania, August 2012.

[6] Christian Dittmar and Meinard Müller. Towards transient restoration in score-informed audio decomposition. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx)*, pages 145–152, Trondheim, Norway, December 2015.

[7] Jason A. Hockman, Matthew E. P. Davies, and Ichiro Fujinaga. One in the Jungle: Downbeat Detection in Hardcore, Jungle, and Drum and Bass. In *Proc. of the Int. Soc. for Music Inf. Retrieval Conf. (ISMIR)*, pages 169–174, Porto, Portugal, October 2012.

[8] Jason A. Hockman, Matthew E. P. Davies, and Ichiro Fujinaga. Computational strategies for breakbeat classification and resequencing in Hardcore, Jungle and Drum & Bass. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx)*, Trondheim, Norway, December 2015.

[9] Peter Knees, Ángel Faraldo, Perfecto Herrera, Richard Vogl, Sebastian Böck, Florian Hörschläger, and Mickael Le Goff. Two data sets for tempo estimation and key detection in electronic dance music annotated from user corrections. In *Proc. of the Int. Soc. for Music Inf. Retrieval Conf., ISMIR 2015, Málaga, Spain, October 26-30, 2015*, pages 364–370, 2015.

[10] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *Proc. of the Neural Inf. Processing Systems (NIPS)*, pages 556–562, Denver, CO, USA, 2000.

[11] Henry Lindsay-Smith, Skot McDonald, and Mark Sandler. Drumkit transcription via convolutive NMF. In *Proc. of the Int. Conf. on Digital Audio Effects Conf. (DAFx)*, York, UK, September 2012.

[12] Taro Masuda, Kazuyoshi Yoshii, Masataka Goto, and Shigeo Morishima. Spotting a query phrase from polyphonic music audio signals based on semi-supervised nonnegative matrix factorization. In *Proc. of the Int. Soc. for Music Inf. Retrieval Conf. (ISMIR)*, pages 227–232, Taipei, Taiwan, 2014.

[13] Meinard Müller. *Fundamentals of Music Processing*. Springer Verlag, 2015.

[14] Meinard Müller, Nanzhu Jiang, and Harald Grohganz. SM Toolbox: MATLAB implementations for computing and enhancing similiarty matrices. In *Proc. of the Audio Engineering Soc. AES Conf. on Semantic Audio*, London, UK, 2014.

[15] Jouni Paulus, Meinard Müller, and Anssi P. Klapuri. Audio-based music structure analysis. In *Proc. of the Int. Soc. for Music Inf. Retrieval Conf. (ISMIR)*, pages 625–636, Utrecht, The Netherlands, 2010.

[16] Axel Röbel, Jordi Pons, Marco Liuni, and Mathieu Lagrange. On automatic drum transcription using non-negative matrix deconvolution and itakura saito divergence. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, pages 414–418, Brisbane, Australia, April 2015.

[17] Bruno Rocha, Niels Bogaards, and Aline Honingh. Segmentation and timbre similarity in electronic dance music. In *Proc. of the Sound and Music Computing Conf. (SMC)*, pages 754–761, Stockholm, Sweden, 2013.

[18] Paris Smaragdis. Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs. In *Proc. of the Int. Conf. on Independent Component Analysis and Blind Signal Separation ICA*, pages 494–499, Grenada, Spain, 2004.

[19] Rick Snoman. *Dance Music Manual: Tools, Toys, and Techniques*. Taylor & Francis, 2013.

[20] Jan Van Balen. Automatic recognition of samples in musical audio. Master's thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2011.

[21] Avery Wang. An industrial strength audio search algorithm. In *Proc. of the Int. Soc. for Music Inf. Retrieval Conf. (ISMIR)*, pages 7–13, Baltimore, Maryland, USA, 2003.

[22] Ron J. Weiss and Juan Pablo Bello. Unsupervised discovery of temporal structure in music. *IEEE Journal of Selected Topics in Sig. Proc.*, 5:1240–1251, 2011.

[23] Jordan L. Whitney. Automatic recognition of samples in hip-hop music through non-negative matrix factorization. Master's thesis, University of Miami, 2013.

TWO (NOTE) HEADS ARE BETTER THAN ONE: PEN-BASED MULTIMODAL INTERACTION WITH MUSIC SCORES

Jorge Calvo-Zaragoza, David Rizo, Jose M. Iñesta
Pattern Recognition and Artificial Intelligence Group
Department of Software and Computing Systems
University of Alicante, Spain
{jcalvo,drizo,inesta}@dlsi.ua.es

ABSTRACT

Digitizing early music sources requires new ways of dealing with musical documents. Assuming that current technologies cannot guarantee a perfect automatic transcription, our intention is to develop an interactive system in which user and software collaborate to complete the task. Since conventional score post-editing might be tedious, the user is allowed to interact using an electronic pen. Although this provides a more ergonomic interface, this interaction must be decoded as well. In our framework, the user traces the symbols using the electronic pen over a digital surface, which provides both the underlying image (offline data) and the drawing made by the e-pen (online data) to improve classification. Applying this methodology over 70 scores of the target musical archive, a dataset of 10 230 bimodal samples of 30 different symbols was obtained and made available for research purposes. This paper presents experimental results on classification over this dataset, in which symbols are recognized by combining the two modalities. This combination of modes has demonstrated its good performance, decreasing the error rate of using each modality separately and achieving an almost error-free performance.

1. INTRODUCTION

Music constitutes one of the main tools for cultural transmission. That is why musical documents have been preserved over the centuries, scattered through cathedrals, museums, or historical archives. In an effort to prevent their deterioration, access to these sources is not always possible. This implies that an important part of this historical heritage remains inaccessible for musicological study. Occasionally, these documents are transcribed to a digital format for easier access, distribution and study, without compromising their integrity.

On the other hand, it is important to point out that the massive digitization of music documents also opens several opportunities to apply Music Information Retrieval algorithms, which may be of great interest. Since the manual transcription of these sources is a long, tedious task, the development of automatic transcription systems for early music documents is gaining importance in the last few years.

Optical Music Recognition (OMR) is a field devoted to providing computers the ability to extract the musical content of a score from the optical scanning of its source. The output of an OMR system is the music score encoded in some structured digital format such as MusicXML, MIDI or MEI. Typically, the transcription of early music documents is treated differently with respect to conventional OMR methods due to specific features (for instance, the different notation or the quality of the document). Although there exist several works focused on early music documents transcription [9,10], the specificity of each type of notation or writing makes it difficult to generalize these developments. This is especially detrimental to the evolution of the field because it is necessary to implement new processing techniques for each type of archive. Even worse, new labelled data are also needed to develop techniques for automatic recognition, which might imply a significant cost.

Notwithstanding the efforts devoted to improving these systems, their performance is far from being optimal [12]. In fact, assuming that a totally accurate automatic transcription is not possible, and might never be, user-centred recognition is becoming an emergent framework. Instead of a fully-automatized process, *computer-aided* systems are being considered, with which the user collaborates actively to complete the recognition task [16].

The goal of this kind of systems is to facilitate the task for the user, since it is considered the most valuable resource [2]. In the case of the transcription of early music documents, the potential user is the expert musicologist who understands the meaning of any nuance that appears in the score. However, very often these users find the use of a pen more natural and comfortable than keyboard entry or drag-and-drop actions with the mouse. Using a tablet device and e-pen, it is possible to develop an ergonomic interface to receive feedback from users' drawings. This is specially true for score post-edition where the user, instead of sequentially inputting symbols has to correct some of them, and for that, direct manipulation is the preferred interaction style.

© Jorge Calvo-Zaragoza, David Rizo, Jose M. Iñesta. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jorge Calvo-Zaragoza, David Rizo, Jose M. Iñesta. "Two (note) heads are better than one: pen-based multimodal interaction with music scores", 17th International Society for Music Information Retrieval Conference, 2016.

Such an interface could be used to amend errors made by the system in a simpler way for the user, as has been proposed for automatic text recognition [1]. However, there are studies showing that, when the task is too complex, users prefer to complete the task by themselves because the human-machine interaction is not friendly enough [14]. Therefore, this interface could also be used to develop a manual transcription system that would be more convenient and intuitive than conventional score editors. Moreover, this transcription system might be useful in early stages of an OMR development, as it could be used to acquire training data more efficiently and ergonomically, which is specially interesting for old music notations.

Unfortunately, although the user is provided with a more friendly interface to interact with the system, the feedback input is not deterministic this way. Unlike the keyboard or mouse entry, for which it is clear what the user is inputting, the pen-based interaction has to be decoded and this process might have errors.

For all the reasons above, this article presents our research on the capabilities of musical notation recognition with a system whose input is a pen-based interface. To this end, we shall assume a framework in which the user traces symbols on the score, regardless of the purpose of this interaction (OMR error correction, digitizing the content, acquire labelled data, etc.). As a result, the system receives a multimodal signal: on one hand, the sequence of points that indicates the path followed by the e-pen on the digital surface —usually referred to as *online* modality; on the other hand, the piece of image below the drawn, which contains the original traced symbol —*offline* mode. One of the main hypothesis of this study is that the combination of both modalities leads to better results than using just either the pen data or the symbol image.

The rest of the paper is structured as follows: Section 2 introduces the corpora collected and utilized, which comprises data of Spanish early music written in White Mensural notation; Section 3 describes a multimodal classifier that exploits both offline and online data; Section 4 presents the results obtained with such classifier; and Section 5 concludes the present work.

2. MULTIMODAL DATA COLLECTION

This work is a first seed of a case study to digitize a historical musical archive of early Spanish music. The final objective of the whole project is to encode the musical content of a huge archive of manuscripts dated between 16th and 18th centuries, handwritten in mensural notation, in the variant of the Spanish notation at that time [5]. A short sample of a piece from this kind of document is illustrated in Figure 1.

This section describes the process developed to collect multimodal data of isolated musical symbol from images of scores. A massive collection of data will allow us to develop a more effective classification system and to go deeper into the analysis of this kind of interaction. Let us note that the important point in our interactive system is to better understand user actions. While a machine is assumed to make some mistakes, it is unacceptable to force the user to draw the same symbol of score many times. To this end, our intention is to exploit both offline data (image) and online data (e-pen user tracing) received.

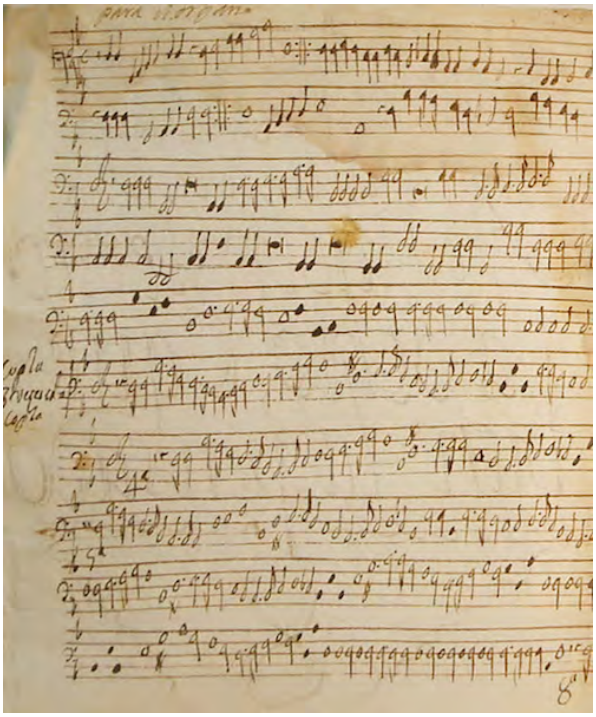
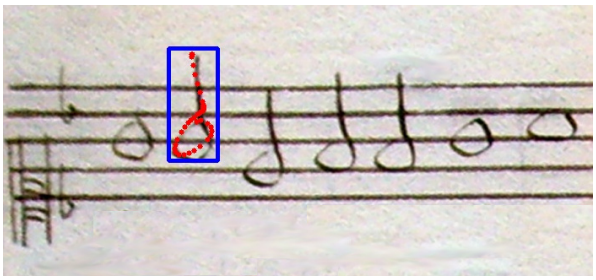


Figure 1. Example of page of a music book written in handwritten white mensural notation from Spanish manuscripts of centuries 16th to 18th.

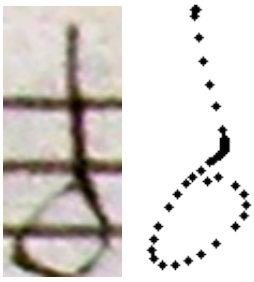
Our idea is to simulate the same scenario of a real application. Therefore, we loaded the images of the scores on a digital surface to make users trace the symbols using the electronic pen. The natural symbol isolation of this kind of input is the set of strokes —data collected between pen-down and pen-up actions. To allow tracing symbols with several strokes, a fixed elapsed time is used to detect when a symbol has been completed. If a new stroke starts before this time lapse, it is considered to belong to the same symbol than the previous one.

Once online data is collected and manually grouped into symbol classes, the offline data is also extracted from this information. A bounding box is obtained from each group of strokes belonging to the same symbol, storing the maximum and minimum values of each coordinate (plus a small margin) among all the trace points collected. This bounding box indicates where the traced symbol can be found in the image. Therefore, with the sole effort of the tracing process, both online and offline data are collected. Note that the extraction of the offline data is driven by the tracing process, instead of deciding at every moment the bounds of each symbol.

Figure 2 illustrates the process explained above for a single symbol. Although the online data is drawn in this example, the actual information stored is the sequence of 2D points in the same order they were collected, indicating



(a) Tracing process



(b) Offline data (c) Online data

Figure 2. Example of extraction of a *minima*. Above, the sequence of points collected by the e-pen. The box represents the bounding box of the sequence. Below, the multimodal data extracted from the same sample.

the path followed by the e-pen.

Following this approach, several advantages are found: the final effort of collecting multimodal data is halved, since the online data collection simultaneously provides the offline data collection; the collected data mimics the scenario that might be found in the final application, when the user interacts with the machine; and the process becomes more user-friendly, which usually leads to a lower number of errors.

The collection was extracted by five different users from 70 different musical scores of different styles from the Spanish white mensural notation of 16th-18th centuries. The *Samsung Galaxy Note Pro 12.2* device (247 ppi resolution) was used and symbols were written by means of the stylus *S-Pen*. All the score images used are in the same scale, in which staff lines spacing is about 24 DP. [1] Due to the irregular conditions of the documents, this value is approximate but it can be used for normalizing with respect to other scores.

The obtained dataset consists of 10230 samples, each of which contains both a piece of image and the strokes followed during its tracing. These samples are spread over 30 classes. Table 1 lists the set of labels, including a typographic example and the number of samples per each. The number of symbols of each class is not balanced but it depicts the same distribution found in the documents.

Every symbol that must be differentiated for preservation purposes was considered as a different class. For

[1] DP stands for *device independent pixels* in (Android) mobile application development

Label	Image	Count
barline		46
brevis		210
coloured brevis		28
brevis rest		171
c-clef		169
common time		29
cut time		56
dot		817
double barline		73
custos		285
f-clef 1		52
f-clef 2		43
fermata		75
flat		274
g-clef		174
beam		85
longa		30
longa rest		211
minima		2695
coloured minima		1578
minima rest		427
proportio minor		28
semibrevis		1109
coloured semibrevis		262
semibrevis rest		246
semiminima		328
coloured semiminima		403
semiminima rest		131
sharp		170
proportio maior		25

Table 1. Details of the dataset obtained through the tracing process over 70 scores (images from 'Capitán' font).

instance, there are two *f-clef* types because the graphical symbol is quite different despite having the same musical meaning. However, the orientation of the symbols does not make a different class since the same graphical representation with a vertical inversion can be found. In the case it was needed, the orientation could be obtained through an easy post-processing step.

We are making the dataset freely available at `http://grfia.dlsi.ua.es/`, where more information about the acquisition and representation of the data is detailed.

3. MULTIMODAL CLASSIFICATION

This section provides a classification experiment over the data described previously. Two independent classifiers are proposed that exploit each of the modalities presented by the data. Eventually, a late-fusion classifier that combines the two previous ones will be considered.

Taking into account the features of our case of study, an instance-based classifier was considered. Specifically, the Nearest Neighbour (NN) rule was used, as it is one of the most common and effective algorithms of this kind [3]. The choice is justified by the fact that it is specially suitable for interactive scenarios like the one found in our task: it is naturally adaptive, as the simple addition of new prototypes to the training set is sufficient (no retraining is needed) for incremental learning from user feedback. The size of the dataset can be controlled by distance-based data reduction algorithms [7] and its computation time can be improved by using fast similarity search techniques [17].

Decisions given by NN classifiers can be mapped onto probabilities, which are needed for the late fusion classifiers. Let $\mathcal{X}$ be the input space, in which a pairwise distance $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is defined. Let $\mathcal{Y}$ be the set of labels considered in the classification task. Finally, let T denote the training set of labelled samples $\{(x_i, y_i) : x_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_{i=1}^{|T|}$.

Let us now assume that we want to know the posterior probability of each class $y \in \mathcal{Y}$ for the input point $x \in \mathcal{X}$ ($P(y|x)$) following the NN rule. A common estimation makes use of the following equations [4]:

$$p(y|x) = \frac{1}{\min_{(x',y') \in T : y' = y} d(x, x') + \epsilon} \quad (1)$$

$$P(y|x) = \frac{p(y|x)}{\sum_{y' \in \mathcal{Y}} p(y'|x)}, \quad (2)$$

where ϵ is a negligible value used to avoid infinity calculations. That is, the probability of each class is defined as the inverse of the distance to the nearest sample of that class in the training set. Note that the second term is used to ensure that the sum over the probability of each class is 1. Finally, the decision $\hat{y}$ of the classifier for an input x is given by a *maximum a posteriori* criterion:

$$\hat{y} = \arg\max_{y} P(y|x) \quad (3)$$

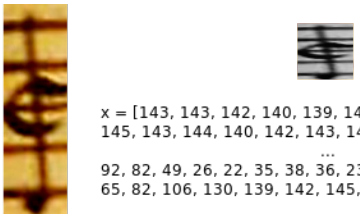


Figure 3. Offline modality of a *cut time* symbol for classification: feature vector containing the greyscale value of each position of the rescaled image.

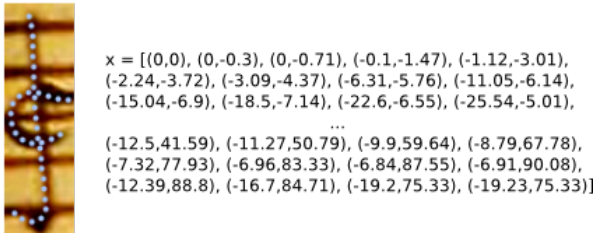


Figure 4. Online modality of a *cut time* symbol for classification: sequence of coordinates indicating the path followed by the e-pen during the tracing process.

3.1 Offline classifier

The offline classifier takes the image of a symbol as input. To simplify the data, the images are converted to greyscale. Then, since they can be of different sizes, a fixed resizing process is performed, in the same way that can be found in other works, like that of Rebelo et al. [11]. At the end, each image is represented by an integer-valued feature vector of equal length that stores the greyscale value of each pixel (see Figure 3). Over this data, Euclidean distance can be used for the NN classifier. A preliminary experimentation fixed the size of the images to 30×30 (900 features), although the values within the configurations considered did not vary considerably.

3.2 Online classifier

In the online modality, the input is a series of 2D points that indicates the path followed by the pen (see Figure 4). It takes advantage of the local information, expecting that a particular symbol follows similar paths. The information contained in this modality provides a new perspective on the recognition and it does not overlap with the nature of the offline recognition.

The digital surface collects the strokes at a fixed sampling rate so that each one may contain a variable number of points. However, several distance functions can be applied to this kind of data. Those considered in this work are the following:

- Dynamic Time Warping (DTW) [15]: a technique for measuring the dissimilarity between two time signals which may be of different duration.
- Edit Distance with Freeman Chain Code (FCC): the sequence of points representing a stroke is converted

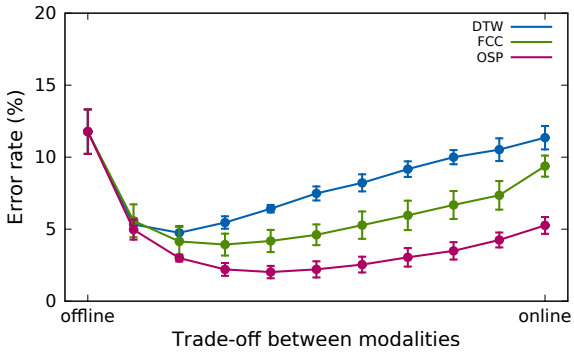


Figure 5. Average results with respect to the weight (α) given to each modality for the configurations considered, from *offline* ($\alpha = 0$) to *online* ($\alpha = 1$).

into a string using a codification based on Freeman Chain Code [6]. Then, a Edit Distance [8] can be applied to measure distance.

- Edit Distance for Ordered Set of Points (OSP) [13]: an extension of the Edit Distance for its use over ordered sequences of points, such those collected by the e-pen.

3.3 Late-fusion classifier

A straightforward late fusion has been used here. The idea is to combine linearly the decisions taken by the two base classifiers. That is, probabilities of individual classifiers are combined by a weighted average:

$$P_{\text{fusion}}(y|x) = \alpha \cdot P_{\text{on}}(y|x) + (1 - \alpha) \cdot P_{\text{off}}(y|x) \quad (4)$$

where P_{off} and P_{on} denote the probabilities obtained by offline and online classifiers, respectively. A parameter $\alpha \in [0, 1]$ is established to tune the relevance given to each modality. We will consider several values of α ranging from 0 to 1 during experimentation.

4. EXPERIMENTATION

Experimentation followed a 10-fold cross-validation scheme. The independent folds were randomly created with the sole constraint of having the same number of samples per class (where possible) in each of them. All the dissimilarities described in Section 3 for the online classifier will be tested.

Table 2 illustrates the error rate (%) achieved with respect to α for this experiment. Note that $\alpha = 0$ column yields the results of the offline classifier as well as $\alpha = 1$ is equal to the online classifier. A summary of the average results is also illustrated in Figure 5.

An initial remark to begin with is that the worst results of the late-fusion classifiers are achieved when each is modality is used separately, with an average error of 11.77 for the offline modality and of 11.35, 9.38 and 5.26 for DTW, FCC and OSP, respectively. Not surprisingly, best results are those that combine both natures of the data, satisfying the hypothesis that two signals are better than one.

α	DTW	FCC	OSP
0.0	11.8 ± 1.5	11.8 ± 1.5	11.8 ± 1.5
0.1	5.3 ± 0.4	5.5 ± 1.1	4.9 ± 0.7
0.2	$\mathbf{4.7 \pm 0.4}$	4.1 ± 1.0	3.0 ± 0.2
0.3	5.4 ± 0.4	$\mathbf{3.9 \pm 0.8}$	2.2 ± 0.4
0.4	6.4 ± 0.3	4.1 ± 0.7	$\mathbf{2.0 \pm 0.4}$
0.5	7.4 ± 0.5	4.6 ± 0.7	2.2 ± 0.5
0.6	8.2 ± 0.6	5.2 ± 0.9	2.5 ± 0.5
0.7	9.1 ± 0.5	5.9 ± 1.0	3.0 ± 0.6
0.8	9.8 ± 0.5	6.6 ± 0.9	3.4 ± 0.6
0.9	10.5 ± 0.8	7.3 ± 0.9	4.2 ± 0.5
1.0	11.3 ± 0.8	9.3 ± 0.7	5.2 ± 0.5

Table 2. Error rate (average $\pm$ std. deviation) obtained for a 10-fold cross validation experiment with respect to the value used for tuning the weight given to each modality (α) and the distances for the online modality (DTW, FCC and OSP). Bold values represent the best average result for each configuration considered.

Results also report that the tuning of α is indeed relevant since it makes the error vary noticeably. An interesting point to mention is that, although the online modality is more accurate than the offline one by itself, the best tuning in each configuration always gives more importance to the latter. This might be caused by the lower variability in the writing style of the original scribes.

The best results, on average, are reported by the late-fusion classifier considering OSP distance for the online modality, with an $\alpha = 0.4$. In such case, just 2 % of error rate is obtained, which means that the interaction is well understood by the system in most of the cases. Note that a more comprehensive search of the best α may lead to a better performance —for instance, in the range $(0.3, 0.5)$— but the improvement is not expected to be significant.

Although the results report a fair accuracy, the use of semantic music models is expected to avoid some of these mistakes by using contextual information. Therefore, a nearly optimal performance could be obtained during the interaction with the user.

5. CONCLUSIONS

This paper presents a new approach to interact with musical documents, based on the use of an electronic pen. Our framework assumes that the user traces each musical symbol of the score, and the system receives a *multimodal* input accordingly: the sequence of coordinates indicating the trajectory of the e-pen (online mode) and the underlying image of the score itself (offline mode).

An interface based on this idea could be used in a number of contexts related to interact with music scores in a more intuitive way for the user. For instance, to amend OMR errors, to acquire training data in the early stages of the development, or even as a part of a complete manual

transcription system.

This framework has been applied to a music archive of Spanish music from the 16th to 18th centuries, handwritten in white mensural, with the objective of obtaining data for our experiments. The result of processing this collection has been described and made available for research purposes.

Experimentation with this dataset is presented, considering several classifiers. The overall analysis of this experiments is that it is worth to consider both modalities in the classification process, as accuracy is noticeably improved with a combination of them than that achieved by each separately.

As a future line of work, the reported analysis will be used to build a whole *computer-aided* system, in which the user interacts with the system by means of an electronic pen to digitize music content. Since the late-fusion classifier is close to its optimal performance, it seems to be more interesting to consider the development of semantic models that can amend misclassifications by using contextual information (*e.g.*, a score starts with a clef). In addition, further effort is to be devoted to visualization and user interfaces.

6. ACKNOWLEDGEMENTS

This work has been supported by the Spanish Ministerio de Educación, Cultura y Deporte through a FPU fellowship (Ref. AP2012–0939) and the Spanish Ministerio de Economía y Competitividad through project TIMuL (No. TIN2013-48152-C2-1-R, supported by UE FEDER funds).

Authors would like to thank Ichiro Fujinaga for inspiring the title of this paper, and Antonio Ezquerro and Luis Antonio González (Instituci Milà i Fontanals, Spanish National Research Council), who allowed the utilization of the music sources for research purposes.

7. REFERENCES

[1] Vicent Alabau, Carlos D. Martínez-Hinarejos, Verónica Romero, and Antonio L. Lagarda. An iterative multimodal framework for the transcription of handwritten historical documents. *Pattern Recognition Letters*, 35:195–203, 2014.

[2] Jesús Andrés-Ferrer, Verónica Romero, and Alberto Sanchis. *Multimodal Interactive Pattern Recognition and Applications*, chapter General Framework. Springer, 1st edition edition, 2011.

[3] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, January 1967.

[4] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, New York, NY, 2nd edition, 2001.

[5] Antonio Ezquerro Esteban, editor. *Música de la Catedral de Barcelona a la Biblioteca de Catalunya*. Biblioteca de Catalunya, Barcelona, 2001.

[6] Herbert Freeman. On the encoding of arbitrary geometric configurations. *IRE Transactions on Electronic Computers*, EC-10(2):260–268, 1961.

[7] Salvador García, Julián Luengo, and Francisco Herrera. *Data Preprocessing in Data Mining*, volume 72 of *Intelligent Systems Reference Library*. Springer, 2015.

[8] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, 1966.

[9] João Rogério Caldas Pinto, Pedro Vieira, and João Miguel da Costa Sousa. A new graph-like classification method applied to ancient handwritten musical symbols. *IJDAR*, 6(1):10–22, 2003.

[10] Laurent Pugin. Optical music recognition of early typographic prints using hidden markov models. In *ISMIR 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 8-12 October 2006, Proceedings*, pages 53–56, 2006.

[11] A. Rebelo, G. Capela, and Jaime S. Cardoso. Optical recognition of music symbols. *International Journal on Document Analysis and Recognition (IJDAR)*, 13(1):19–31, 2010.

[12] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, André R. S. Marçal, Carlos Guedes, and Jaime S. Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, 2012.

[13] Juan R. Rico-Juan and Jose M. Iñesta. Edit distance for ordered vector sets: A case of study. In *Structural, Syntactic, and Statistical Pattern Recognition*, volume 4109 of *Lecture Notes in Computer Science*, pages 200–207. Springer Berlin Heidelberg, 2006.

[14] V. Romero and J. Andreu Sanchez. Human Evaluation of the Transcription Process of a Marriage License Book. In *12th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1255–1259, Aug 2013.

[15] Hiroaki Sakoe and Seibi Chiba. Readings in Speech Recognition. In Alex Waibel and Kai-Fu Lee, editors, *Readings in Speech Recognition*, chapter Dynamic Programming Algorithm Optimization for Spoken Word Recognition, pages 159–165. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[16] Alejandro Hector Toselli, Verónica Romero, Moisés Pastor, and Enrique Vidal. Multimodal interactive transcription of text images. *Pattern Recognition*, 43(5):1814–1825, 2010.

[17] Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal, and Michal Batko. *Similarity Search - The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Kluwer, 2006.

Oral Session 4

Musicologies

ANALYZING MEASURE ANNOTATIONS FOR WESTERN CLASSICAL MUSIC RECORDINGS

Christof Weiß[1] **Vlora Arifi-Müller**[1] **Thomas Prätzlich**[1]
Rainer Kleinertz[2] **Meinard Müller**[1]
[1] International Audio Laboratories Erlangen, Germany
[2] Institut für Musikwissenschaft, Saarland University, Germany
{christof.weiss, meinard.mueller}@audiolabs-erlangen.de

ABSTRACT

This paper approaches the problem of annotating measure positions in Western classical music recordings. Such annotations can be useful for navigation, segmentation, and cross-version analysis of music in different types of representations. In a case study based on Wagner's opera "Die Walküre", we analyze two types of annotations. First, we report on an experiment where several human listeners generated annotations in a manual fashion. Second, we examine computer-generated annotations which were obtained by using score-to-audio alignment techniques. As one main contribution of this paper, we discuss the inconsistencies of the different annotations and study possible musical reasons for deviations. As another contribution, we propose a kernel-based method for automatically estimating confidences of the computed annotations which may serve as a first step towards improving the quality of this automatic method.

1. INTRODUCTION

Archives of Western classical music often comprise documents of various types and formats including text, symbolic data, audio, image, and video. Dealing with an opera, for example, one may have different versions of musical scores, libretti, and audio recordings. When exploring and analyzing the various kinds of information sources, the establishment of semantic relationships across the different music representations becomes an important issue. For a recorded performance, time positions are typically indicated in terms of *physical units* such as seconds. On the other hand, the musical score typically specifies time positions using *musical units* such as measures. Knowing the measure positions in a given music recording not only simplifies access and navigation [15, 19] but also allows for transferring annotations from the sheet music to the audio domain (and vice versa) [16]. Furthermore, a measure-based alignment of several performances enables cross-performance analysis tasks [12, 13].

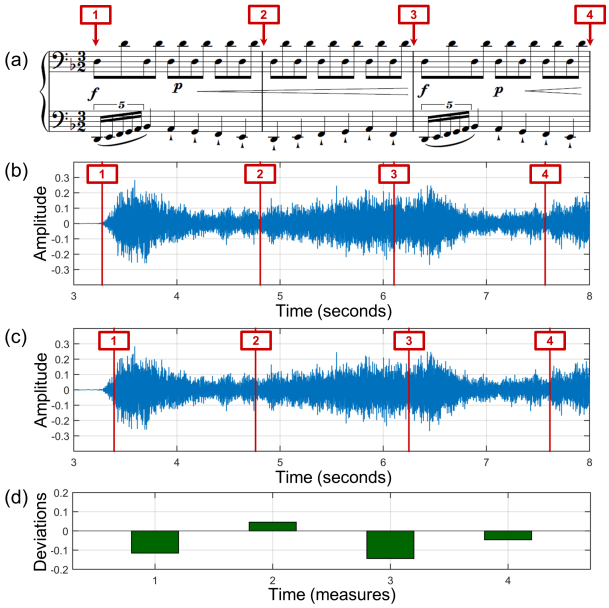


Figure 1. Two measure annotations for a Karajan performance of Wagner's opera "Die Walküre", first act, measures 1–3. **(a)** Piano reduction of the score. **(b)** Annotation A1. **(c)** Annotation A2. **(d)** Deviation of A1 from A2.

In this paper, we report on a case study based on the opera cycle "Der Ring des Nibelungen" WWV 86 by Richard Wagner where we consider the first act of the second opera "Die Walküre" (The Valkyrie). For this challenging scenario, we examine different types of measure annotations—either supplied by human annotators (*manual annotations*) or generated automatically using synchronization techniques (*computed annotations*). Figure 1 illustrates this scenario. Surprisingly, even the manual annotations (not to speak of the annotations obtained by automated methods) often deviate significantly from each other. As one contribution, we analyze such inconsistencies and discuss their implications for subsequent music analysis tasks. After describing the dataset and the annotation process (Section 2), we first analyze the properties of manual annotations stemming from different human annotators (Section 3). Subsequently, we evaluate computer-generated annotations that are derived from score-to-audio

© Christof Weiß, Vlora Arifi-Müller, Thomas Prätzlich, Rainer Kleinertz, Meinard Müller. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Christof Weiß, Vlora Arifi-Müller, Thomas Prätzlich, Rainer Kleinertz, Meinard Müller. "Analyzing Measure Annotations for Western Classical Music Recordings", 17th International Society for Music Information Retrieval Conference, 2016.

synchronization results (Section 4). Hereby, we examine correlations between inter-human inconsistencies and errors of the automated approach and identify musical reasons for the deviations. Finally, we propose a method to derive confidence values for the computed annotations from the synchronization procedure (Section 5).

2. DATA AND ANNOTATIONS

2.1 Music Scenario

Wagner's four-opera cycle "Der Ring des Nibelungen" WWV 86 is an exceptionally long work of about 14–15 hours duration. Because of its large scale and complex structure, it constitutes a challenging scenario for computer-assisted analysis methods [16]. In this paper, we consider the first act of "Die Walküre" WWV 86 B for a first case study. For analyzing such music, several issues are relevant. Work-related aspects such as motifs, instrumentation, chords, or coarse-scale harmonic structures as well as performance-related phenomena such as tempo, timbre, or loudness play a role. Furthermore, the relation between such aspects and the libretto may be of interest.

For their analyses, musicologists traditionally use the musical score which corresponds to the musical idea of the composer. Scores or piano reductions provide a compact overview of the musical content and are particularly suitable for harmony analysis. For performance-related aspects of the music, we need to analyze audio recordings. For this paper, we consider both types of data. Regarding symbolic data, we use a piano-reduced version of the score by Kleinmichel. [1] The sheet music is processed with OMR software (*Avid*™ *PhotoScore*) followed by manual correction using notation software. This piano-reduced score constitutes a kind of "harmonic excerpt" of the music. From the notation software, we export symbolic data types such as MIDI or MusicXML. For the audio domain, we consider an interpretation by Karajan with the Berlin Philharmonic (1966 Deutsche Grammophon, Berlin). [2] The duration of the first act in this recording is 67 minutes.

The types of music representations differ in the way how time and tempo are encoded. Audio recordings have a physical time axis usually given in seconds. In contrast, scores exhibit a musical time axis given in measures or beats. The physical length of a musical unit—such as a measure—depends on the *tempo* and the *time signature*. In operas, both tempo and time signature change frequently. To establish relations between the representations, we need to interconnect their time axes. One way to do this is to specify the measure positions in the audio recordings.

Such *measure annotations* may fulfill several purposes. First, they facilitate navigation and segmentation using musically meaningful units such as motifs, passages, or scenes [19]. Second, they enable the transfer of semantic annotations or analysis results from one domain to the other [16]. Third, *cross-version analysis* specifically uses the relation between different performances in order to stabilize analysis results [12].

[1] This piano reduction is publicly avaible on http://www.imslp.org.

[2] In our experiments, we also consider further performances yielding similar results as the ones reported for the Karajan performance.

2.2 Manual Annotations

To obtain measure annotations for our opera, we first consider a manual approach where five students with a strong practical experience in Western classical music annotated the measure positions for the full Karajan recording. We refer to these annotators as A1, . . . , A5. While following a vocal score [20] used as reference, the annotators listened to the recording and marked the measure positions using the public software *Sonic Visualizer* [2]. After finishing a certain passage, the annotators corrected erroneous or inaccurate measure positions. The length of these passages, the tolerance of errors, and the overall duration of the annotation process differed between the annotators. Roughly three hours were necessary to annotate one hour of music.

Beyond that, the annotators added comments to specify ambiguous measure positions. As musical reasons for such ambiguities, they mentioned tempo changes, fermatas, tied notes over barlines, or very fast passages. Furthermore, they reported performance-specific problems such as asynchronicities between orchestra and singers or masking of onsets through prominent other sounds. For some of these critical passages, the annotators reported problems arising from the use of a piano reduction instead of the full score.

Due to these (and other) difficulties, one can find significant deviations between the different annotations (see Figure 1 for an illustration). One goal of this paper is to analyze the annotation consistency and to uncover possible problems in the annotation process (Section 3).

2.3 Computed Annotations

The manual generation of measure annotations for music recordings is a time-consuming and tedious procedure. To automate this process, different strategies are possible. For example, one could start with a beat tracking algorithm and try to find the downbeats which yields the measure positions [17]. Moreover, beat information may help to obtain musically meaningful features [6]. For classical music, however, beat tracking is often not reliable [9, 10]. In [4], Degara *et al.* have automatically estimated the reliablity of a beat tracker.

In this paper, we follow another strategy based on synchronization techniques. The general goal of music synchronization (or audio-to-score alignment) is to establish an alignment between *musically corresponding time positions* in different representations of the same piece [1, 3, 5, 11]. Based on a symbolic score representation where measure positions are given explicitly, we use the computed alignment to transfer these positions to the audio recording.

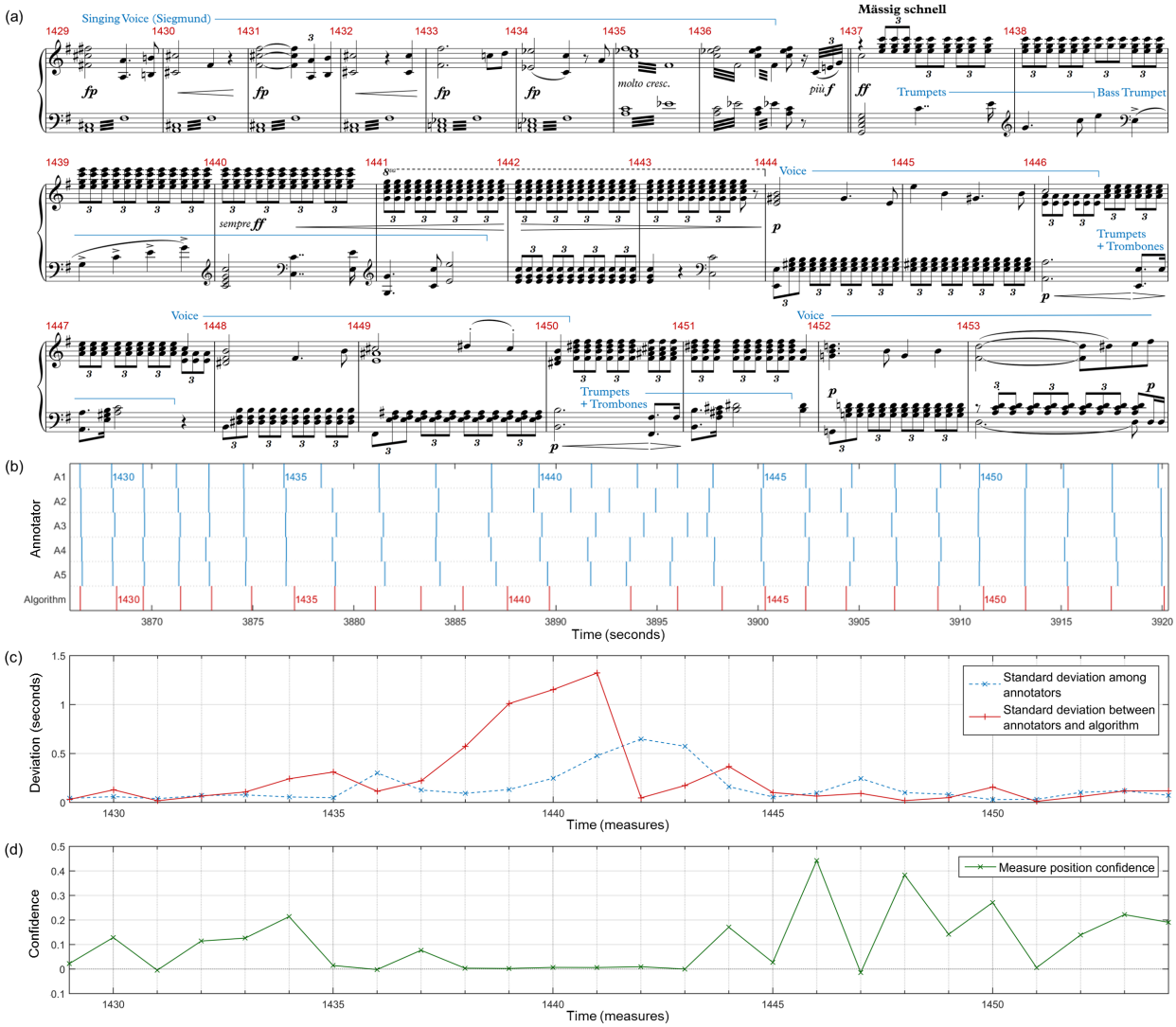


Figure 2. Measure annotations for a Karajan performance of R. Wagner's opera "Die Walküre", Act 1, Measures 1429–1453. **(a)** Piano reduction of the score (Kleinmichel). **(b)** Measure positions from manual (blue) and computed annotations (red). **(c)** Standard deviations among the manual annotations (blue dashed line) and between the mean manual annotation versus the algorithm's annotation (red solid line). **(d)** Measure position confidences derived from the similarity matrix.

In the following experiments, we use an alignment method based on Dynamic Time Warping (DTW) [15]. First, the audio recording and the symbolic score are transformed into a common feature representation. For this, we use CENS [15] features—a variant of chroma features which are well-suited for capturing the coarse harmonic progression of the music—combined with features capturing note onset information (see [7] for details). Using a suitable cost measure, DTW is applied to compute a cost-minimizing alignment between the two different versions [7]. As our opera recording is long (67 minutes), memory requirements and run time become an issue. To this end, we use a memory-efficient multiscale variant of DTW that allows for explicitly controlling the memory requirements [18]. The main idea of this DTW variant is to use rectangular constraint regions on which local alignments are computed independently using DTW. Macrae and Dixon [14] have used a similiar approach.

3. ANALYSIS OF MANUAL ANNOTATIONS

In our analysis, we first consider the manual annotations (see Section 2.2). As an example, Figure 2 shows a passage of "Die Walküre", Act 1. In Figure 2b, we plot the physical time position of the measures' beginning (horizontal axis) for the different annotators (vertical axis). At the beginning of this example, the annotators more or less agree. Sometimes, a single annotator slightly deviates from the others. As an example, annotator A1 sets an early position for measure 1436 compared to A2, . . . , A5. To quantify the overall disagreement for a specific measure, we calculate the standard deviation over the physical time position by all annotators. The blue dashed curve in Figure 2c shows this quantity for our exemplary passage. For example, one can see a small increase in measure 1436.

From measure 1440 on, the standard deviation consider-

ably increases over several measures. Looking at the annotations, we see that this disagreement does not stem from a single annotator but results from a substantial disagreement between all annotators. Annotator A2 exhibits the largest deviations by specifying the positions for the measures 1441–1443 much earlier than the other annotators. The confusion ends at measure 1445 for which the annotators seem to agree almost perfectly.

Looking at the score, we find possible hints for these deviations. For both measures 1436 and 1438–1443, the chord, voicing and instrumentation do not change with respect to the previous measure. In the measures 1437–1441, however, a prominent trumpet melody is present which probably served as an orientation for the annotators. In accordance with this assumption, we find the highest disagreement for the measures 1442 and 1443 where this melody has ended and only a constant and dense instrumentation of the C major chord is played. [3] Nevertheless, this observation does not explain the high deviations in measures 1440–1441. Listening to the recording, we noticed that the bass trumpet melody (left hand in m. 1439–1441) is covered by the orchestra and thus, practically not audible in this recording. Three of the annotators marked this problem as "masking". In measure 1444, a remarkable chord change (C major to E major) and a new melody in the tenor yield an orientation point where all annotators agree again.

By means of this examplary passage, we have already shown two important cues that may help humans to find measure boundaries: (1) Distinct harmonic changes and (2) salient melodic lines that can be followed easily. As for the second class, singing melodies seem to be more helpful than instrumental lines in the orchestra which are often superimposed by other instruments. For measures 1441 and 1445, we similarly find the present chord and instrumentation continued together with a melodic line. In the case of the trumpet line (m. 1441), the agreement is low. In contrast, the tenor melody (m. 1445) leads to high agreement. On the one hand, percussive speech components such as consonants and fricatives yield good temporal cues. On the other hand, solo voices play an important role in operas and often stand out musically as well as acoustically.

We have seen that humans may disagree substantially in their measure annotations. We now want to quantify such deviations on the basis of the full opera act. Since we do not have a "correct" ground truth annotation, we calculate for each measure the mean position across all manual annotations. Then, we calculate the offset of each annotation with respect to this mean position and plot a histogram over these offset values for all measures of the act. Figure 3 shows the resulting distributions for all five human annotators. In these plots, we observe typical offsets of about 0.1 seconds in both directions (measures annotated too early and too late). Deviations larger than 0.2 seconds are rare. Beyond this, we notice some systematic offsets towards one direction. For example, the distribution of A1 has its maximal bin at -0.04 seconds. For annotators A2, A3, and A5, the maximal bin is centered at zero but positive deviations are more frequent than negative ones. Overall, systematic offsets seem to be rather small. For single measures, deviations up to 0.2 seconds occur in both directions. In the following, we use the mean positions of all annotators as reference for evaluating our automatic approach.

[3] The full score shows 8th triplets (winds) and 16th arpeggios (strings). Our reduction focuses on the triplets that are hard to perceive in the audio.

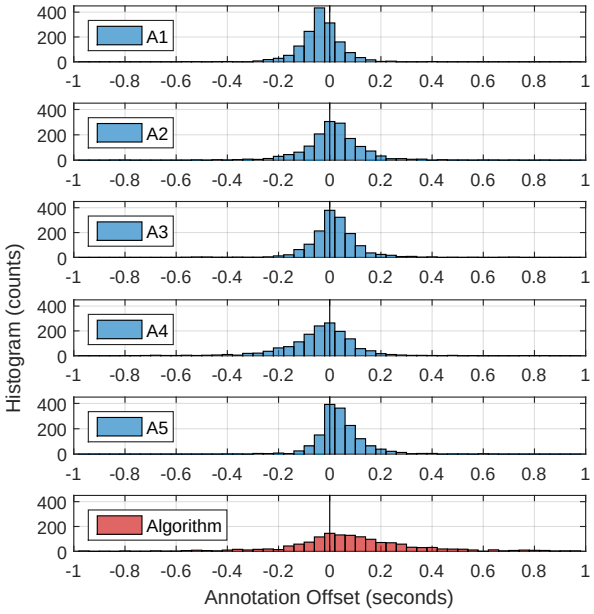


Figure 3. Histograms of annotation offsets for the individual annotations (full act). The deviations refer to the mean position of all annotations for the respective measure (labelled as zero). Positive offset values indicate "too late", negative values indicate "too early" positions compared to the mean position. The lowest plot refers to the computed annotation generated by our synchronization algorithm.

4. ANALYSIS OF COMPUTED ANNOTATIONS

In this section, we analyze the computed annotations with respect to the manual annotations. Let us consider Figure 2 again. In the lowest row of Figure 2b (red), we show the measure positions as generated by the algorithm. The red curve in Figure 2c quantifies the deviation between the algorithm's and the average manual measure position. For the first measures 1429–1435, the computed positions seem to coincide with the manual annotations. Similarly, the annotations for the final measures 1445–1454 more or less agree. For the middle section, we find a different situation. In measures 1436–1441, the algorithm strongly deviates from the human annotators. For example, the position of measure 1441 is close to the human's position of measure 1440—a deviation of more than two seconds. Interestingly, the algorithm then produces a very long measure 1441 leading to a good coincidence with the manual annotations in measure 1442 again.

Looking at the score, we may find an explanation for this behaviour. The measures 1437–1443 (where the algorithm

strongly deviates) are harmonically restricted to a single C major triad. The listeners may have used the trumpet melody as cue to follow the rhythm. However, the trumpet only plays pitches from the C major chord which is present in the accompaniment. For chroma features which are the basis of the synchronization approach, these pitches contribute to the same chroma entries. For this reason, the chroma-based feature representation does not yield suitable cues for the matching process. For measures with clear harmonic change such as 1446 or 1448, we mostly see high agreement. Interestingly, one finds a small deviation for measure 1442 which is the most ambiguous measure among the human annotators. Here, we have to carefully interpret the figure since we use the mean manual annotations as reference. For measures with strong human disagreement, this mean position is not a reliable reference and, thus, the small deviation may be rather accidental.

In contrast, the relatively large deviation for measure 1444 seems surprising since we have a prominent harmonic change here (C major to E major chord). The situation becomes clearer when we listen to the audio recording. Actually, the onset of the singing voice (note B4) in measure 1444 is too early in this interpretation (by almost a quarter note) with respect to the chord change of the orchestra. The human annotators consistently followed the voice whereas the chroma-based synchronization method relies on the harmonic content dominated by the orchestra.

Let us consider Figure 3 again. The lowest plot (red) shows a histogram over the annotation offsets of the synchronization algorithm with respect to the mean manual annotation. This distribution is much flatter than the human ones. Large deviations such as the one in measure 1439 are more frequent for the automated approach. Furthermore, there is a remarkable systematic offset towards late measure positions. The majority of the annotations lies within a window of ± 0.3 seconds around the humans' mean position. For passages with strong disagreement, the algorithm finds back after a few measures—as for our example in Figure 2. Overall, we conclude that the automated approach does not reach the reliablity of the manual annotations but yields reasonable results for most measure positions.

5. CONFIDENCES FOR COMPUTED ANNOTATIONS

As we have seen from our previous analysis, there is a need for improving automated procedures. As a first step towards an improvement, we now introduce an approach for generating confidence values for the computed measure positions. Recall that the core of our method is a synchronization algorithm (see Section 2.3). Our idea is to use the local reliablity of the synchronization for estimating the confidence of the computed measure positions. Since the synchronization is based on chroma features, the change in harmony influences the quality of the alignment. Having similar chords in neighbouring measures usually results in similar chroma vectors. This often leads to situations

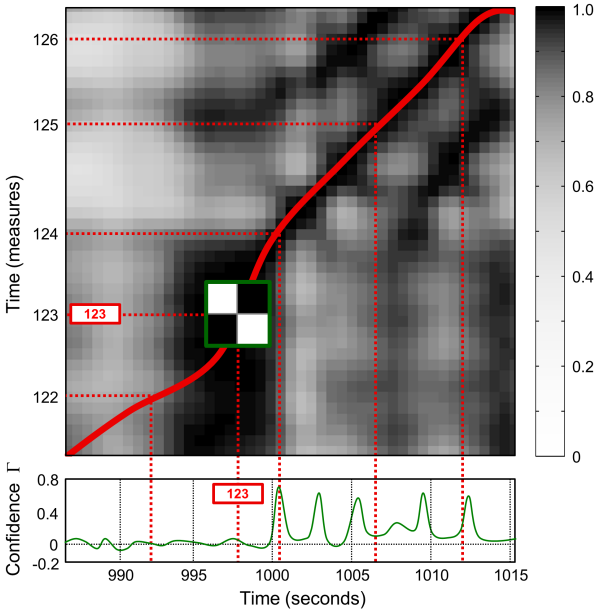


Figure 4. Estimation of measure position confidences (schematically). In the similarity matrix, we shift a checkerboard kernel along the warping path and calculate a confidence value for each measure position.

where the measure position is ambiguous. In contrast, measure boundaries that coincide with harmonic changes often lead to reliable measure annotations.

On the basis of this observation, we propose a novelty-based confidence measure. To compute the novelty score, we transfer our music recording to a sequence of chroma features $X = (x_1, \ldots, x_N)$ with a resolution of 10 Hz. Similarly, we compute a feature representation $Y = (y_1, \ldots, y_M)$ of the symbolic data (score). Then, we derive a similarity matrix $\mathcal{S} \in \mathbb{R}^{N \times M}$ from the two feature sequences using a cosine measure to compare feature vectors. The automated synchronization procedure (see Section 2.3) yields an alignment in terms of a warping path which we project on the given feature resolution. To estimate local confidence values for this warping path, we adapt an idea by Foote [8] who computes a novelty function by shifting a checkerboard kernel $\mathcal{K} \in \mathbb{R}^{K \times K}$ along the diagonal of a self-similarity matrix (SSM) and locally measures the similarity between $\mathcal{K}$ and the underlying region of the SSM. Here, we compute a novelty function by shifting the kernel along the *warping path* and locally measuring the similarity between $\mathcal{K}$ and the region of our similarity matrix $\mathcal{S}$. In our experiments, we use a kernel $\mathcal{K}$ of size $K = 10$ features (one second of the recording).

With this procedure and a subsequent normalization step, we obtain a curve $\Gamma : \{1, 2, \ldots, N\} \to [-1, 1]$ which measures the novelty of the local chroma vectors along the warping path. For a feature index $n \in \{1, \ldots, N\}$, a value of $\Gamma(n) \approx 1$ indicates high similarity between the local region of $\mathcal{S}$ and the structure of $\mathcal{K}$. Intuitively, we then expect a structural change in the features' properties. Musically spoken, $\Gamma(n) \approx 1$ implies clear change in local

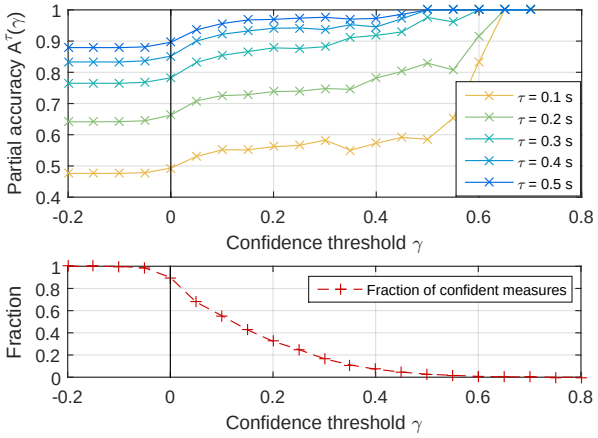


Figure 5. Confidence-dependent accuracy values for the entire act (1523 measures). For fixed tolerance values τ, the upper plot shows the partial accuracies $A^\tau(\gamma)$ of the computed measure annotations over the confidence threshold γ. The lower plot shows the fraction of measures under consideration.

harmony. At such positions, we expect the synchronization algorithm to work accurately. For $\Gamma(n) \approx 0$, there is little change in the features, which points to a harmonically homogenous situation in a neighbourhood of n. Finally, we evaluate Γ on those time instances that correspond to the measure positions. Figure 4 outlines this principle.

For quantitatively evaluating the computed annotations, we consider the mean of all five manual annotations as a reference. Using a tolerance parameter $\tau \in \mathbb{R}$, we regard a computed position to be correct if it lies within an interval of length 2τ centered at the reference position. Let $\mathcal{M} := \{1, 2, \ldots, L\}$ be the set of all measures and A^τ the fraction of correctly annotated measures with respect to τ. For $\tau = 0.2$ s, for example, a fraction $A^\tau = 62.5\,\%$ of the measures in $\mathcal{M}$ lies within this interval. We further define a subset $\mathcal{M}_\gamma := \{m \in \mathcal{M} \,|\, \Gamma(m) \geq \gamma\}$ which only includes measures with a confidence above the threshold $\gamma \in \mathbb{R}$. Additionally, we define a partial accuracy $A^\tau(\gamma)$ which only refers to the measures in $\mathcal{M}_\gamma$.

Figure 5 shows the results for the full first act of "Die Walküre". In the upper part, we show the curve $\gamma \to A^\tau(\gamma)$ for fixed τ. The lower plot displays the curve $\gamma \to |\mathcal{M}_\gamma|/|\mathcal{M}|$. In general, the accuracy increases for larger confidence thresholds γ. For $\tau = 0.3$ s (cyan curve), for example, $A^\tau(\gamma)$ improves from 78.2 % (for $\gamma = 0$) to 87.8 % (for $\gamma = 0.2$). At the same time, the fraction of considered measures decreases. To obtain accuracies $A^\tau(\gamma) > 90\,\%$, we end up evaluating less than 10 % of the measures (for $\tau = 0.3$ s). A good tradeoff seems to be at $\gamma = 0.1$ where we get up to 10 % increase of $A^\tau(\gamma)$ while still having half of the measure annotations included. These more "confident" measure positions may serve as a kind of anchor points for improving the quality of the automated approach. For example, one could replace the measure position with low consistency using linear interpolation or a smoothed tempo curve.

Finally, let us consider our running example again. Figure 2d shows the confidence values for this passage. We see that for the measures 1437–1443, the confidences are low due to the harmonic homongeneity (C major chord over seven measures). In contrast, we find high confidences for distinct chord changes as in measure 1444 (C major → E major), measure 1446 (E major → A minor), or measure 1448 (A minor → B major). Let us compare these values to the corresponding annotation consistency (red line in Figure 2c). For some of the high-confident measures (1446, 1448–1450, 1452–1454), the measure position is consistent with manual annotations. The situation is different for measure 1444. Here, our confidence value is high but the position deviates from the manual annotations. Remembering the audio properties discussed in Section 4, we can understand this behaviour. Here, the human annotators consistently follow the entry of the voice which is too early compared to the orchestra's onset. Thus, the high confidence indicates a good measure position which is correct with respect to *harmony*. The deviation from the manual annotations arises from the asynchronicity.

In general, we can only draw conclusions for measures with high confidence $\Gamma(m)$. A low confidence does not necessarily indicate a bad estimate of the measure position. In Figure 2, measures 1431 and 1445 are examples for such a behaviour, where we find low Γ-values but high consistency of measure positions with manual annotations.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we analyzed different types of measure annotations for an opera recording. For manual annotations generated by different human annotators, we identified musical challenges which can lead to inconsistencies among the annotators. In contrast, harmonic changes and melodic lines seem to be important cues for the listeners to accurately locate measure boundaries. Furthermore, we analyzed measure annotations generated by a computer using score-to-audio alignment. This approach provides useful results but is less accurate than manual annotations. In particular, harmonic homogeneity can be problematic for chroma-based approaches. Based on this observation, we automatically estimate the confidence of the computed annotations. To this end, we shift a checkerboard kernel along the warping path. The resulting confidence values seem to be useful for identifying reliable measure position. Thus, they may serve as a first step towards improving synchronization-based annotation strategies.

7. ACKNOWLEDGMENTS

We thank all students involved in the annotation work. This work was supported by the German Research Foundation (DFG MU 2686/7-1, DFG KL 864/4-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg FAU and Fraunhofer Institute for Integrated Circuits IIS.

8. REFERENCES

[1] Andreas Arzt and Gerhard Widmer. Real-time music tracking using multiple performances as a reference. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 357–363, Málaga, Spain, 2015.

[2] Chris Cannam, Christian Landone, Mark Sandler, and Juan Pablo Bello. The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 324–327, Victoria, Canada, 2006.

[3] Roger B. Dannenberg and Ning Hu. Polyphonic audio matching for score following and intelligent audio editors. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 27–34, San Francisco, USA, 2003.

[4] Norberto Degara, Enrique Argones Rúa, Antonio Pena, Soledad Torres-Guijarro, Matthew E. P. Davies, and Mark D. Plumbley. Reliability-informed beat tracking of musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):290–301, 2012.

[5] Simon Dixon and Gerhard Widmer. MATCH: A music alignment tool chest. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, London, UK, 2005.

[6] Daniel P.W. Ellis, Courtenay V. Cotton, and Michael I. Mandel. Cross-correlation of beat-synchronous representations for music similarity. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 57–60, Las Vegas, USA, 2008.

[7] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[8] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 452–455, New York, USA, 2000.

[9] Peter Grosche, Meinard Müller, and Craig Stuart Sapp. What makes beat tracking difficult? A case study on Chopin Mazurkas. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 649–654, Utrecht, The Netherlands, 2010.

[10] André Holzapfel, Matthew E.P. Davies, José R. Zapata, João Oliveira, and Fabien Gouyon. On the automatic identification of difficult examples for beat tracking: towards building new evaluation datasets. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 89–92, Kyoto, Japan, 2012.

[11] Cyril Joder, Slim Essid, and Gaël Richard. A conditional random field framework for robust and scalable audio-to-score matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(8):2385–2397, 2011.

[12] Verena Konz, Meinard Müller, and Rainer Kleinertz. A cross-version chord labelling approach for exploring harmonic structures – a case study on Beethoven's Appassionata. *Journal of New Music Research*, pages 1–17, 2013.

[13] Cynthia C.S. Liem and Alan Hanjalic. Expressive timing from cross-performance and audio-based alignment patterns: An extended case study. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 519–524, Miami, USA, 2011.

[14] Robert Macrae and Simon Dixon. Accurate real-time windowed time warping. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 423–428, Utrecht, The Netherlands, 2010.

[15] Meinard Müller. *Fundamentals of Music Processing*. Springer Verlag, 2015.

[16] Kevin R. Page, Terhi Nurmikko-Fuller, Carolin Rindfleisch, David M. Weigl, Richard Lewis, Laurence Dreyfus, and David De Roure. A toolkit for live annotation of opera performance: Experiences capturing Wagner's ring cycle. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 211–217, Málaga, Spain, 2015.

[17] Hélène Papadopoulos and Geoffroy Peeters. Joint estimation of chords and downbeats from an audio signal. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):138–152, 2011.

[18] Thomas Prätzlich, Jonathan Driedger, and Meinard Müller. Memory-restricted multiscale dynamic time warping. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Shanghai, China, 2016.

[19] Thomas Prätzlich and Meinard Müller. Freischütz Digital: a case study for reference-based audio segmentation of operas. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 589–594, Curitiba, Brazil, 2013.

[20] Richard Wagner. *Die Walküre. Vocal score based on the complete edition*. Schott Music (ed. Egon Voss), Mainz, Germany, 2013.

INSTRUMENTAL IDIOM IN THE 16TH CENTURY: EMBELLISHMENT PATTERNS IN ARRANGEMENTS OF VOCAL MUSIC

David Lewis, Tim Crawford, Daniel Müllensiefen

Goldsmiths, University of London

{d.lewis, t.crawford, d.mullensiefen}@gold.ac.uk

ABSTRACT

Much surviving 16th-century instrumental music consists of arrangements ('intabulations') of vocal music, in tablature for solo lute. Intabulating involved deciding what to omit from a score to fit the instrument, and making it fit under the hand. Notes were usually added as embellishments to the original plain score, using idiomatic patterns, typically at cadences, but often filling simple intervals in the vocal parts with faster notes.

Here we test whether such patterns are both characteristic of lute intabulations as a class (vs original lute music) and of different genres within that class. We use patterns identified in the musicological literature to search two annotated corpora of encoded lute music using the SIA(M)ESE algorithm. Diatonic patterns occur in many chromatic forms, accidentals being added depending how the arranger applied the conventions of musica ficta. Rhythms must be applied at three different scales as notation is inconsistent across the repertory. This produced over 88,000 short melodic queries to search in two corpora totalling just over 6,000 encodings of lute pieces.

We show that our method clearly discriminates between intabulations and original music for the lute ($p < .001$); it also can distinguish sacred and secular genres within the vocal models ($p < .001$).

1. INTRODUCTION

A large proportion of surviving solo instrumental music from before the 17th century is made up of arrangements, or *intabulations*, of pre-existing vocal music, much of it printed in the 16th century. [1] Most of these are for solo lute and are notated in tablature; several collections of intabulations for keyboard instruments also exist, though they are not considered in this paper. These arrangements are rarely strict reproductions of the scores of their vocal models. Even those which aim at faithful representation of the vocal parts almost invariably contain pitch and rhythmic alterations, as well as omitted and added notes.

© David Lewis, Tim Crawford, Daniel Müllensiefen.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** David Lewis, Tim Crawford, Daniel Müllensiefen, "Instrumental idiom in the 16th century: embellishment patterns in arrangements of vocal music", 17th International Society for Music Information Retrieval Conference, 2016.

These changes come in different forms and might arise from various motivations:

- They may reflect unnotated aspects of performance practice for the vocal model;
- They may reflect attempts to make the music better match the idioms of instrumental style;
- They may result from the practical limitations of playing multiple voices on a single instrument;
- They may arise from the individual style of the intabulator, the musical genre or current fashion.

Studying the process of intabulation, then, has the potential to reveal much about several aspects of vocal and instrumental music of the time, along with the nature of individual style, by allowing us to draw attention to explicit and conscious changes to a text and to attempt to infer the motivation for those changes.

In this paper, we describe an experiment tracing the use, in two collections of encoded lute music, of typical melodic embellishment patterns identified in the musicological literature. The collections, which are broadly representative of the 16th-century lute repertory, contain a mixture of original idiomatic music composed for the lute and intabulations of vocal music, and we wanted to determine the extent to which embellishment patterns might be viewed as diagnostic features – can we use them to distinguish intabulations from music originally composed for the lute? We also carried out a further investigation into whether the occurrence of the patterns can be associated with the style of the music being arranged – were they applied differently to sacred and secular vocal music by the arrangers? These may be considered as simple proxies for more complex questions about the distinct identity of lute intabulation as a genre and are intended as a first step towards richer and larger-scale musicological studies which, for example, might compare intabulations with encoded scores of the original vocal music, or induce patterns of embellishment automatically.

2. BACKGROUND

About half of the surviving repertory of sixteenth-century lute music consists of intabulations. The Early Music Online (EMO) resource hosted by the British Library and RHUL comprises digital images of 300 books of printed music from before 1700.[1] About 10% are books of lute music, written in tablature. A number of these have been

[1] http://www.earlymusiconline.org

incorporated in the Electronic Corpus of Lute Music (ECOLM).[2] In the Transforming Musicology project[3] one of the three main work-packages is a programme of research on workflows and methods for musicological investigation of these resources. The tablature encodings used in this paper, together with detailed metadata, will soon be released as Linked Open Data.[4]

There is a considerable degree of repertorial overlap between the vocal and instrumental music in EMO; much of the vocal music also appears in intabulations for lute or keyboard, sometimes in multiple arrangements. This overlap provides the basis for ongoing musicological investigations within Transforming Musicology.

There is some specialist literature on the methods of intabulation adopted by various lutenist-composers of the 16th century [9][10][11], but systematic (computational) study has not hitherto been possible owing to a general lack of suitable annotated corpora of encodings. Here we describe an experiment to test the notion that, while some of the melodic patterns found in the corpus may be part of the common stylistic currency of a period, others may be diagnostic of an idiom of arrangement, or of a musical genre (in the musicological, rather than MIR, sense). That is, we ask whether some patterns are found across all lute music of a certain time or place, while others are encountered more in intabulations than in more exclusively idiomatic music composed for the instrument such as dances, preludes or fantasias.

Since we know that there were generally stylistic differences between vocal music composed for religious contexts and those for secular use, we further ask whether the use of embellishment patterns suggest that intabulators treated sacred music differently than secular in their arrangements. In this period, and with this repertory, a distinction based on language is generally safe – that is, Latin masses and motets are classified as sacred, whilst vernacular chansons and madrigals in French and Italian (or occasionally English or German), are considered secular even if the events they describe are biblical.[5]

The chromatic inflection of pitches in 16th-century staff-based music notation is not fully explicit. When such music is intabulated, however, the precise chromatic pitch of each note is given in the tablature. The resulting chromatic changes to the plain diatonic vocal model could be assigned to two contrasting motivations on the part of an intabulator: either as a record of a contemporary interpretation of standard *musica ficta* practice, or as the result of idiosyncracies of instrumental music which may in turn have had an effect on the emergence of the modern tonal system. Considered as the former, they have been used to attempt to reconstruct elements of vocal performance practice [2].

As an example of the practice, in Figure 1 we show the opening of a popular madrigal, first published in 1541. The original vocal version is shown (a) in short score, together with (b) one of its many intabulations, from a late 16th-century source. The interpolated notes in the bass (bar 1) and alto (bar 3) parts represent typical embellishment patterns of the type we discuss in this paper. Note also that the lute version supplies *musica ficta* accidentals not present in the vocal original; these are likely to differ from those supplied in other intabulations.

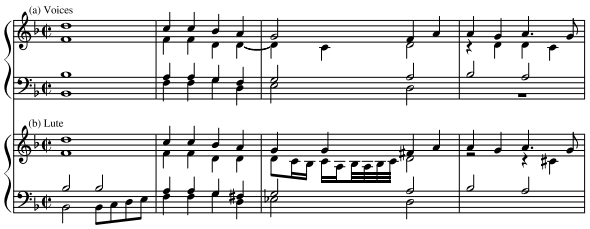


Figure 1. The opening of Berchem's 'O s'io potessi donna': (a) the original for voices, and (b) as intabulated by Emanuel Adriansen (*Pratum Musicum*, 1584)

For this paper, we study melodic embellishment patterns as possible stylistic markers of instrumental idiom. A few scholars have published 'vocabularies' of such patterns, inspired by the significant number of treatises from between 1535 and 1620 which give copious examples by way of instruction for players of string and wind instruments. [7][8]

In this preliminary study, we focus on a seminal example by a leading musicologist of the last century. In [3], Brown considers the embellishment patterns applied by three different early 16th-century intabulators to a single popular (secular) madrigal, Giachet Berchem's 'O si'o potessi donna', and provides a table of the patterns he identifies (78 distinct patterns). This table is effectively a set of embellishment templates that might be used by intabulators to expand simple intervals in the vocal parts of the model. An extract from Brown's table, showing some of the embellishment patterns used by Dominico Bianchini i n his intabulation in [4] is shown as Figure 2. The patterns are given without clefs or accidentals, as the precise pitches and intervals used can be expected to vary depending on context (including diatonic transposition and *musica ficta*).

In [5] Robison studied a single source of lute intabulations from 1558, listing a set of 95 such patterns found in the 76 arrangements therein. Neither he nor Brown attempted to generalize this work or to assess empirically the extent to which the patterns identified are themselves indicative of something more general, although in [7] Brown had discussed the tradition of instrumental and vocal embellishment through the numerous treatises published in the 16th and early 17th centuries.

[2] http://www.ecolm.org
[3] http://www.transforming-musicology.org
[4] Much of the metadata is already published at http://slickmem.data.t-mus.org/ .
[5] An example of the latter case is the enormously popular chanson by Orlande de Lassus, 'Susanne un jour', which concerns the story of Susanna and the Elders from the Book of Daniel.

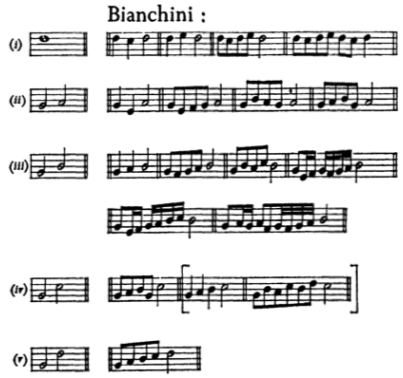


Figure 2. An extract from Brown's table of embellishment patterns found in lute intabulations of Berchem's 'O s'io potessi donna' (from [3])

3. METHOD

For this experiment, we use two test corpora:[6]

- a snapshot of the works currently in ECOLM, containing some 1,351 lute pieces. Works in ECOLM are transcribed as they appear in the original sources (including printing or scribal errors) and have been added and selected based on a series of academic research projects funded by the UK Arts and Humanities Research Council since 1999;
- a private collection, created by Sarge Gerbode, of 4,723 performing editions of lute pieces. These have been edited by the compiler, who is not an academic musicologist. The test corpus used is a subset of a larger collection, including only pieces from the 16th or very early 17th century. The corpus is known to contain some duplicates, for example to support performance on different instruments, but these represent a very small proportion of the works.

	Intabulations	Other	Total
ECOLM	147	1,204	1,351
Gerbode	1,048	3,675	4,723
Total	1,195	4,879	6,074

Table 1. Summary of our two test corpora

Metadata for the corpora includes genre and subgenre labels. The distribution of pieces is shown in Table 1. Dates, places and attributions are also available for some of the pieces, but there is likely to be some approximation involved. For the purpose of this experiment, pieces in these collections are represented as a sequence of `<onset, pitch>` tuples and interrogated using a Javascript implementation of SIA(M)ESE. [6] All versions of the subject of Brown's study [3], Berchem's madrigal, 'O s'io potessi donna', were deliberately excluded from both corpora to avoid bias.

[6] Both corpora, in the format we used for our experiment, and related metadata will be made available in due course to researchers via the Transforming Musicology web-site (see note 3).

Lute tablature has no voicing, pitch spelling or individual note-duration built explicitly into the notation. Instead, it indicates the fret/string positions of the left-hand fingers and the duration between successive chords or single notes struck with the right hand. We have encoded Brown's set of 78 (monophonic) embellishment templates using diatonic note-names. To search for passages matching one of the templates in the onset/pitch matrix, we need to realize all the possible chromatic inflections of all notes in each template. These are represented using chromatic pitch.

Each pattern was taken to start on (diatonic) C and all its notes realized in such a way that they could be spelled using 16th-century staff notation. This allowed pitches to be spelled as any of [C, C#, D, Eb, E, F, F#, G, G#, Ab, A, Bb, B]. All possible spellings of the pattern's note sequence starting on C and corresponding queries were generated, and then the process was repeated starting on D, and so on.

Figure 3a shows one of the simplest patterns from the Brown set, a plain passing-note motion. This is listed by Brown as one of those used to fill a rise of a third in a vocal line of the model (cf Figure 2, (iii)). In practice, the third might (in modern terms) be major or minor and filled by a combination of tones and semitones dependent on the local tonal context. Figure 3b-d are all what, in modern terms, would be called diatonic, in either major or minor modes. Figure 3e-g are less 'tonal' in that sense, and they are unlikely to occur in this context in 16th-century music. This does not mean that the note patterns within them will not occur in our corpora. Figure 3e, for example, is a simple chromatic scale and, while it is unlikely to happen in the context of 'filling a third', since the diminished third it connects probably never happens, in other contexts, it may be fairly common.

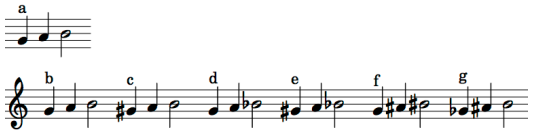


Figure 3. An embellishment pattern from Brown's list (the first at (iii) in Fig. 2) and some of its realisations.

For the 78 unique patterns in [3], this produced a total of 29,476 queries, which we have classified, based on the scale degrees chosen, according to whether or not they make tonal sense within the modern major/minor mode system, the idea being that we might be able to eliminate *a priori* those transformations with vanishingly small probability of occurrence. Of the realizations, we found 58 that could be interpreted in either a major or minor context, 88 just in major and 4,936 in minor (permitting the sixth and seventh degrees to be flattened or natural). This may be a fairly naïve approach from a music-theoretic standpoint, but it is reasonably complete and inclusive. The instrumental music of the period was changing rapidly in terms of scales, tonality and chro-

matic inflection, and we have not attempted to generate rules from the musical data itself. On the other hand, the only cost of generating too many patterns is computational time – if our searches include unlikely patterns, their effect on the final analysis will be negligible, since they will produce very few or no results. In the event we carried out all our searches using the full set of 29,476 patterns. If the major or minor forms had proved as diagnostically useful as the full set of patterns, then searches in future experiments could be limited to these, dramatically reducing the time required for searching, however this was not the case.

Turning to the time dimension, the relationship between metrical level and rhythmic notation is especially varied during this period, particularly so for lute music, the notation of which favors short durations. For this reason, each pattern is tested using not only the rhythms given by Brown, but also with the durations doubled and halved, trebling the final number of search patterns to 88,428.

As we have indicated, some of the shorter patterns described are trivial, and can be expected to occur universally, not only as elaboration patterns in other genres, but also as melodic elements in their own right. (The example given in Figure 3 is such a case; the other, more elaborate patterns in Figure 2 (iii) are less likely to appear ubiquitously.)

As an alternative method of *a priori* selection, those of Brown's list of patterns judged by an expert musicologist to be most characteristic of intabulations were labelled in advance. Although this was not used for pruning the search as originally intended, it gives an informal set of 'ground truth' judgements which offer us the opportunity to carry out a further test of our method.[7] In general, the patterns preselected by our expert were longer than those rejected, so we also did a similar evaluation based on pattern length.

Analyses were carried out on a complete set of results obtained from the exhaustive search of our two test-sets using all 88,428 transformations of the patterns as queries; the resulting 'hits' were stored in an SQL database together with metadata from the annotated corpus as the search was done. Since SIA(M)ESE is a partial-matching algorithm (as opposed to approximate-matching) we had the possibility of recording incomplete matches, which we limited to a threshold of 80% of the notes in the query; a simple SQL operation enables us to filter out the partial matches if necessary.

4. RESULTS

58.3% of all queries from our list of patterns found matches in intabulations, while only 48.3% of all queries on other lute pieces produced matches. This association of patterns with the intabulation repertoire was confirmed by a χ^2-test, comparing observed and expected frequencies of hits and non-hits between intabulations and other lute pieces ($\chi^2 = 6041$, $df = 1$, $p < .001$).

63.7% of the queries on intabulations from the sacred repertoire found patterns, while only 57.1% of queries on non-sacred intabulations contained patterns. A subsequent χ^2-test using data only from intabulations confirmed the strong association of pattern occurrences with the sacred repertoire ($\chi^2 = 508$, $df = 1$, $p < .001$).

However, simple χ^2-tests do not offer the possibility of investigating for the influence of other variables that might affect the likelihood of a pattern occurring in a given piece beyond the fact that it is an intabulation or not. Therefore we analysed the results with binomial mixed effects models where the variables of primary interest (`isIntabulation` and `isSacred`) can be entered alongside other variables we wish to control for. The dependent variable was a binary indicator recording whether or not a particular query returned a 'hit' (i.e. the query pattern was found in the queried piece). The main independent variable in the first model was `isIntabulation`. But we also controlled for the influence of three additional variables in the model (all described above): `isLong` (long patterns were defined to have between 7 and 13 notes, short patterns ranged from 3 to 6 notes), `IsMajor`, and `HasExpertLabel`. We used the identifier of the 78 patterns as a random effects variable.

Modelling was done using the glmer function in the R package lme4 [13] and started from a fully saturated model specifying all main effects as well as all 2-way, 3-way and the 4-way interaction effect. We then used a step-wise backward model selection procedure based on improvements in the Bayesian Information Criterion (BIC) to arrive at a more parsimonious final model which only contained highly significant effect terms (all p-values < .001). The parameter estimates, their standard errors, z-values of the Wald-statistic, associated significance level and the odds ratios derived from the parameter estimates of the final model are given in Table 2.

The model confirms the result from the first χ^2-test. The significant main effect for `IsIntabulation` indicates that patterns are more likely to be found in intabulations compared to the other pieces (odds ratio = 2.14). Thus, the collection of patterns can therefore be regarded as strongly characteristic of the intabulation repertoire. Further insights from the other three variables in the model suggest that overall: shorter patterns occur more frequently; major variants are less likely; and patterns preselected by the expert musicologist are less frequent. Insight from the significant 2-way interaction effects suggests that versions of patterns that are longer or are major are more likely in intabulations and also that longer versions with expert labels are more likely in intabulations.

[7] The full set of patterns we used, with those preselected as 'likely' highlighted, is available in music notation from: http://intabulations.data.t-mus.org/ .

Factors	Coeff.	SE	z	OR
(Intercept)	2.15	0.08	27.32***	8.58
isIntabulation	0.76	0.02	48.79***	2.14
isLong	-3.23	0.11	-28.88***	0.04
Major	-1.45	0.01	-178.40***	0.23
HasExpertLabel	-0.77	0.11	-6.76***	0.46
isIntabulation x isLong	0.13	0.02	5.20***	1.14
isIntabulation x Major	0.10	0.02	6.13***	1.11
isLong x Major	-0.11	0.01	-8.15***	0.90
isIntabulation x HasExpertLabel	-0.33	0.04	-7.42***	0.72
isLong x HasExpertLabel	-0.12	0.12	-0.96	0.89
isIntabulation x isLong x HasExpertLabel	0.30	0.05	6.00**	1.35

Table 2. Intabulations vs. non-intabulations: coefficient estimates, standard errors, z-values of Wald statistic with associated significance levels, and odds ratios for the independent variables in the binomial mixed effects model of all database queries. Coefficients represent the difference between the binary feature being present in the query compared with the feature being absent (the reference level). Significance levels are coded as follows:
* < .05, ** < .01, *** < .001.

To answer the question whether the patterns are more common in secular music, a second binomial mixed effects model was fitted to the data from the intabulations only. This model included the same variables as fixed effects. However, the main variable of interest was `isSacred`, a binary variable indicating whether the queried piece was from a sacred or secular repertoire. The same modelling selection strategy was employed, starting from a fully saturated model including all higher-order interaction effects and then applying a step-wise backward selection procedure based on the improvement in BIC. The parameter estimates, their standard errors, z-values of the Wald-statistic, associated significance level and the odds ratios derived from the parameter estimates of the final model are given in Table 3.

In this model `isSacred` on its own was not a significant predictor ($p = .259$; OR = 1.05). However, two 2nd order effects involving `isSacred` were highly significant (p-values < .001): queries that were either major variants or were longer queries on pieces from the sacred repertoire had a higher chance of returning a hit. Thus, the second order effects suggest that once pattern variants possess certain characteristics, they are more likely to be found in the sacred repertoire.

In summary, the results of the statistical analysis show that a) the embellishment patterns we used are more common in intabulations and b) the variants of these patterns with certain features (relatively long or major) are more common in the sacred repertoire. In sum, the results indicate the usefulness and relevance of embellishment patterns as descriptors of the idiom of intabulations but also highlight the importance of structural features (length, mode) of the patterns which should be taken into consideration in future studies to select specific subsets of patterns for musicological queries.

Factors	Coeff.	SE	z	OR
(Intercept)	2.85	0.13	22.51***	17.29
IsSacred	0.05	0.05	1.13	1.05
isLong	-3.93	0.20	-20.06***	0.02
Major	-1.42	0.02	-60.87***	0.24
isSacred x isLong	0.60	0.05	11.03***	1.82
isSacred x Major	0.59	0.06	10.46***	1.80
isLong x Major	-0.08	0.03	-2.29*	0.92
isSacred x isLong x Major	-0.55	0.07	-7.38***	0.58

Table 3. Sacred vs. secular: coefficient estimates, standard errors, z-values of Wald statistic with associated significance levels and odds ratios for the independent variables in the binomial mixed effects model of database queries including intabulations only. Coefficients represent the difference between the binary feature being present in the query compared with the feature being absent (the reference level). Significance levels are coded as follows:
* < .05, ** < .01, *** < .001.

5. CONCLUSIONS

This experiment shows that there is a clear stylistic difference between lute intabulations of vocal music and music composed directly for the lute in our corpus of 6,000 pieces, and that this can be revealed by comparing the frequency of occurrence of embellishment patterns identified in the musicological literature. To our knowledge, this is the first time a corpus of early instrumental music has been analysed in this way, and this result suggests that these patterns have promise for further use in stylistic analysis.

There are some biases in the corpora that affect this finding. Possibly the most important consideration from the statistical point of view is that not only do the corpora contain different arrangements of the same vocal piece, but they also may have multiple instances of the same intabulation from different sources. These versions should be note-identical in principle, but will often differ either in a few details (due to printing or transmission errors) or more substantially. The extent of these concordances is hard to estimate from the metadata we have, nor is it clear how they should be treated once identified, but it is certain that they will affect the assumption of independence of samples. We consider that identifying these and assessing their significance is an important musicological task, which will also help to make our methodology more sound.

Most of the intabulations present are from a narrower date range than the corpus as a whole. For the ECOLM collection especially, this bias is not particularly grave, due to the data gathering policy used so far, but evaluating its extent is not straightforward. Dating works is not easy and, although most printed collections have a known year of publication during this period, this may be later than the date of composition of the works in the book.

Another bias that is hard to evaluate is due to the relative length of pieces. The intabulations in both collections are longer, on average, than other genres and contain more notes. This could increase the likelihood of any given pattern occurring in any given piece; however, the relationship is unlikely to be straightforwardly linear and, since French chansons of the time (the favourite vocal models for intabulation) are characterized by a significant degree of repetition, even the increased note count cannot be taken to indicate a larger amount of distinct musical material. Further analysis is needed to determine the extent to which the same embellishment patterns were reused upon repetitions in the model.

6. FURTHER WORK

In the present study, we used melodic indicators drawn by a musicologist from three exemplary arrangements of a single vocal work published at around the same time. The madrigal in question in fact survives in over a dozen 16th-century intabulations (in both printed and manuscript sources for lute and keyboard), so an obvious next step would be to broaden our list of embellishment patterns to include at least some of these. The list could be further augmented by including patterns given in treatises for other instruments.

In general, our assertions here would be strengthened by a more fine-grained analysis. In particular, if a date and place of composition can be established for a sufficiently large number of works, then we should be able to evaluate the extent of any biases in our corpora. Similarly, where the intabulator is known and also composed original lute music, the two can be compared.

Intabulations are also distinguished by other parameters, most obviously texture. Even when one or more original voice-parts are omitted in an intabulation, one might expect the latter to maintain the texture of the remaining voices fairly consistently throughout the piece, whereas in freely-composed lute music there is in general no such obligation on the part of the composer. Though some fantasias and recercars (the main contrapuntal genres of 'pure' lute music) maintain a strict three, four or even five-voice texture, this is much less likely in dance music, for example.

In future studies, it will be helpful to make use of voice-leading where it can be derived from the tablature. In the case of much 16th-century keyboard music, especially that notated in so-called 'German organ tablature' the notes are separated into voices and given durations; with lute tablature the voices and durations (sometimes ambiguous, even for experienced players) have to be deduced, so in future work on lute music we shall apply recently-developed techniques such as that described in [12].

From a musicological standpoint, this experiment represents a first step towards a more detailed, corpus-level understanding of how intabulation worked as an artistic activity. Separating out the different influences and intentions of a composer or arranger is difficult, but it is our belief that some steps towards that separation can be made using approaches like this one.

The methods presented here can easily be replicated with a larger set of examples and on an enlarged corpus. Given encodings of the vocal models for the intabulations in the collections and a means of aligning the two, we hope to perform more nuanced studies, looking at arrangement as a process as well as simply studying the end product. In particular, we intend to investigate the notion of 'playability' as an aspect of this process and of the choice of repertory for intabulation, in the belief that the prescriptive nature of tablature itself captures much useful evidence which has not yet been exploited by scholars.

7. ACKNOWLEDGEMENTS

We are grateful to Sarge Gerbode for permission to use his personal collection of lute-music encodings, and to John Robinson for advice on lute-related matters. The work reported here was funded by the Arts and Humanities Research Council under grant AH/L006820/1.

8. REFERENCES

[1] H. M. Brown: *Instrumental Music Printed before 1600: A Bibliography*, Harvard University Press, Cambridge, Mass., 1965.

[2] F. Jürgensen, "Cadential Accidentals in the 'Buxheim Organ Book' and its Concordances: A Mid-Fifteenth-Century Context for 'musica ficta' Practice", *Acta Musicologica*, vol. 83, fasc. 1, pp. 39-68, 2011.

[3] H. M. Brown, "Embellishment in Early Sixteenth-century Italian Intabulations". *Proceedings of the Royal Musical Association*, vol. 100, pp. 49–83, 1973. Available online (accessed 16 March 2016): http://www.jstor.org/stable/766176

[4] D. Bianchini, *Intabolatura de lauto ... di recercari motetti madrigali canzon francese napolitane et balli*, Antonio Gardane, Venice, 1546.

[5] J. Robison, "Ornamentation in Sebastian Ochsenkun's *Tabulaturbuch auff die Lauten*," *Journal of the Lute Society of America*, Vol. xv, pp. 5–26, 1982.

[6] G. Wiggins, K, Lemström and D. Meredith, "SIA(M)ESE: An algorithm for transposition invariant, polyphonic content-based music retrieval," *Proceedings of the Third International Symposium on Music Information Retrieval (ISMIR 2002)*, Paris 2002, p. 283-284.

[7] H. M. Brown, *Embellishing Sixteenth-century Music*, Early music series, vol. 1, Oxford University Press, 1976.

[8] R. Erig and V. Gutmann, *Italian Diminutions from 1553-1638*, Zurich: Amadeus Press, 1979.

[9] H. M. Brown, "Intabulation," *Grove Music Online*, http://www.oxfordmusiconline.com/subscriber/book/omo_gmo, accessed 11 March 2016.

[10] H. Minamino, "Sixteenth-century lute treatises with emphasis on process and techniques of intabulation," unpub. PhD dissertation, Chicago University, 1988.

[11] J. Ward, "The Use of Borrowed Material in 16th-Century Instrumental Music," *Journal of the American Musicological Society*, vol. 5, no. 2, pp. 88-98, 1952

[12] R. de Valk, "Bringing 'Musicque into the tableture': machine-learning models for polyphonic transcription of 16th-century lute tablature", *Early Music*, vol. 43, no. 4, pp. 563-576, 2015.

[13] D. Bates, M. Maechler, B. Bolker and S. Walker, "Fitting linear mixed-effects models using lme4", *Journal of Statistical Software*, vol. 67, no. 1, pp. 1-48, 2015

THE SOUSTA CORPUS: BEAT-INFORMED AUTOMATIC TRANSCRIPTION OF TRADITIONAL DANCE TUNES

Andre Holzapfel
Austrian Research Institute
for Artificial Intelligence (OFAI)
andre@rhythmos.org

Emmanouil Benetos
Centre for Digital Music
Queen Mary University of London
emmanouil.benetos@qmul.ac.uk

ABSTRACT

In this paper, we present a new corpus for research in computational ethnomusicology and automatic music transcription, consisting of traditional dance tunes from Crete. This rich dataset includes audio recordings, scores transcribed by ethnomusicologists and aligned to the audio performances, and meter annotations. A second contribution of this paper is the creation of an automatic music transcription system able to support the detection of multiple pitches produced by lyra (a bowed string instrument). Furthermore, the transcription system is able to cope with deviations from standard tuning, and provides temporally quantized notes by combining the output of the multi-pitch detection stage with a state-of-the-art meter tracking algorithm. Experiments carried out for note tracking using 25ms onset tolerance reach 41.1% using information from the multi-pitch detection stage only, 54.6% when integrating beat information, and 57.9% when also supporting tuning estimation. The produced meter aligned transcriptions can be used to generate staff notation, a fact that increases the value of the system for studies in ethnomusicology.

1. INTRODUCTION

Automatic music transcription (AMT), the process of converting a music recording into notation, has largely focused on genres of eurogenetic [17] popular and classical music and especially on piano repertoire; see [3] for a recent overview. This is reflected in various AMT datasets, which consist of audio recordings along with a machine readable reference notation that specifies the time values of note onsets and offsets. Such datasets include the RWC database [14], the MAPS dataset [12], and the Bach10 dataset [9]. The reasons for the focus on certain styles seem manifold: Some aspects that might play a role are the cultural background of the AMT engineers, the relative ease of compiling reference notations for a piano using *MIDI*, and the predominant goal of transcription, i.e. the piano-roll, being closely related to the piano. As a point of fundamental importance, eurogenetic music lends itself nicely to the task of transcription, because in most cases a composition is first notated, and then performed using this notation. Hence, the notation can be interpreted as the ground-truth for an AMT system. The attempt to reconstruct this ground-truth, which is seen as a hidden generative concept for the performance [6], appears, at least at first glance, to be a well-defined task.

However, in the field of ethnomusicology, the process of transcribing a music performance mainly serves the means to analyse the structure of previously not notated music [11]. As a first contribution of this paper, we align a set of such recordings to transcriptions by ethnomusicologists, this way compiling an evaluation corpus for AMT that can enable us to monitor the performance of AMT systems on the music of a specific oral tradition. The music consists of Cretan dance tunes that were performed by Cretan musicians and recorded and transcribed by ethnomusicologists in the Crinnos project [2] that targeted the documentation of that specific music idiom. Only a small subset of the pieces recorded in the Crinnos project were transcribed, due to the large amount of effort that manual transcription takes. While it is clear that the building blocks of the tunes are small melodic phrases (see Section 2 for more detail), the way these phrases are strung together is largely improvised in the performance. These choices are not verbalized by the musicians, and an accurate transcription method will constitute an important tool to infer the grammar that underlies folk dance tunes in the area of the Eastern Mediterranean and beyond.

Therefore, as the second contribution of this paper, we extend an existing transcription algorithm [5] to be able to cope with tuning deviations and to take into account the metrical structure of the dance tunes. In Cretan music, as well as in many other music styles in the world, musicians tune their instruments according to personal preference. To the authors' knowledge, whilst several AMT systems support the extraction of multiple pitches in a high frequency resolution (e.g. [9, 13]), no AMT system has yet exploited that information for estimating the overall tuning level and to compensate for tuning deviations during the pitch quan-

AH is supported by the Austrian Science Fund (FWF: M1995-N31), and by the Vienna Science and Technology Fund (WWTF, project MA14-018).

EB is supported by a UK Royal Academy of Engineering Research Fellowship (grant no. RF/128).

Both authors contributed equally to this paper.

© Andre Holzapfel, Emmanouil Benetos. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Andre Holzapfel, Emmanouil Benetos. "The Sousta corpus: Beat-informed automatic transcription of traditional dance tunes", 17th International Society for Music Information Retrieval Conference, 2016.

tisation step. In addition, for many music styles, especially when related to dance, a clear metrical organization and a predictable tempo development enable for synchronisation between dancers and musicians in performances. For that reason, we apply a state-of-the-art meter tracking algorithm [15] for tracking beats and measures, and apply this information in order to achieve a temporal quantisation of note positions obtained from our AMT system. This way, we can obtain a transcription with temporal precision that is clearly increased to that of previously presented systems. In addition, this step enables us to obtain a visualisation of the transcription in a staff notation including bar positions, a perspective that marks an important step beyond the piano-roll as AMT output.

Our paper is structured as follows; Section 2 provides some detail about the musical idiom and corpus, and describes the process that was followed to align transcriptions with performances on the note level. Section 3 summarizes the chosen AMT system, and describes the extensions proposed in this paper. We then evaluate the performance of our systems, and provide illustrative examples in Section 4. Section 5 concludes the paper.

2. THE SOUSTA CORPUS

2.1 Background and motivation

The recordings that constitute the Sousta corpus presented in this paper were conducted in 2004 within the Crinnos project [2] in Rethymnon, Crete, Greece. Within the Crinnos project 444 pieces of Cretan music were recorded, and 40 of these performances were transcribed by ethnomusicologists. The transcriptions contain the melody played by the main melody instrument, as well as the vocal melody if vocals are present in a piece, and ignore the rhythmic accompaniment. Half of the 40 transcriptions regard a specific dance called *Sousta*. These transcriptions were chosen for a note-to-note alignment for several reasons.

First, this way we obtain a music corpus that is highly consistent in terms of musical style, which made a unified alignment strategy applicable to the recordings. The Sousta dance is usually notated in $2/4$ meter, and is characterized by a relatively stable tempo that lies between 110-130 beats per minute (bpm). The instrumental timbres are highly consistent, with usually two Cretan lutes playing the accompaniment, and one Cretan lyra (a pear-shaped fiddle) playing the main melody. All recordings were performed in the same studio, but with differing musicians. Apart from supporting our alignment procedure, the consistency of the recordings will enable a style comparison between individual musicians as part of our future work.

The **second** reason to choose the Sousta tunes lies with their value for music segmentation approaches. Like many tunes in the Eastern Mediterranean, the Sousta dance follows an underlying syntax that has been termed as *parataxis* [18]. In parataxis tunes, the building elements are short melodic phrases that are strung together in apparently arbitrary order without clear conjunctive elements. These phrases have a length of typically two measures for the Sousta dance. The 20 transcriptions were analysed within the Crinnos project and its elementary melodic phrases were identified by the experts. This way, a catalogue of 337 phrases was compiled that describes the melodic content of the tunes. Each measure of the corpus is assigned to a particular phrase. The note-to-note alignment that is made available in this paper enables to identify the phrase boundaries within the recordings, and this way the corpus can serve for music segmentation experiments. Such a corpus can form a basis for the development of an accurate system for syntactic analysis of music styles in the Eastern Mediterranean and elsewhere.

Such an analysis system, however, needs to be built on an AMT system that works as accurately as possible, in order to be able to analyse performances for which no manual transcription is available. We take this as a motivation to use for the first time, to the best of our knowledge, a set of performance transcriptions as the source for what is usually called ground-truth in MIR. This way, as our **third** motivation for choosing this specific style, we contribute to a larger diversity in available AMT datasets, by providing access to the aligned data for research purposes. The challenging aspects for AMT systems are the high density of notes, the tuning deviations, and the necessary focus on a bowed string instrument (lyra) within a pitched percussive accompaniment (lutes).

2.2 Alignment procedure

The first step to obtain a note-to-note alignment is to correct for transpositions between transcription and performance. Four out of the 20 pieces were played either one or two semitones higher than the transcription implied. Apparently, transcribers preferred to notate the upper empty string of the lyra as the note A, even if the player tuned the instrument one or several semitones higher.

As a second step, we conduct a meter tracking to obtain estimations for beat and measure positions, using the algorithm presented in [15]. The meter tracker was trained on the meter-annotated Cretan music used in [15], and then applied to track the meter in the 20 Sousta performances (for more details on the tracking algorithm see Section 3.4).

After that, the MIDI file obtained from the transcription is synthesized, and the algorithm from [16] is used to obtain an initial alignment of the MIDI file to the recorded performance. The timing of the measures is extracted from the aligned MIDI using the Matlab MIDI Toolbox [10]. Each of the estimated measures in the MIDI is then corrected to take the time value of the closest beat as obtained from the meter tracker from the recording. This step was included to compensate for timing inaccuracies of the automatic alignment. The obtained downbeats were manually corrected using Sonic Visualizer [1] . The output of this process is the exact timing of all measures that are notated in the transcription.

These manually corrected measure positions are then used as a source for the exact timing of the pre-aligned

[1] `http://sonicvisualiser.org/`

MIDI, by determining an alignment curve that corrects all note onsets accordingly. After that, also the note durations are edited to fit the notated length in seconds (*e.g.* a quarter note at 120 bpm should last 0.5 s). The result was again manually checked for inaccuracies. In addition, vocal sections were manually annotated. During vocal sections, the main instrument usually stops, and the transcription of musical phrases is of interest only for the instrumental sections in a recording. To the authors' knowledge, this measure-informed process is a novel and promising way to generate note-level transcriptions, as opposed to performing manual note corrections on an automatically aligned MIDI file, or by relying on an expert musician to follow and perform the recorded music in real-time [20].

The obtained corpus contains 35357 aligned notes in 4455 measures, distributed along the 20 recordings with a total length of 71m16s [2], 84% being instrumental. The average polyphony (on voiced frames only) is 1.08, and the average note duration is 108ms.

3. BEAT-INFORMED TRANSCRIPTION

This section describes the AMT system developed to transcribe the traditional dance corpus of Section 2. The main contributions of the proposed system are: (i) Supporting the transcription of lyra, a bowed string instrument that is present in all recordings of the corpus, by supplying the system with lyra templates; (ii) Estimating the overall tuning level and compensating for deviations from 440Hz tuning (cf. Section 1 for a discussion on related work for tuning estimation in AMT systems); (iii) Incorporating meter and beat information (from either manual meter annotations or estimations from a state-of-the-art meter tracking system [15]), resulting in a temporally quantised music transcription.

As a basis for the proposed work, the AMT system of [4] is adapted, which was originally aimed for transcribing 12-tone equal tempered music and supported Eurogenetic orchestral instruments. The system is based on probabilistic latent component analysis (PLCA), a *spectrogram factorization* method that decomposes an input time-frequency representation into a series of note templates and note activations. The system of [4] also supports the extraction of tuning information per transcribed note, which is used in this paper to estimate the overall tuning level. A diagram for the proposed system can be seen in Fig. 1, with all system components being presented in the following subsections.

3.1 Time-Frequency Representation

As input time-frequency representation for the transcription system, the variable-Q transform (VQT) spectrogram is used [19], denoted $V_{\omega,t}$ (ω is the log-frequency index and t is the time index). Here, the interpolated VQT spectrogram has a frequency resolution of 60 bins/octave (i.e. 20 cent resolution), using a variable-Q parameter $\gamma = 30$, with a minimum frequency of 36.7 Hz (i.e. at D1). As with the constant-Q transform (CQT), this VQT representation allows for pitch changes to be represented by shifts across the log-frequency axis, whilst offering an increased temporal resolution in lower frequencies compared to the CQT.

[2] For a list of recordings see www.rhythmos.org/ISMIR2016Sousta.html

3.2 Multi-pitch Detection

The multi-pitch detection model takes as input the VQT spectrogram of an audio recording and returns an initial estimate of note events. Here, we adapt the PLCA-based spectrogram factorization model of [4] for transcribing music produced by lyra. The model approximates $V_{\omega,t}$ as a bivariate probability distribution $P(\omega, t)$, which is in turn decomposed into a series of probability distributions, denoting note templates, pitch activations, tuning deviations, and instrument/source contributions.

The model is formulated as:

$$P(\omega,t) = P(t)\sum_{q,p,f,s} P(\omega|q,p,f,s)P_t(f|p)P_t(s|p)P_t(p)P_t(q|p) \quad (1)$$

where q denotes the sound state (e.g. attack, sustain parts of a note), p denotes pitch, s denotes instrument source, and f denotes log-frequency shifting with respect to 12-tone equal temperament (12-TET) at a tuning of 440 Hz for note A4. In (1), $P(t)$ is the energy of the VQT spectrogram, which is known. $P(\omega|q,p,f,s)$ is a 5-dimensional tensor that represents the pre-extracted spectral templates of lyra notes, per sound state q, pitch p and instrument model s, which are also pre-shifted across log-frequency f (cf. Section 4.1 on the extraction of lyra templates). $P_t(f|p)$ is the time-varying log-frequency shifting distribution per pitch (used to estimate tuning deviations per produced note), $P_t(s|p)$ is the source contribution per pitch over time, $P_t(q|p)$ is the time-varying sound state activation per pitch, and finally $P_t(p)$ is the pitch activation, i.e. the resulting multi-pitch detection output. In the proposed model, $p \in \{1\ldots,88\}$, with $p = 1$ denoting A0 and $f \in \{1,\ldots,5\}$, which respectively denote $\{-40,-20,0,20,40\}$ cent deviation from ideal tuning using 12-TET.

The unknown model parameters ($P_t(f|p)$, $P_t(s|p)$, $P_t(p)$, $P_t(q|p)$) are iteratively estimated using the expectation-maximization (EM) algorithm [8], with the update rules described in [4]. With 30 iterations set in the system, the runtime for multi-pitch detection is approximately 3×real-time using a Sony VAIO S15 laptop. The output of the model is $P(p,t) = P(t)P_t(p)$, which represents pitch activation probability in semitone scale.

3.3 Tuning Estimation - Postprocessing

The output of the multi-pitch detection model, $P(p,t)$, is non-binary and needs to be converted into a list of note events or a MIDI file. Firstly, in order to compensate for any tuning deviations from A4=440 Hz, a tuning estimation step is proposed, utilising information from the pitch

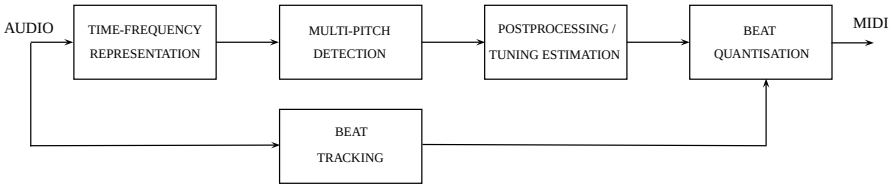


Figure 1: Diagram for the proposed system.

shifting parameter $P_t(f|p)$. The tuning probability vector is computed as:

$$P(f) = \sum_{p,t} P_t(f|p)P_t(p)P(t). \quad (2)$$

$P(f)$ provides an estimate on tuning deviations from 440 Hz tuning, with the various f values corresponding to $\{-40, -20, 0, 20, 40\}$ cent deviation. The final tuning estimate is given by $\text{argmax}_f\, P(f)$. Then, a 20 cent resolution time-pitch representation is computed:

$$P(f', t) = [P(f, 1, t) \cdots P(f, 88, t)] \quad (3)$$

where $P(f, p, t) = P_t(f|p)P_t(p)P(t)$, and $f' = \{1, \ldots, 88 * 5\}$ denotes pitch values between 1 and 88 with 20 cent resolution. The time-pitch representation is subsequently shifted towards 440 Hz tuning by reassigning the index $f' = f' + \text{argmax}_f\, P(f) - 3$ (since $f = 3$ represents 0 cent tuning deviation). Then, a tuning-compensated pitch activation $P(p, t)$ is re-computed from $P(f', t)$:

$$P(p, t) = \sum_{f'=5p-4}^{5p} P(f', t), \;\; \forall p \in \{1, \ldots, 88\}. \quad (4)$$

Following tuning compensation, thresholding is performed on $P(p, t)$, followed by a process for removing note events with a duration less than 40 ms. This results in a list of note events, denoted by onset, offset, and pitch, denoted $\text{nmat}_m(on, off, p)$, with $m \in \{1, \ldots, M\}$ denoting the note index, with on and off being the onset and offset times, respectively.

3.4 Meter Tracking & Quantisation

Since most dance tunes have an underlying stable meter and a relatively predictable tempo that enables dancers to synchronize, a quantisation of the estimated notes onto a tight metrical grid is likely to improve transcription performance. In addition, the notes in the obtained transcription are assigned positions within the meter, and obtain quantised note durations, which enables for an immediate visualisation as staff notation including a time signature.

In this paper, beat and measure positions for a recording are computed using the Bayesian meter tracker presented in [15]. Given a series of observations/features $\mathbf{y}_k$, with $k \in \{1, ..., K\}$, computed from a music signal, a set of hidden variables x_k is estimated. The hidden variables describe at each analysis frame k the position Φ_k within a measure, and the tempo in positions per frame ($\dot{\Phi}_k$). The goal is to estimate the hidden state sequence that maximizes the posterior (MAP) probability $P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$. If we express the temporal dynamics as a Hidden Markov Model (HMM), the posterior is proportional to

$$P(\mathbf{x}_{1:K}|\mathbf{y}_{1:K}) \propto P(\mathbf{x}_1) \prod_{k=2}^{K} P(\mathbf{x}_k|\mathbf{x}_{k-1})P(\mathbf{y}_k|\mathbf{x}_k) \quad (5)$$

In (5), $P(\mathbf{x}_1)$ is the *initial state distribution*, $P(\mathbf{x}_k|\mathbf{x}_{k-1})$ is the *transition model*, and $P(\mathbf{y}_k|\mathbf{x}_k)$ is the *observation model*. When discretising the hidden variable $\mathbf{x}_k = [\Phi_k, \dot{\Phi}_k]$, the inference in this model can be performed using the Viterbi algorithm. As in [15], a uniform initial state distribution $P(\mathbf{x}_1)$ was chosen. The transition model factorizes into two components according to

$$P(\mathbf{x}_k|\mathbf{x}_{k-1}) = P(\Phi_k|\Phi_{k-1}, \dot{\Phi}_{k-1})P(\dot{\Phi}_k|\dot{\Phi}_{k-1}) \quad (6)$$

with the two components describing the transitions of position and tempo states, respectively. The position transition model increments from Φ_{k-1} to Φ_k deterministically using the tempo $\dot{\Phi}_{k-1}$, starting from a value of 1 (at the beginning of a metrical cycle) to a value of 800. The tempo transition model allows for tempo transitions to the adjacent tempo states, allowing for gradual tempo changes. The observation model $P(\mathbf{y}_k|\mathbf{x}_k)$ divides the $2/4$-bars of meter-annotated Sousta tunes used in [15] into 32 discrete bins. Spectral-flux features are assigned to one of these metrical bins, and parameters of a Gaussian Mixture Model (GMM) are determined. The computation follows exactly the procedure described in [15], which lead to an almost perfect beat tracking for the Cretan tunes.

In order to quantise the detected note events nmat_m with respect to the estimated beat positions, firstly a metrical grid is created from the beat positions (beat_n, $n \in \{1, \ldots, N\}$). The metrical grid times are:

$$\text{grid}_{D(n-2)+d+1} = \text{beat}_{n-1} + (d/D)(\text{beat}_n - \text{beat}_{n-1}) \quad (7)$$

which are computed for $n = 2, \ldots, N$. In (7), D is the beat subdivision factor ($D = 4, 8$ corresponds to 16th and 32nd note subdivisions, respectively) and $d = \{0, \ldots, D-1\}$. Then, the beat-quantised transcription is produced by changing the onset time for each detected note nmat_m to the closest time instant computed from (7).

4. EXPERIMENTS

4.1 Training

Spectral templates for lyra are extracted from 20 short segments of solo lyra recordings, taken from the Crinnos

project [2] (disjoint from the recordings in the corpus described in Section 2). These are used as $P(\omega|q,p,f,s)$ in the model of (1). The recordings are partially annotated, identifying non-overlapping pitches. Then, for each recording the VQT spectrogram is computed as in Section 3.1 and spectral templates for each note are extracted using standard PLCA, whilst keeping the pitch activation matrix fixed to the reference pitch annotations. The templates are pre-shifted across log-frequency to account for tuning deviations, and templates for missing notes are created by shifting the extracted templates across the log-frequency axis. The resulting note range for the training templates is B3-F5.

4.2 Metrics

For assessing the performance of the proposed system in terms of multi-pitch detection, we utilise the onset-based metric used in the MIREX note tracking evaluations [1]. Here, a note event is assumed to be correct if its pitch corresponds to the ground truth pitch and its onset is within a ± 25 ms range of the ground truth onset. This is in contrast with the ± 50 ms onset tolerance setting used in MIREX, since the current corpus has fast tempo with short note durations. Using the above rule, the precision ($\mathcal{P}$), recall ($\mathcal{R}$), and F-measure ($\mathcal{F}$) metrics are defined:

$$\mathcal{P} = \frac{N_{tp}}{N_{sys}}, \quad \mathcal{R} = \frac{N_{tp}}{N_{ref}}, \quad \mathcal{F} = \frac{2 \cdot \mathcal{R} \cdot \mathcal{P}}{\mathcal{R} + \mathcal{P}} \tag{8}$$

where N_{tp} is the number of correctly detected pitches, N_{sys} is the number of detected pitches, and N_{ref} is the number of ground truth pitches. The above metrics are computed only for the recording regions that do not contain any vocal parts (a comparative experiment is done in Section 4.3).

4.3 Results

Using the evaluation metrics of Section 4.2, average results on the corpus described in Section 2 are presented in Table 1. Various configurations for the proposed system are used to evaluate the performance of each system component. Configuration 1 refers to simply using the output of the multi-pitch detection method from Section 3.2. Configuration 2 involves multi-pitch detection plus the proposed tuning estimation method from Section 3.3. Configuration 3 refers to multi-pitch detection combined with meter tracking from Section 3.4, thus producing a beat-aligned note output. Configuration 4 combines multi-pitch detection, tuning estimation, and meter tracking. Finally, Configuration 5 is an oracle version of Configuration 4, with the automatically estimated beats being replaced by the manually annotated measure positions, obtained as described in Section 2.2. In all configurations that utilise beat information the beat subdivision factor used is $D = 4$ (corresponding to 16th notes).

As can be seen from Table 1, when integrating tuning estimation the system performance improves by +2.2% in terms of F-measure. Likewise, by incorporating meter tracking, system performance improves by +13.5%, whereas when integrating both tuning and meter information the overall improvement is at +16.8%. Finally, using the reference measure annotations (Configuration 5) leads to an improvement of only +0.3% over the automatic beat extraction, indicating the reliability of meter tracking. Indeed, comparing the manually corrected measure annotations with those obtained from the automatic tracking, we obtain an F-measure [7] of 94.5%. This is an even higher meter tracking performance than observed on the Cretan recordings in [15], possibly caused by the fact that the recordings used in this paper were all conducted in the studio, and all tunes relate to the same dance. A discrepancy is also observed between average precision and average recall; the lower recall is mostly attributed to repeated notes in the ground truth, which are merged into single note events in the output transcription. The aforementioned results are approximately at the level of the state-of-the-art for AMT, when using other datasets [1]; results for individual recordings range from $\mathcal{F} = 70.9\%$ to 34.2% (the latter for a particularly idiosyncratic recording).

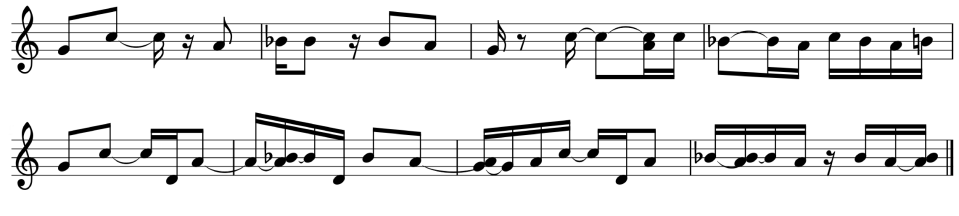
System	$\mathcal{F}$	$\mathcal{P}$	$\mathcal{R}$
Configuration 1	41.12%	45.33%	37.79%
Configuration 2	43.37%	48.12%	39.64%
Configuration 3	54.61%	66.38%	46.53%
Configuration 4	57.92%	70.71%	49.21%
Configuration 5	58.25%	71.14%	49.47%

Table 1: Average multi-pitch detection results using the corpus of Section 2, using various system configurations explained in Section 4.3.

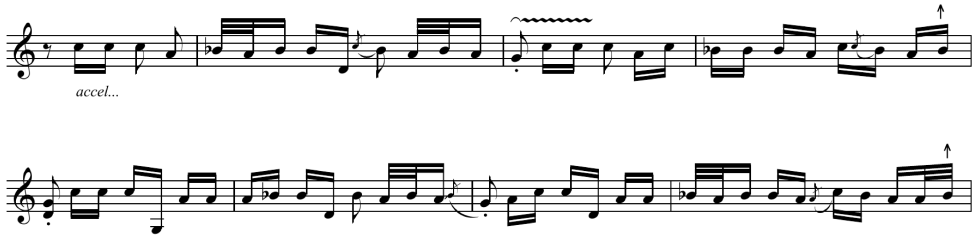
In Figure 2, an example of transferring a beat-quantised transcription obtained with Configuration 4 ($\mathcal{F} = 56.17\%$ for this piece) to staff notation is depicted, along with the manually transcribed reference notation. Spurious differences occur (*e.g.* added note G in the first measure) and the style of notation seems artificial. However, the resemblance between melodic contour in reference and automatic transcription is apparent. According to the analysis in the Crinnos project, the phrase depicted in Figure 3 is repeated with slight variations four times in the eight bars of this example, and comparing the phrase with each two consecutive bars in the transcriptions, this structure can be recognized. Figure 4 depicts the VQT and the pitch activations for the same eight bars. Further examples, all transcriptions obtained with Configuration 4 (MIDI and audio), and the reference annotations will be available on the paper's website [3] .

A comparison with a state-of-the-art AMT method is also made, employing the system of [21], which is based on non-negative matrix factorization. The aforementioned system decomposes a pitched sound as the sum of narrowband spectra. Results using multi-pitch detection only reach $\mathcal{F} = 26.08\%$ (in contrast with 41.12% for the proposed system). By integrating multi-pitch detection with beat information, the performance of [21] reaches

[3] www.rhythmos.org/ISMIR2016Sousta.html



(a) Automatic transcription



(b) Manual transcription, source [2]

Figure 2: Four repetitions of a two-bar phrase.

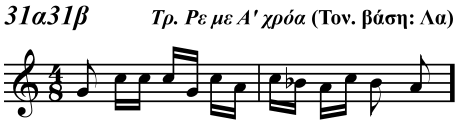


Figure 3: Sousta phrase that is repeated (in slight variations) in Figure 2 four times, source [2].

$\mathcal{F} = 35.94\%$ (as compared with 54.61% for the proposed method). It should be noted that tuning estimation cannot be achieved using the aforementioned system, as the output is quantised on the MIDI scale.

Experiments are also carried out using a larger onset tolerance for the metrics of Section 4.2, set to 50 ms (as in the MIREX evaluations [1]). When evaluating Configuration 4, $\mathcal{F} = 60.74\%$, while using the method of [21] $\mathcal{F} = 38.17\%$. The relatively small difference between using 25 ms or 50 ms tolerance is attributed to the fact that the employed corpus contains several short repeated notes; since the utilised evaluation metrics consider duplicate notes in the same temporal region as false alarms, a larger tolerance window penalises the systems' performance.

As mentioned in Section 4.2, the results presented in Table 1 are computed only for instrumental regions of the corpus, thus excluding any vocal parts. When also transcribing vocal parts, performance using Configuration 4 drops by 1.9% ($\mathcal{F} = 55.84\%$), due to the fact that the training data do not contain vocal templates; however, the transcription of vocal music is not in the scope of this work. Finally, experiments were carried out using a beat subdivision factor $D = 8$, which corresponds to 32nd notes. This results in $\mathcal{F} = 48.2\%$, which indicates that the onsets for some of the detected notes were placed in incorrect temporal positions on the metrical grid.

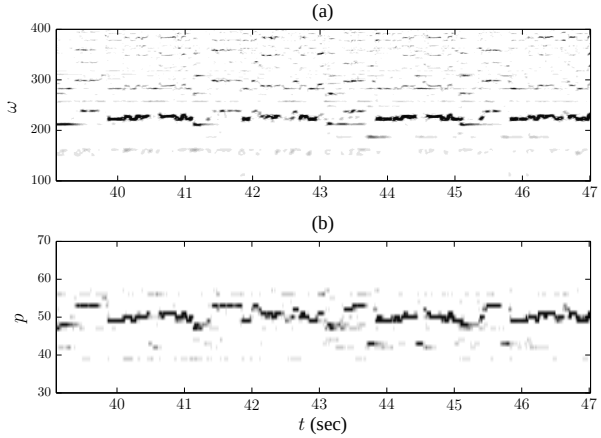


Figure 4: (a) The VQT spectrogram for the section transcribed in Figure 2. (b) The corresponding pitch activation $P(p, t)$.

5. DISCUSSION

In this paper, we presented a corpus for evaluation of AMT systems that is based on performance transcriptions manually compiled by experts in ethnomusicology. We then proposed an AMT system that can cope with tuning deviations, and we improve the performance of the AMT system by quantising its output on a metrical grid that was estimated using a state of the art meter tracker. Apart from the performance improvement, this quantisation enables for a straightforward generation of staff notation. For future work, we intend to improve the transcription by including instrument templates for the accompaniment instruments, which will enable for a better estimation of the main melody. Furthermore, we plan to conduct a user study with ethnomusicologists, who will evaluate the performance of our AMT system.

6. REFERENCES

[1] Music Information Retrieval Evaluation eXchange (MIREX). http://music-ir.org/mirexwiki/.

[2] Website of the Crinnos project. http://crinnos.ims.forth.gr. Accessed: 2016-03-16.

[3] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: challenges and future directions. *J. Intelligent Information Systems*, 41(3):407–434, December 2013.

[4] E. Benetos and T. Weyde. An efficient temporally-constrained probabilistic model for multiple-instrument music transcription. In *16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 701–707, Malaga, Spain, October 2015.

[5] Emmanouil Benetos and Andre Holzapfel. Automatic transcription of Turkish microtonal music. *Journal of the Acoustical Society of America*, 138(4):2118–2130, 2015.

[6] A. T. Cemgil, H. J. Kappen, and D. Barber. A generative model for music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(2):679–694, 2006.

[7] M. E. P. Davies, N. Degara, and M. D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. Technical Report C4DM-TR-09-06, Queen Mary University of London, Centre for Digital Music, 2009.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[9] Z. Duan, B. Pardo, and C. Zhang. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):2121–2133, November 2010.

[10] Tuomas Eerola and Petri Toiviainen. *MIDI Toolbox: MATLAB Tools for Music Research*. University of Jyväskylä, Jyväskylä, Finland, 2004.

[11] Ter Ellingson. Transcription. In Helen Myers, editor, *Ethnomusicology: An Introduction*, pages pp. 110–152. MacMillan, London, 1992.

[12] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1643–1654, August 2010.

[13] B. Fuentes, R. Badeau, and G. Richard. Harmonic adaptive latent component analysis of audio and application to music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(9):1854–1866, September 2013.

[14] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: music genre database and musical instrument sound database. In *International Symposium for Music Information Retrieval*, October 2003.

[15] Andre Holzapfel, Florian Krebs, and Ajay Srinivasamurthy. Tracking the "odd": Meter inference in a culturally diverse music corpus. In *Proceedings of ISMIR - International Conference on Music Information Retrieval*, pages 425–430, Taipei, Taiwan, 2014.

[16] R. Macrae and S. Dixon. Accurate real-time windowed time warping. In *International Society for Music Information Retrieval Conference*, pages 423–428, Utrecht, Netherlands, 2010.

[17] Robert Reigle. Reconsidering the idea of timbre: A brief history and new proposals. In *MusiCult '14: Music and Cultural Studies Conference*, pages 233–243, 2014.

[18] Haris Sarris, Tassos Kolydas, and Panagiotis Tzevelekos. Parataxis: A framework of structure analysis for instrumental folk music. *Journal of interdisciplinary music studies*, 4(1):71–90, 2010.

[19] C. Schörkhuber, A. Klapuri, N. Holighaus, and M. Dörfler. A Matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution. In *AES 53rd Conference on Semantic Audio*, page 8 pages, London, UK, January 2014.

[20] L. Su and Y.-H. Yang. Escaping from the abyss of manual annotation: New methodology of building polyphonic datasets for automatic music transcription. In *Int. Symp. Computer Music Multidisciplinary Research (CMMR)*, June 2015.

[21] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):528–537, March 2010.

LEARNING A FEATURE SPACE FOR SIMILARITY IN WORLD MUSIC

Maria Panteli, Emmanouil Benetos, Simon Dixon
Centre for Digital Music, Queen Mary University of London, United Kingdom
{m.panteli, emmanouil.benetos, s.e.dixon}@qmul.ac.uk

ABSTRACT

In this study we investigate computational methods for assessing music similarity in world music. We use state-of-the-art audio features to describe musical content in world music recordings. Our music collection is a subset of the Smithsonian Folkways Recordings with audio examples from 31 countries from around the world. Using supervised and unsupervised dimensionality reduction techniques we learn feature representations for music similarity. We evaluate how well music styles separate in this learned space with a classification experiment. We obtained moderate performance classifying the recordings by country. Analysis of misclassifications revealed cases of geographical or cultural proximity. We further evaluate the learned space by detecting outliers, i.e. identifying recordings that stand out in the collection. We use a data mining technique based on Mahalanobis distances to detect outliers and perform a listening experiment in the 'odd one out' style to evaluate our findings. We are able to detect, amongst others, recordings of non-musical content as outliers as well as music with distinct timbral and harmonic content. The listening experiment reveals moderate agreement between subjects' ratings and our outlier estimation.

1. INTRODUCTION

The analysis, systematic annotation and comparison of world music styles has been of interest to many research studies in the fields of ethnomusicology [5, 14, 20] and Music Information Retrieval (MIR) [7, 12, 28]. The former studies rely on manually annotating musical attributes of world music recordings and investigating similarity via several clustering techniques. The latter studies rely on automatically extracting features to describe musical content of recordings and investigating music style similarity via classification methods. We focus on research studies that provide a systematic way of annotating music; a method that often disregards specific characteristics of a music culture but makes an across-culture comparison feasible. We are interested in the latter and follow a computational approach to describe musical content of world music recordings and investigate similarity across music cultures.

© Maria Panteli, Emmanouil Benetos, Simon Dixon. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Maria Panteli, Emmanouil Benetos, Simon Dixon. "Learning a feature space for similarity in world music", 17th International Society for Music Information Retrieval Conference, 2016.

This study falls under the general scope of music corpus analysis. While several studies have focused on popular (mainly Eurogenetic) music corpus analysis, for example, the use of modes in American popular music [21], pitch, loudness and timbre in contemporary Western popular music [23], harmonic and timbral aspects in USA popular music [16], only a few studies have considered world or folk music genres, for example, the use of scales in African music [17]. Research projects have focused on the development of MIR tools for world music analysis [1], but no study, to the best of our knowledge, has applied such computational methods to investigate similarity in a world music corpus.

While the notion of world music is ambiguous, often mixing folk, popular, and classical musics from around the world and from different eras [4], it has been used to study stylistic similarity between various music cultures. We focus on a collection of folk recordings from countries from around the world, and use these to investigate music style similarity. Here we adopt the notion of music style by [19], 'style can be recognized by characteristic uses of form, texture, harmony, melody, and rhythm'. Similarly, we describe music recordings by features that capture aspects of timbral, rhythmic, melodic, and harmonic content [2].

The goal of this work is to infer similarity in collections of world music recordings. From low-level audio descriptors we are interested to learn high-level representations that project data to a music similarity space. We compare three feature learning methods and assess music similarity with a classification experiment and outlier detection. The former evaluates recordings that are expected to cluster together according to some ground truth label and helps us understand better the notion of 'similarity'. The latter evaluates examples that are different from the rest of the corpus and is useful to understand 'dissimilarity'. Outlier detection in large music collections can also be applied to filter out irrelevant audio or discover music with unique characteristics. We use an outlier detection method based on Mahalanobis distances, a common technique for detecting outliers in multivariate data [1]. To evaluate our findings we perform a listening test in the 'odd one out' framework where subjects are asked to listen to three audio excerpts and select the one that is most different [27].

Amongst the main contributions of this paper is a set

[1] Digital Music Lab (http://dml.city.ac.uk), CompMusic (http://compmusic.upf.edu/node/1), Telemata (https://parisson.github.io/Telemeta/)

[2] The use of form is ignored in this study as our music collection is restricted to 30-second audio excerpts.

of low-level features to represent musical content in world music recordings and a method to assess music style similarity. Our results reveal similarity in music cultures with geographical or cultural proximity and identify recordings with possibly unique musical content. These findings can be used in subsequent musicological analyses to track influence and cultural exchange in world music. We performed a listening test with the purpose of collecting similarity ratings to evaluate our outlier detection method. In a similar way, ratings can be collected for larger collections and used as a reference for ground truth similarity. The data and code for extracting audio features, detecting outliers and running classification experiments as described in this study are made publicly available [3] .

The paper is structured as follows. First a detailed description of the low-level features used in this study is presented in Section 2. Details of the size, type, and spatio-temporal spread of our world music collection are presented in Section 3. Section 4 presents the feature learning methods with specifications of the models and Section 5 describes the two evaluation methods, namely, classification and outlier detection. In Section 5.2 we provide details of the listening test designed to assess the outlier detection accuracy. Results are presented in Section 6 and finally a discussion and concluding remarks are summarised in Section 7 and 8 respectively.

2. FEATURES

Over the years several toolboxes have been developed for music content description and have been applied for tasks of automatic classification and retrieval [13, 18, 25]. For content description of world music styles, mainly timbral, rhythmic and tonal features have been used such as roughness, spectral centroid, pitch histograms, equal-tempered deviation, tempo and inter-onset interval distributions [7, 12, 28]. We are interested in world music analysis and add to this list the requirement of melodic descriptors.

We focus on state-of-the-art descriptors (and adaptations of them) that aim at capturing relevant rhythmic, melodic, harmonic, and timbral content. In particular, we extract onset patterns with the scale transform [10] for rhythm, pitch bihistograms [26] for melody, average chromagrams [3] for harmony, and Mel frequency cepstrum coefficients [2] for timbre content description. We choose these descriptors because they define low-level representations of the musical content, i.e. less abstract representations but ones that are more likely to be robust with respect to the diversity of the music styles we consider. In addition, these features have achieved state-of-the-art performances in relevant classification or retrieval tasks, for example, onset patterns with scale transform perform best in classifying Western and non-Western rhythms [9, 15] and pitch bihistograms have been used successfully in cover song (pitch content-based) recognition [26]. The low-level descriptors are later used to learn high-level representations using various feature learning methods (Section 4).

The audio features used in this study are computed with the following specifications. For all features we fix the sampling rate at 44100 Hz and compute the (first) frame decomposition using a window size of 40 ms and hop size of 5 ms. We use a second frame decomposition to summarise descriptors over 8-second windows with 0.5-second hop size. This is particularly useful for rhythmic and melodic descriptors since rhythm and melody are perceived over longer time frames. For consistency, the timbral and harmonic descriptors considered in this study are summarised by their mean and standard deviation over this second frame decomposition.

Rhythm and Timbre. For rhythm and timbre features we compute a Mel spectrogram with 40 Mel bands up to 8000 Hz using Librosa [4] . To describe rhythmic content we extract onset strength envelopes for each Mel band and compute rhythmic periodicities using a second Fourier transform with window size of 8 seconds and hop size of 0.5 seconds. We then apply the Mellin transform to achieve tempo invariance [9] and output rhythmic periodicities up to 960 bpm. The output is averaged across low and high frequency Mel bands with cutoff at 1758 Hz. Timbral aspects are characterised by 20 Mel Frequency Cepstrum Coefficients (MFCCs) and 20 first-order delta coefficients [2]. We take the mean and standard deviation of these coefficients over 8-second windows with 0.5-second hop size.

Harmony and Melody. To describe melodic and harmonic content we compute chromagrams using variable-Q transforms [22] with 5 ms hop size and 20-cent pitch resolution to allow for microtonality. Chromagrams are aligned to the pitch class of maximum magnitude for key invariance. Harmonic content is described by the mean and standard deviation of chroma vectors using 8-second windows with 0.5-second hop size. Melodic aspects are captured via pitch bihistograms which denote counts of transitions of pitch classes [26]. We use a window $d = 0.5$ seconds to look for pitch class transitions in the chromagram. The resulting pitch bihistogram matrix is decomposed using non-negative matrix factorization [24] and we keep 2 basis vectors with their corresponding activations to represent melodic content. Pitch bihistograms are computed again over 8-second windows with 0.5-second hop size.

3. DATASET

Our dataset is a subset of the Smithsonian Folkways Recordings, a collection of documents of "people's music", spoken word, instruction, and sounds from around the world [5] . We use the publicly available 30-second audio previews and from available metadata we choose the country of the recording as a proxy for music style. We choose a minimum number of $N = 50$ recordings for each country to capture adequate variability of its style-specific characteristics. For evaluation purposes we further require the

[3] https://code.soundsoftware.ac.uk/projects/feature-space-world-music

[4] https://bmcfee.github.io/librosa/

[5] http://www.folkways.si.edu

dataset to have the same number of recordings per country. By manually sub-setting the data we observe that an optimal number of recordings is obtained for $N = 70$, resulting in a total of 2170 recordings, 70 recordings chosen at random from each of 31 countries from North America, Europe, Asia, Africa and Australia. According to the metadata these recordings belong to the genre 'world' and have been recorded between 1949 and 2009.

4. FEATURE LEARNING

For the low-level descriptors presented in Section 2 and the music dataset in Section 3, we aim to learn feature representations that best characterise music style similarity. Feature learning is also appropriate for reducing dimensionality, an essential step for the amount of data we currently analyse. In our analysis we approximate style by the country label of a recording and use this for supervised training and cross-validating our methods. We learn feature representations from the 8-second frame-based descriptors.

The audio features described in Section 2 are standardised using z-scores and aggregated to a single feature vector for each 8-second frame of a recording. A recording consists of multiple 8-second frame feature vectors, each annotated with the country label of the recording. Feature representations are learned using Principal Component Analysis (PCA), Non-Negative Matrix Factorisation (NMF) and Linear Discriminant Analysis (LDA) methods [24]. PCA and NMF are unsupervised methods and try to extract components that account for the most variance in the data. LDA is a supervised method and tries to identify attributes that account for the most variance between classes (in this case country labels).

We split the 2170 recordings of our collection into training (60%), validation (20%), and testing (20%) sets. We train and test our models on the frame-based descriptors; this results in a dataset of 57282, 19104, and 19104 frames for training, validation, and testing, respectively. Frames used for training do not belong to the same recordings as frames used for testing or validation and vice versa as this would bias results. We use the training set to train the PCA, NMF, and LDA models and the validation set to optimise the number of components. We investigate performance accuracy of the models when the number of components ranges between 5 and the maximum number of classes. We use the testing set to evaluate the learned space by classification and outlier detection tasks as explained below.

5. EVALUATION

5.1 Objective Evaluation

To evaluate whether we have learned a meaningful feature space we perform two experiments. One experiment aims at assessing similarity between recordings from the same country (which we expect to have related styles) via a classification task, i.e. validating recordings that lie close to each other in the learned feature space. The second experiment aims at assessing dissimilarity between recordings by detecting 'outliers', i.e. recordings that lie far apart in the learned feature space.

Classification. For the classification experiment we use three classifiers: K-Nearest Neighbors (KNN) with $K = 3$ and Euclidean distance metric, Linear Discriminant Analysis (LDA), and Support Vector Machines (SVM) with a Radial Basis Function kernel. We report results on the accuracy of the predicted frame labels and the predicted recording labels. To predict the label of the recording we consider the vote of its frame labels and select the most popular label.

Outlier Detection. The second experiment uses a method based on squared Mahalanobis distances to detect outliers in multivariate data [1,8]. We use the best performing feature learning method, as indicated by the classification experiment, to transform all frame-based features of our dataset. For each recording we calculare the average of its transformed feature vectors and use this to compute its Mahalanobis distance from the set of all recordings. Using Mahalanobis, an n-dimensional feature vector is expressed as the distance to the mean of the distribution in standard deviation units. Data points that lie beyond a threshold, here set to the 99.5% quantile of the chi-square distribution with n degrees of freedom [6], are considered outliers.

5.2 Subjective Evaluation

To evaluate the detected outliers we perform a listening experiment in the 'odd one out' fashion [27]. A listener is asked to evaluate triads of audio excerpts by selecting the one that is most different from the other two, in terms of its musical characteristics. For the purpose of evaluating outliers, a triad consists of one outlier excerpt and two inliers as estimated by their Mahalanobis distance from the set of all recordings.

To distinguish outliers from inliers (the most typical examples) and other excerpts which are neither outliers nor inliers, we set two thresholds for the Mahalanobis distance. Distances above the upper threshold identify outliers, and distances below the lower threshold identify inliers. The thresholds are selected such that the majority of excerpts are neither outliers nor inliers. We randomly select 60 outliers and for each of these outliers we randomly select 10 inliers, in order to construct 300 triads (5 triads for each of 60 outliers), which we split into 10 sets of 30 triads. Each participant rates one randomly selected set of 30 triads.

The triads of outlier-inlier examples are presented in random order to the participant and we additionally include 2 control triads to assess the reliability of the participant. A control triad consists of two audio excerpts (the inliers) extracted from the first and second half, respectively, of the same recording and exhibiting very similar musical attributes, and one excerpt (the outlier) from a different recording exhibiting very different musical attributes. At the end of the experiment we include a questionnaire for demographic purposes.

We report results on the level of agreement between the computational outliers and the audio excerpts selected as the odd ones by the participants of the experiment. We

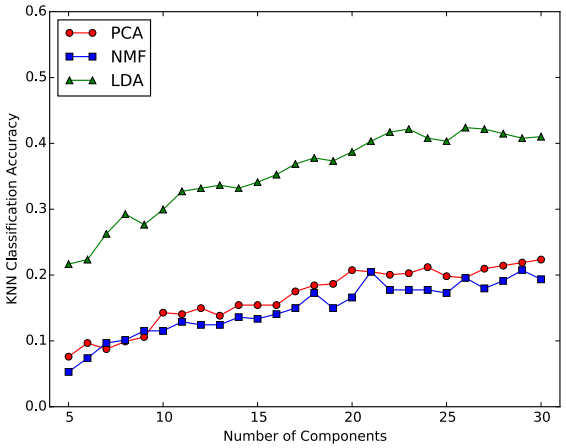


Figure 1. Classification accuracy for different numbers of components for PCA, NMF, and LDA methods (random baseline is 0.03 for 31 classes).

focus on two metrics; first, we measure the average accuracy between detected and rated outlier across all 300 triads used in the experiment, and second, we measure the average accuracy for each outlier, i.e. for each of 60 outliers we compute the average accuracy of its corresponding rated triads. Further analysis such as how the music culture and music education of the participant influences the similarity ratings is left for future work.

6. RESULTS

In this section we present results from the feature learning methods, their evaluation and the listening test as described in Sections 4 and 5.

6.1 Number of Components

First we present a comparison of classification performance when the number of components for PCA, NMF and LDA methods ranges between 5 and 30. For each number of components we train a PCA, NMF and LDA transformer and report classification accuracies on the validation set. The accuracies correspond to predictions of the label as estimated by a vote count of its predicted frame labels. We use the KNN classifier with $K = 3$ neighbors and Euclidean distance metric. Results are shown in Figure 1. We observe that the best feature learning method is LDA and achieves its best performance when the number of components is 26. PCA and NMF achieve optimal results when the number of components is 30 and 29 respectively. We fix the number of components to 30 as this gave good average classification accuracies for all methods.

6.2 Classification

Using 30 components we compute classification accuracies for the PCA, NMF and LDA transformed testing set. We also compute classification accuracies for the non-transformed testing set. In Table 1 we report accuracies

Classifier	Transform. Method	Frame Accuracy	Recording Accuracy
KNN	–	0.175	0.281
	PCA	0.177	0.279
	NMF	0.139	0.214
	LDA	0.258	**0.406**
LDA	–	**0.300**	0.401
	PCA	0.230	0.283
	NMF	0.032	0.032
	LDA	**0.300**	0.401
SVM	–	0.038	0.035
	PCA	0.046	0.044
	NMF	0.152	0.177
	LDA	0.277	0.350

Table 1. Classification accuracies for the predicted frame labels and the predicted recording labels based on a vote count (– denotes no transformation).

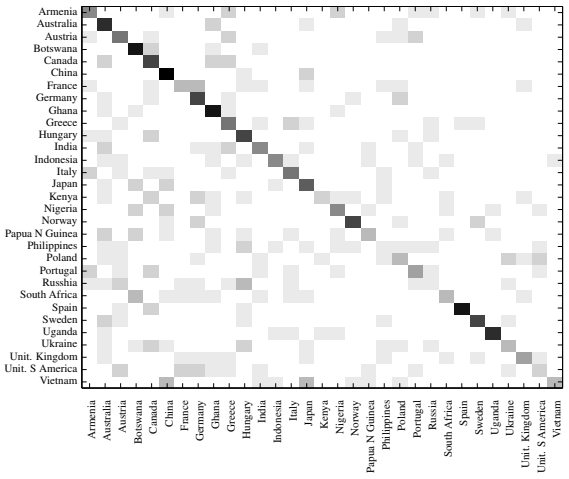


Figure 2. Confusion matrix for the best performing classifier, KNN with LDA transform (Table 1).

for the predicted frame labels and the predicted recording labels as estimated from a vote count (Section 4). The KNN classifier with the LDA transform method achieved the highest accuracy, 0.406, for the predicted recording labels. For the predicted frame labels the LDA classifier and transform was best with an accuracy of 0.300. In subsequent analysis we use the LDA transform as it was shown to achieve optimal results for our data.

For the highest classification accuracy achieved with the KNN classifier and the LDA transformation method (Table 1), we compute the confusion matrix shown in Figure 2. From this we note that China is the most accurate class and Russia and Philippines the least accurate classes. Analysing the misclassifications we observe the following: Vietnam is often confused with China and Japan, United States of America is often confused with Austria, France and Germany, Russia is confused with Hungary, and South Africa is confused with Botswana. These cases are characterised by a certain degree of geographical or cultural proximity which could explain the observed confusion.

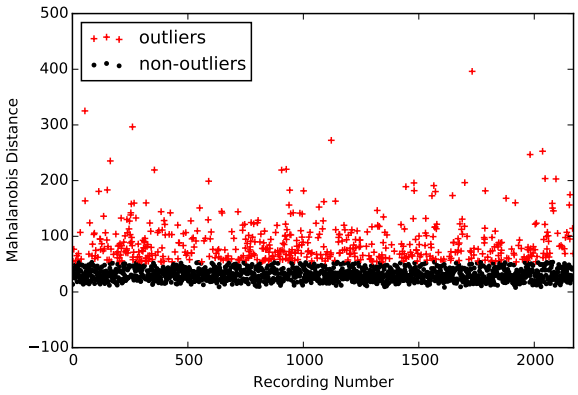


Figure 3. Mahalanobis distances and outliers at the 99.5% quantile of chi-square distribution.

6.3 Outlier Detection

The second experiment to evaluate the learned space aims at detecting outliers in the dataset. In this experiment we are not interested in how close music recordings of the same country are to each other, but we are rather interested in recordings that are very different from the rest. We use the LDA method as found optimal in the classification experiment (Section 6.2) to transform all frame-based feature vectors in our collection. Each recording is characterised by the average of its transformed frame-based descriptors.

From our collection of 2170 recordings (70 recordings for each of 31 countries), 557 recordings (around 26%) are detected as outliers at the chi-square 99.5% quantile threshold. In Figure 3 we plot the Mahalanobis distances for all samples in our dataset and indicate the ones that have been identified as outliers. The three recordings with maximum distances, i.e. standing out the most from the corpus, are identified as follows (in order of high to low Mahalanobis distance): 1) A recording of the dav dav instrument from the culture group 'Khmu' from Vietnam [6], 2) a rather non-musical example of bells from Greece [7], 3) an example of the angklung instrument from Indonesia [8]. These recordings can be characterised by distinct timbral and harmonic aspects or, in the case of the second example, by a distinct combination of all style attributes considered.

We plot the number of detected outliers per country on a world map (Figure 4) to get an overview of the spatial distribution of outliers in our music collection. We observe that Germany was the only country without any outliers (0 outliers out of 70 recordings) and Uganda was the country with the most outliers (39 outliers out of 70 recordings). Other countries with high number of outliers were Nigeria (34 outliers out of 70 recordings), Indonesia and Botswana (each with 31 outliers out of 70 recordings). We note that Botswana and Spain had achieved a relatively high classification accuracy in the previous evaluation (Section 6.2) and were also detected with a relatively high number of outliers (31 and 26 outliers, respectively). This could indicate that recordings from these two countries are consistent in their music characteristics but also stand out in comparison with other recordings of our world music collection.

[6] http://s.si.edu/1RuJfuu
[7] http://s.si.edu/21DgzP7
[8] http://s.si.edu/22yz0qP

6.4 Listening Test

The listening test described in Section 5.2 aimed at evaluating the outlier detection method. A total of 23 subjects participated in the experiment. There were 15 male and 8 female participants and the majority (83%) aged between 26 and 35 years old. A small number of participants (5) reported they are very familiar with world music genres and a similar number (6) reported they are quite familiar. The remaining participants reported they are not so familiar (10 of 23) and not at all familiar (2) with world music genres.

Following the specifications described in Section 5.2, participant's reliability was assessed with two control triads and results showed that all participants rated both these triads correctly. From the data collected, each of the 300 triads (5 triads for each of 60 detected outliers) was rated a minimum of 1 and maximum of 5 times. Each of the 60 outliers was rated a minimum of 9 and maximum of 14 times with an average of 11.5.

We received a total of 690 ratings (23 participants rating 30 triads each). For each rating we assign an accuracy value of 1 if the odd sample selected by the participant matches the 'outlier' detected by our algorithm versus the two 'inliers' of the triad, and an accuracy of 0 otherwise. The average accuracy from 690 ratings was 0.53. A second measure aimed to evaluate the accuracy per outlier. For this, the 690 ratings were grouped per outlier, and an average accuracy was estimated for each outlier. Results showed that each outlier achieved an average accuracy of 0.54 with standard deviation of 0.25. One particular outlier was never rated as the odd one by the participants (average accuracy of 0 from a total of 14 ratings). Conversely, four outliers were always in agreement with the subjects' ratings (average accuracy of 1 for about 10 ratings for each outlier). Overall, there was agreement well above the random baseline of 33% between the automatic outlier detection and the odd one selections made by the participants.

7. DISCUSSION

Several steps in the overall methodology could be implemented differently and audio excerpts and features could be expanded and improved. Here we discuss a few critical remarks and point directions for future improvement.

Numerous audio features exist in the literature suitable to describe musical content in sound recordings depending on the application. Instead of starting with a large set of features and narrowing it down to the ones that give best performance, we chose to start with a small set of features selected upon their state-of-the-art performance and relevance and expand the set gradually in future work. This way we can have more control of what the contribution is from each feature and each music dimension, timbre, rhythm, melody or harmony, as considered in this

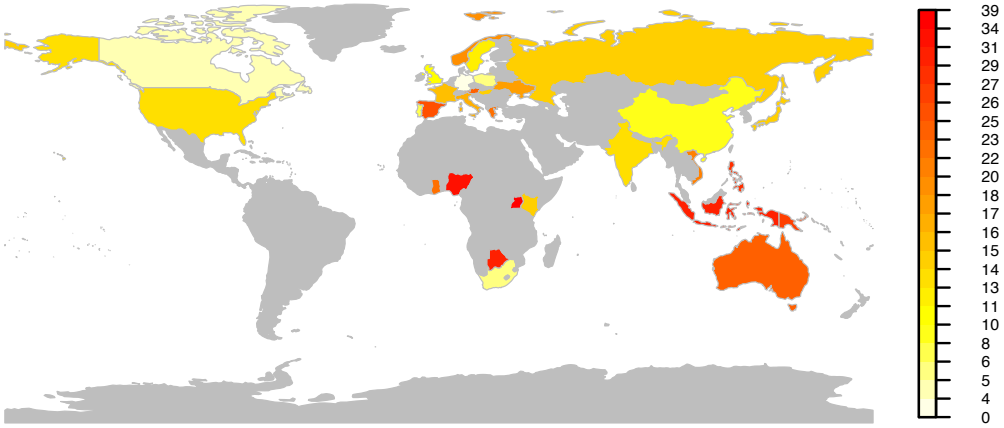


Figure 4. Number of outliers for each of the 31 countries in our world music collection (grey areas denote missing data).

study. The choice of features and implementation parameters could be improved, for example, in this study we have assumed descriptor summaries over 8-second windows but the optimal window size could be investigated further.

We used feature learning methods to learn higher-level representations from our low-level descriptors. We have only tested three methods, namely PCA, NMF, LDA, and did not exhaustively optimise parameters. Depending on the data and application, more advanced methods could be employed to learn meaningful feature representations [11]. Similarly, the classification and outlier detection methods could be tuned to give better accuracies.

The bigger aim of this work is to investigate similarity in a large collection of world music recordings. Here we have used a small dataset to assess similarity as estimated by classification and outlier detection tasks. It is difficult to gather representative samples of 'all' music of the world but at least a larger and better geographically (and temporally) spread dataset than the one used in this study could be considered. In addition, more metadata can be incorporated to define ground truth similarity of music recordings; in this study we have used country labels but other attributes more suitable to describe the music style or cultural proximity can be considered. An unweighted combination of features was used to assess music similarity. Performance accuracies can be improved by exploring feature weights. What is more, analysing each feature separately can reveal which music attributes characterise most each country or which countries share aspects of rhythm, timbre, melody or harmony.

Whether a music example is selected as the odd one out depends vastly on what it is compared with. Our outlier detection algorithm compares a single recording to all other recordings in the collection (1 versus 2169 samples) but a human listener could not do this with similar efficiency. Likewise, we could only evaluate a limited set of 60 outliers from the total of 557 outliers detected due to time limitations of our subjects. We evaluated comparisons from sets of three recordings and we used computational methods to create 'easy' triads, i.e. select three recordings from which one is as different as possible compared to the other two. However in some cases, as also reported by some of the participants, the three recordings were very different from each other which made it difficult to select the odd one out. In future work this could be improved by restricting the genre of the triad, i.e. selecting three audio examples from the same music style or culture. In addition the selection criteria could be made more specific; in our experiment we let participants decide on 'general' music similarity but in some cases it is beneficial to focus on, for example, only rhythm or only melody.

8. CONCLUSION

In this study we analysed a world music corpus by extracting audio descriptors and assessing music similarity. We used feature learning techniques to transform low-level feature representations. We evaluated the learned space in a classification manner to check how well recordings of the same country cluster together. In addition, we used the learned space to detect outliers and identify recordings that are different from the rest of the corpus. A listening test was conducted to evaluate our findings and moderate agreement was found between computational and human judgement of odd samples in the collection.

We believe there is a lot for MIR research to learn from and to contribute to the analysis of world music recordings, dealing with challenges of the signal processing tools, data mining techniques, and ground truth annotation procedures for large data collections. This line of research makes a large scale comparison of recorded music possible, a significant contribution for ethnomusicology, and one we believe will help us understand better the music cultures of the world.

9. ACKNOWLEDGEMENTS

EB is supported by a RAEng Research Fellowship (RF/128). MP is supported by a Queen Mary Principal's research studentship and the EPSRC-funded Platform Grant: Digital Music (EP/K009559/1).

10. REFERENCES

[1] C. C. Aggarwal and P. S. Yu. Outlier detection for high dimensional data. In *International Conference on Management of Data (ACM SIGMOD)*, pages 37–46, 2001.

[2] J. J. Aucouturier, F. Pachet, and M. Sandler. "The way it sounds": Timbre models for analysis and retrieval of music signals. *IEEE Transactions on Multimedia*, 7(6):1028–1035, 2005.

[3] M. A. Bartsch and G.H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, 2005.

[4] P. V. Bohlman. *World Music: A Very Short Introduction*. Oxford University Press, 2002.

[5] S. Brown and J. Jordania. Universals in the world's musics. *Psychology of Music*, 41(2):229–248, 2011.

[6] P. Filzmoser. A Multivariate Outlier Detection Method. In *International Conference on Computer Data Analysis and Modeling*, pages 18–22, 2004.

[7] E. Gómez, M. Haro, and P. Herrera. Music and geography: Content description of musical audio from different parts of theworld. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 753–758, 2009.

[8] V. Hodge and J. Austin. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[9] A. Holzapfel, A. Flexer, and G. Widmer. Improving tempo-sensitive and tempo-robust descriptors for rhythmic similarity. In *Proceedings of the Sound and Music Computing Conference*, pages 247–252, 2011.

[10] A. Holzapfel and Y. Stylianou. Scale Transform in Rhythmic Similarity of Music. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):176–185, 2011.

[11] E. J. Humphrey, A. P. Glennon, and J. P. Bello. Nonlinear semantic embedding for organizing large instrument sample libraries. In *Proceedings of the International Conference on Machine Learning and Applications*, volume 2, pages 142–147, 2011.

[12] A. Kruspe, H. Lukashevich, J. Abeßer, H. Großmann, and C. Dittmar. Automatic Classification of Musical Pieces Into Global Cultural Areas. In *AES 42nd International Conference*, pages 1–10, 2011.

[13] O. Lartillot and P. Toiviainen. A Matlab Toolbox for Musical Feature Extraction From Audio. In *International Conference on Digital Audio Effects*, pages 237–244, 2007.

[14] A. Lomax. *Folk song style and culture*. American Association for the Advancement of Science, 1968.

[15] U. Marchand and G. Peeters. The modulation scale spectrum and its application to rhythm-content description. In *International Conference on Difital Audio Effects*, pages 167–172, 2014.

[16] M. Mauch, R. M. MacCallum, M. Levy, and A. M. Leroi. The evolution of popular music: USA 1960-2010. *Royal Society Open Science*, 2(5):150081, 2015.

[17] D. Moelants, O. Cornelis, and M. Leman. Exploring African Tone Scales. In *Proccedings of the International Society for Music Information Retrieval Conference*, pages 489–494, 2009.

[18] G. Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. *Technical Report. IRCAM*, 2004.

[19] S. Sadie, J. Tyrrell, and M. Levy. *The New Grove Dictionary of Music and Musicians*. Oxford University Press, 2001.

[20] P. E. Savage, S. Brown, E. Sakai, and T. E. Currie. Statistical universals reveal the structures and functions of human music. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 112(29):8987–8992, 2015.

[21] E. G. Schellenberg and C. von Scheve. Emotional cues in American popular music: Five decades of the Top 40. *Psychology of Aesthetics, Creativity, and the Arts*, 6(3):196–203, 2012.

[22] C. Schörkhuber, A. Klapuri, N. Holighaus, and M. Dörfler. A Matlab Toolbox for Efficient Perfect Reconstruction Time-Frequency Transforms with Log-Frequency Resolution. In *AES 53rd Conference on Semantic Audio*, pages 1–8, 2014.

[23] J. Serrà, Á. Corral, M. Boguñá, M. Haro, and J. L. Arcos. Measuring the Evolution of Contemporary Western Popular Music. *Scientific Reports*, 2, 2012.

[24] L. Sun, S. Ji, and J. Ye. *Multi-Label Dimensionality Reduction*. CRC Press, Taylor & Francis Group, 2013.

[25] G. Tzanetakis and P. Cook. MARSYAS: a framework for audio analysis. *Organised Sound*, 4(3):169–175, 2000.

[26] J. Van Balen, D. Bountouridis, F. Wiering, and R. Veltkamp. Cognition-inspired Descriptors for Scalable Cover Song Retrieval. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 379–384, 2014.

[27] D. Wolff and T. Weyde. Adapting Metrics for Music Similarity Using Comparative Ratings. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 73–78, 2011.

[28] F. Zhou, Q. Claire, and R. D. King. Predicting the Geographical Origin of Music. In *IEEE International Conference on Data Mining*, pages 1115–1120, 2014.

Oral Session 5

Structure

SYSTEMATIC EXPLORATION OF COMPUTATIONAL MUSIC STRUCTURE RESEARCH

Oriol Nieto
Pandora Media, Inc.
onieto@pandora.com

Juan Pablo Bello
Music and Audio Research Laboratory
New York University
jpbello@nyu.edu

ABSTRACT

In this work we present a framework containing open source implementations of multiple music structural segmentation algorithms and employ it to explore the hyper parameters of features, algorithms, evaluation metrics, datasets, and annotations of this MIR task. Besides testing and discussing the relative importance of the moving parts of the computational music structure eco-system, we also shed light on its current major challenges. Additionally, a new dataset containing multiple structural annotations for tracks that are particularly ambiguous to analyze is introduced, and used to quantify the impact of specific annotators when assessing automatic approaches to this task. Results suggest that more than one annotation per track is necessary to fully address the problem of ambiguity in music structure research.

1. INTRODUCTION

In recent years, numerous open source packages have been published to facilitate research in the field of music information retrieval. These publications tend to focus on a specific part of the standard methodology of MIR: audio feature extraction (e.g., Essentia [2], librosa [14]), datasets (e.g., SALAMI [22], MSD [1]), evaluation metrics (e.g., mir_eval [20]), and task-specific algorithm implementations (e.g., segment boundary detection [13], pattern discovery [16], beat tracking [4]). What is often missing are integrated frameworks where the choice of different moving blocks of the whole process (i.e., feature design, algorithm implementations, annotated datasets and evaluation metrics) can be interchanged in a seamless fashion, allowing the type of in-depth comparative studies on state of the art techniques that are virtually impossible in the context of MIREX [1]: e.g., what combination of features or preprocessing stages maximize results? What mixture of approaches should be used if highly accurate boundary localization is important? What implementations are more resilient to changes in data, features or prior information?

In this work we introduce an open source framework to facilitate reproducibility and encourage research in music structural segmentation. Building on top of existing open projects [9, 14, 20, 22], this framework combines feature computation, algorithm implementations, evaluation metrics, and annotated datasets in a standalone software focused on this area of MIR. Besides describing the architecture of this framework, we show its potential by compiling a new dataset composed of poly-annotated tracks carefully selected by the presented software, and conducting a series of experiments to systematically explore the impact of each moving part of this task. These new data and explorations reinforce the notion that this task is highly ambiguous [3], since we show that the ranking of computational approaches in terms of performance depends not only on what feature or dataset is employed, but on which annotation is used as reference.

The rest of this article is organized as follows: In Section 2 the framework is introduced. Section 3 discusses the creation of the new dataset. In Section 4 the explorations of the different moving parts of the structural segmentation eco-system are presented. Finally, in Section 5, the conclusions are drawn.

2. MUSIC STRUCTURE ANALYSIS FRAMEWORK

MSAF [2] is an open source framework written in Python that allows to thoroughly analyze the entire music structure segmentation eco-system. In this section we provide an overview of this MIR task and a description of the most relevant characteristics of this framework.

2.1 Structural Segmentation

This task, whose main goal is to automatically identify the large-scale, non-overlapping segments of a given audio signal (e.g., verse, chorus), has been investigated in MIR for over a decade [19], and nowadays it is still one of the most active in MIREX [23]. Potential applications to motivate its research are numerous, e.g., improve intra-track navigation, yield enhanced segment-level music rec-

[1] One exception would be MARSYAS [25], where feature extraction, algorithm implementations for a limited number of tasks, dataset annotations, and evaluations coexist in a single environment.

[2] https://github.com/urinieto/msaf

© Oriol Nieto, Juan Pablo Bello.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Oriol Nieto, Juan Pablo Bello. "Systematic Exploration of Computational Music Structure Research", 17th International Society for Music Information Retrieval Conference, 2016.

ommendation systems, produce educational visualisation tools to better understand musical pieces. This task is often divided in two subproblems: *boundary detection* and *structural grouping*. The former identifies the beginning and end times of each music segment within a piece, and the latter labels these segments based on their acoustic similarity.

Several open source implementations to approach this problem have been published [10, 12, 13, 26], but given the differences in feature extraction, datasets, and evaluation metrics, it can be challenging to easily compare their results (e.g., Weiss' implementation [26] expects features computed with Ellis' code [5]; Levy's implementation [10] is only available in the form of a Vamp Plugin; McFee's publication [12] reports non-standard evaluation metrics with the first and last boundary removed). Our proposed, open-source MSAF seeks to address these issues by integrating these various components.

In the following subsections, the main parts of this framework are described. MSAF is written such that any of these parts could be easily extended.

2.2 Features

Most music structure algorithms accept different types of features in order to discover structural relations in harmony, timbre, loudness or a combination of them. Here we list the set of features that MSAF can compute by making use of librosa [14]: Pith Class Profiles (PCPs, representing harmony), Mel-Frequency Cepstral Coefficients (MFCCs, representing timbre), Tonal Centroids (or Tonnetz [7], representing harmony), and Constant-Q Transform (CQT, representing harmony, timbre and loudness).

Each of these features depend on additional analysis parameters such as sampling rate, FFT size, and hop size. Furthermore, a beat-tracker [4] (contained in librosa) is employed to aggregate all the features at a beat level, thus obtaining the so-called beat-synchronous representations. This process, which is common in structural segmentation, reduces the number of feature vectors while introducing tempo invariance. In this work we rely on this type of features exclusively, even though MSAF can operate both on beat- or frame-synchronous descriptors.

2.3 Algorithms

Algorithms of this task are commonly classified based on the subtask that they aim to solve. MSAF includes: seven algorithms that detect boundaries, and five that group structure (see Table 1).

The implementations in MSAF are either forked from the public repositories of their original publications [10, 12, 13, 26] or implemented from scratch when no access to the source code is available. Some differences in the results might arise given the difficulty of exactly recreating all implementation details, even though these differences appear to be minor.

Algorithm	Boundary	Grouping
2D-Fourier Magnitude Coeffs [15]	No	Yes
Checkerboard Kernel [6]	Yes	No
Constrained Cluster [10]	Yes	Yes
Convex NMF [18]	Yes	Yes
Laplacian Segmentation [12]	Yes	Yes
Ordinal LDA [13]	Yes	No
Shift Invariant PLCA [26]	Yes	Yes
Structural Features [21]	Yes	No

Table 1: Approaches included in MSAF and used in the experiments.

2.4 Evaluation Metrics

Structural segmentation employs multiple metrics to evaluate each of its two subproblems. For boundary detection, the Hit Rate is the most standard one, where the estimated boundaries are considered "hits" if they fall within a certain time window from the reference ones. This yields Precision (how many estimated boundaries are correct) and Recall (how many reference boundaries were estimated) scores, which are weighted with the standard F-measure. The time windows are typically 3 or 0.5 seconds. Moreover, sometimes the first and last boundaries are "trimmed" (i.e., not considered) given the fact that they should correspond to the beginning and end of the track, which should be trivial to detect. It has been shown that having a stronger weight on Precision than Recall tends to better align with perception [17], therefore this weight parameter is also part of MSAF. The other standard metric to report the quality of the boundaries is the Median Deviation [24], where the median deviation from each estimated boundary to its closest reference, and vice versa, are reported.

The most standard metric to assess the quality of the structural grouping subproblem is the Pairwise Frame Clustering [10]. This metric compares each pair of frames by checking whether they belong to the same label (or cluster), both for the estimation and reference. The ratio between the two sets of pairs over the number of similar pairs in the estimation yields the Precision metric, while Recall is the ratio between the two sets over the number of similar pairs in the reference. Again, the F-measure weights these two scores. Finally, an alternative metric named Normalized Conditional Entropy [11], based on the entropy of each frame between the estimation and reference, is also reported. This metric is formed by the under- and over-segmentation scores, which, again, can be compacted in a single score with the F-measure.

These metrics are reported in MIREX, and are transparently implemented in mir_eval [20], which MSAF employs.

2.5 Datasets

The following annotated datasets are the most common for assessing structural segmentation: Isophonics – 298 annotated tracks mostly of popular music [3]; SALAMI – two human references plus three levels of annotation per

[3] `http://isophonics.net/datasets`

track [22]. It contains 769 musical pieces ranging from western popular music to world music [4] ; The Beatles TUT – refined version of 174 annotations of The Beatles corrected and published by members of the Tampere University of Technology [5] .

Additionally, we make use of these uncommon and novel datasets: Cerulean – 104 songs collected by a company, subjectively deemed to be challenging tracks within a large collection. The genre of the songs varies from classical to heavy metal; Epiphyte – another industrial set of 1002 tracks composed mainly of pop music songs; Sargon – small set of 30 minutes of heavy metal tracks released under a Creative Commons license; SPAM – new dataset discussed in the next section.

All of these datasets are converted to the JAMS format [9], which is the default format that MSAF employs, and are publicly available in the MSAF repository (except Cerulean and Epiphyte, which are privately owned). This format is JSON-compatible and allows for multiple annotations in a single file for numerous tasks operating on a given audio track, making it ideal for the purposes of this work.

3. STRUCTURAL POLY-ANNOTATIONS OF MUSIC

SPAM is a new dataset composed of 50 tracks sampled from a large collection containing all the previously discussed sets (a total of 2,173 tracks). Following an approach inspired by [8], all MSAF algorithms were run on these 2,173 tracks. The tracks were then ranked based on the average Hit Rate F-Measure with 3 seconds window (i.e., the most standard metric for boundary detection) across all algorithms. Formally, the rank is computed using the mean ground-truth precision (MGP) score, defined as follows:

$$\text{MGP}_i(B, g) = \frac{1}{M} \sum_{j=1}^{M} g(\mathbf{b}_{ij}) \quad (1)$$

where $B \in \mathbb{R}^{N,M}$ is the matrix containing all the boundary estimations $\mathbf{b}_{ij} \in B$ for track $i \in [1, N]$ using algorithm $j \in [1, M]$, and g is the evaluation function (i.e., Hit Rate at 3 seconds). Ranking the tracks using this metric yields a list sorted by how *challenging* these tracks are for automatic segmentation.

The SPAM dataset is composed by the 45 most challenging tracks (i.e., the 45 at the bottom of the ranked list) plus the 5 least challenging (i.e., the top 5 tracks in the list). The number of tracks was kept small to facilitate the collection of five additional annotations using the same guidelines as in SALAMI. These five annotations were collected by music students (four graduates and one undergraduate) from the Steinhardt School at New York University, with an average number of years in musical training of 15.3 ± 4.9, and with at least 10 years of experience as players of a musical instrument. The goal was to create a set in which to explore the variability of structural annotations across subjects, focusing on the most challenging tracks (45) while still having a reduced control group (5). This split could foster further investigation on the differences between *easy* and *challenging* tracks.

The type of music ranges between jazz and blues, classical, world music, rock, western pop, and live recordings. Due to legal copyright issues, the audio of these tracks is not available, however, the features described in Section 2.2 are included along with the five annotations for each of the 50 tracks of SPAM.

4. EXPERIMENTS

In this section we report a series of experiments to further explore the task of structural segmentation carried out using MSAF, classified by the moving parts described previously. Each experiment can be subdivided based on the subproblems of boundary detection and structural grouping. For each experiment the default parameters are the following, unless otherwise specified: sampling rate is 11025Hz; FFT and hop sizes are 2048 and 512 samples, respectively; default feature type is beat-synchronous PCP; number of octaves and starting frequency for the PCPs are 7 and 27.5Hz, respectively; number of MFCCs is fixed to 14; number of CQT bins, starting at 27.5Hz, is 87; evaluation metrics are the F-measures of the Hit Rate with 3 seconds window and the Pairwise Frame Clustering for boundary detection and structural grouping, respectively; the boundaries used as input to the structural grouping algorithms are annotated; and the default dataset is The Beatles TUT. Code to reproduce the plots and results is available online [6] .

4.1 Features

We start by running all MSAF algorithms [7] using different types of features. In Figure 1 we can see, as expected, that the average scores of the boundary algorithms vary based on the feature types. This aligns with the results of a two-way ANOVA on the F-measure of the Hit Rate using the algorithms and features as factors, where the effect on the type of features is significant ($F(3, 3460) = 4.20, p < 0.01$). Also as expected, there is significant interaction between factors ($F(12, 3460) = 15.15, p < 0.01$), which can be seen in the plot when observing the poor performance of the Constrained Clustering algorithm for the Constant-Q features in comparison with the rest of the features.

A similar behavior occurs when analyzing the performance of the structural grouping algorithms, as can be seen in Figure 2. The two-way ANOVA confirms dependency of the type of features for these algorithms ($F(3, 2768) = 18.07, p < 0.01$), with significant interaction ($F(9, 2768) = 14.5, p < 0.01$) mostly due to the behavior, again, of the Constrained Clustering algorithm

[4] Only the first half of the full SALAMI annotations were used, since the authors did not have access to the rest of audio files.

[5] http://www.cs.tut.fi/sgn/arg/paulus/beatles_sections_TUT.zip

[6] https://github.com/urinieto/msaf-experiments

[7] Except Ordinal LDA and Laplacian Segmentation, since they only accept a specific combination of features as input.

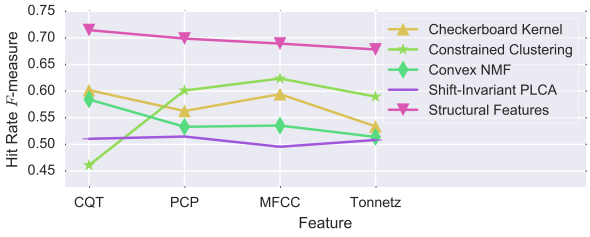


Figure 1: Boundary algorithms' performance depending on the type of features.

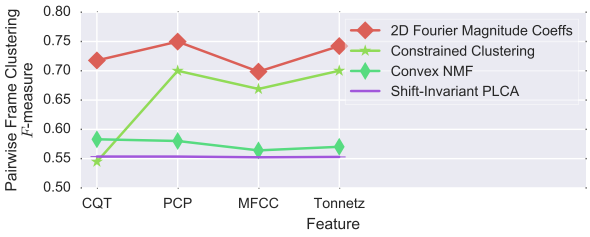


Figure 2: Structural algorithms' performance depending on the type of features.

when using Constant-Q features. Convex NMF still performs slightly better with this type of features, while the rest of the algorithms seem to be optimized to operate on the features suggested in their original publications.

This experiment yields two major points: (i) features describing timbre information (CQT, MFCCs) seem to be slightly better than those describing pitch information (PCPs, Tonnetz) for boundary detection, but the reverse seems to be true for clustering, and (ii) the Structural Features and Convex NMF methods obtain better results when using CQT, while in their original publications they recommend using harmonic features such as PCPs.

4.2 Algorithms

The quality of the segment boundaries can impact the results of the structural grouping subproblem [21]. MSAF lets us explore this by using the output of several boundary algorithms as input to the structural algorithms. Figure 3 shows average scores of the structural algorithms in MSAF. Additionally, the results with annotated boundaries are used and plotted in the first column. The boundary methods are sorted from left to right based on their performance on The Beatles TUT dataset. As expected, the quality of the boundary detection process affects the structural subproblem. A two-way ANOVA on the F-measure of the Pairwise Frame Cluster, with boundary and structural algorithms as factors, confirms this ($F(7, 6920) = 183.10, p < 0.01$). A significant interaction between the two factors is also present ($F(28, 6920) = 16.44, p < 0.01$), suggesting that the ranking of the algorithms will vary depending on the boundaries employed. This is confirmed by the Friedman test, which ranks the structural algorithms using Structural Features boundaries ($F(4) = 242.31, p <$

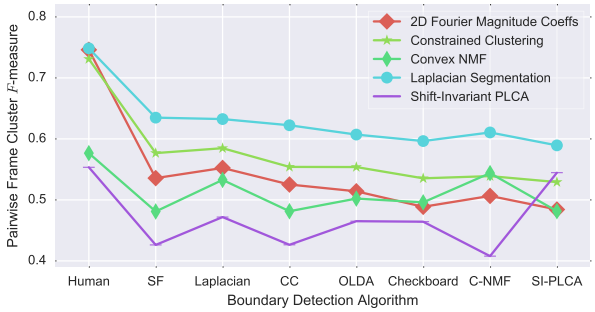


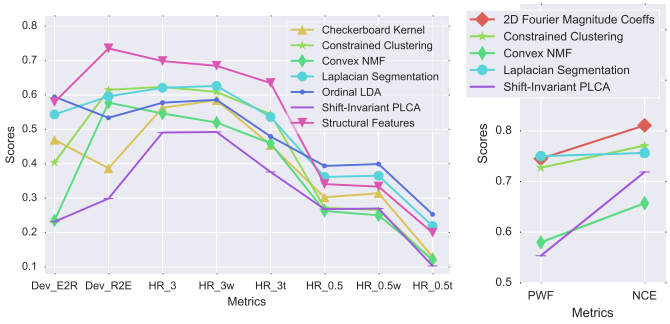
Figure 3: Performance of the structural algorithms contained in MSAF when using different types of previously estimated boundaries as input.

0.01) differently than when using Convex NMF boundaries ($F(4) = 225.05, p < 0.01$). For example, the 2D Fourier Magnitude Coefficients method becomes lower ranked than Convex NMF in the latter case, as can be seen in the plot.

Interesting conclusions can be drawn: first, some structural algorithms are more robust to the quality of the boundaries than others (e.g., 2D-FMC sees a strong impact on its performance when not using annotated boundaries, especially when compared with the Laplacian method). Second, the best performing boundary algorithm will not necessarily make the results of a structural algorithm better, as can be seen in the structural results of C-NMF and SI-PLCA. To exemplify this, note how SF (which tends to outperform all other methods in terms of the standard metric, see Figure 1) produce, in fact, one of the lowest results in structural grouping for the C-NMF method. On the other hand, the Laplacian method (which outputs boundaries that are comparable to the ones by the Checkerboard kernel), obtains results on the structural part that are much better than those by SF. Finally, depending on the boundaries used, structural algorithms will be ranked differently in terms of performance (especially when using annotated boundaries as input). This is something that is not currently taken into account in the MIREX competition, and might be an interesting asset to add in the future for a deeper evaluation of the subtask of structural grouping.

4.3 Evaluation Metrics

In this section we explore the different results obtained by MSAF algorithms when assessed using the available metrics. For boundary detection, the metrics described in Section 2.4 are explored, which are depicted in Figure 4a as "Dev_E2R" for the median deviations from Estimations to References (R2E for the swapped version), and "HR_n" for the Hit Rate with a time window of n seconds (the w and t indicate the weighted and trimmed versions, respectively). The median deviations are divided by 4 in order to normalize the scores within the range of 0 to 1, and then inversed in order to indicate a better performance with a higher score. As expected, scores are significantly different depending on the metric used,



(a) Boundaries

(b) Structure

Figure 4: Scores of MSAF algorithms depending on evaluation metrics.

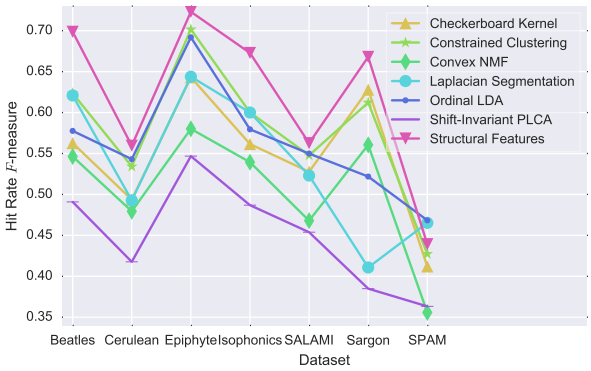


Figure 5: Boundary algorithms' performance depending on dataset.

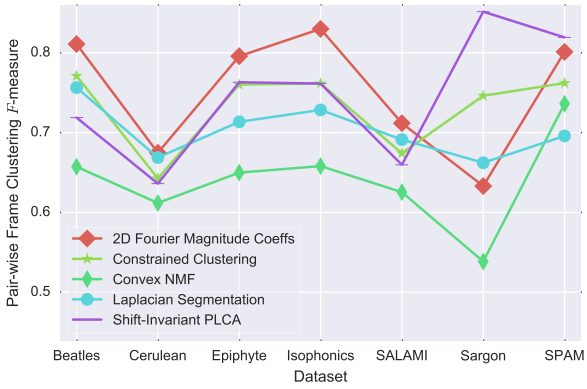


Figure 6: Structural algorithms' performance depending on dataset.

which is confirmed by the two-way ANOVA of the scores with metrics and algorithms as factors (the metric effect is $F(7, 9688) = 458, p < 0.01$). But perhaps more interesting is the fact that some algorithms perform better with some metrics than others (as suggested by the interaction effect of the two-way ANOVA: $F(42, 9688) = 11.24, p < 0.01$). For example, SF is the best algorithm in terms of the Hit Rate with a 3 seconds window, but it is surpassed by the Laplacian and OLDA algorithms when using a shorter window of 0.5, as the Friedman test confirms ($F(6) = 200.13, p < 0.01$) for the ranking of the Hit Rate with 3 seconds, which is different than the one for 0.5 seconds ($F(6) = 210.67, p < 0.01$). Therefore, we can state that, amongst these algorithms, SF is ideal if precise boundary localization is not necessary (HR_3), whereas Laplacian outperforms other methods when this localization has to be accurate (HR_0.5).

In terms of structural algorithms, two metrics (Pairwise Frame Clustering and Normalized Conditional Entropies) are depicted in Figure 4b. A similar behavior occurs here, where algorithms will be ranked differently depending on the metric of choice (Friedman test for the structural algorithms evaluated using the PWF yields $F(4) = 230.11, p < 0.01$ and different ranking than the one for NCE, which results in $F(4) = 215.12, p < 0.01$). Interestingly, all algorithms except Laplacian tend to yield better results when using the NCE scores. Given these results, it would be hard to firmly conclude what the best structural algorithm is for this dataset, since 2D-FMC outperforms Laplacian when evaluated using the NCE scores, which is the opposite behavior when using the PWF.

4.4 Datasets

In Figure 5, the average scores for all boundary algorithms in MSAF on different datasets are depicted. If a dataset contains more than one annotation per track, the first annotator in their JAMS files is used. As expected, different results are obtained depending on the dataset, as confirmed by the two-way ANOVA on the evaluation metric with dataset and algorithm as factors (dataset effect: $F(5, 16604) = 512.18, p < 0.01$). From the plot it can also be seen that some algorithms perform better than others depending on the dataset, which might indicate that they are tuned to solve this problem for a specific type of music. Overall, some datasets seem generally more challenging than others, the SPAM dataset being the one that obtains the worst results, which aligns with the method used to collect their data explained in Section 3.

In terms of the structural algorithms (Figure 6), the two-way ANOVA identifies significant variation, with a relevant effect on the dataset of $F(6, 11875) = 133.16, p < 0.01$. Contrasting with the boundary results, the scores for the SPAM dataset are, on average, one of the highest in terms of structural grouping. This, by itself, warrants discussion, since this dataset was chosen to be particularly challenging from a machine point of view, but only when taking the boundary detection subproblem into account. What these results suggest is that, (i) given the human reference boundaries (which are supposed to be difficult to detect), the structural algorithms perform well at clustering the predefined segments, and/or (ii) we might need a better evaluation metric for the structural subproblem.

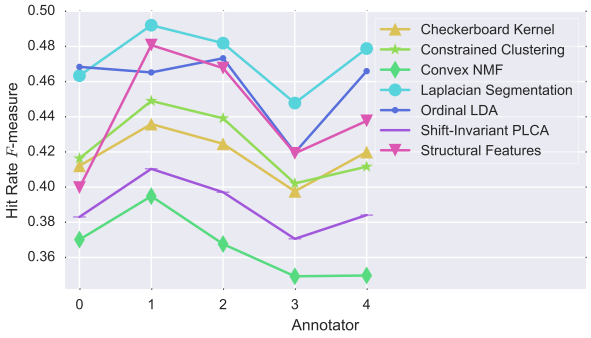


Figure 7: Scores of the boundary algorithms for each human reference in the SPAM dataset.

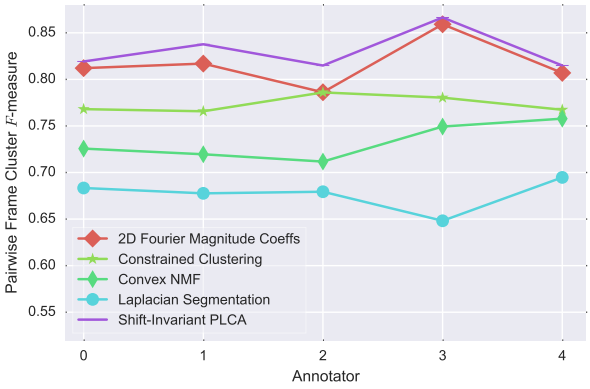


Figure 8: Scores of the structural algorithms for each human reference in the SPAM dataset.

4.5 Human References

The last experiment focuses on analyzing the amount of variation of the MSAF algorithms depending on the annotator used. For this purpose, the five annotations per track of the SPAM dataset become particularly helpful. Starting with the boundaries, we can see in Figure 7 how variable the scores become when using different annotators for the same exact set of audio files. The two-way ANOVA of HR_3 with annotators and algorithms as factors validates this by reporting a significant annotator effect ($F(4, 1705) = 4.05, p < 0.01$). This suggests that subjectivity plays an important role for this subtask, and more than one set of boundaries would be actually valid from a human perspective. Therefore, the idea of a single "ground-truth" for boundary detection can potentially be misleading. Given this amount of variation depending on the annotator, it is interesting to see that the ranking also changes, making it difficult to compare algorithm behaviors. Even though the Laplacian algorithm performs the best for the majority of annotators, it is ranked as second when using annotator 0 by the Friedman test ($F(5) = 21.24, p < 0.01$), while it is ranked as first for the rest of annotators. These results suggest that, given the subjectivity effect in this task, it is indeed important to collect as many references as possible in order to better assess boundary algorithms.

Lastly, the results of the structural algorithms contrast with the previously discussed ones. In this case, there is little dependency on the human reference chosen, as there is no significant effect for the annotator factor in the two-way ANOVA ($F(4, 1225) = 1.08, p = 0.37$), without significant interaction ($F(16, 1225) = 0.93, p = 0.53$). This advocates that the structural grouping subproblem, when applied to a dataset where the grouping is not particularly challenging (as depicted in Figure 6), is not as affected by subjectivity as the boundary detection one, even though further analysis with larger and more challenging datasets —and perhaps with automatically estimated boundaries— should be performed in order to confirm this.

5. CONCLUSIONS

We have presented an open-source framework that facilitates the task of analyzing, assessing, and comparing multiple implementations of structural segmentation algorithms and have employed it to compile a new poly-annotated dataset and to systematically explore the different moving parts of this MIR task. These experiments show that the relative rankings between algorithms tend to change depending on these parts, making it difficult to choose the "best" computational approach. Results also illustrate the problem of ambiguity in this task, and it is our hope that the new SPAM dataset will help researchers to further address this problem. In the future, we wish not only to include more algorithms in this open framework, but to have access to similar frameworks to encourage research on other areas of MIR.

6. REFERENCES

[1] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In *Proc of the 12th International Society of Music Information Retrieval*, pages 591–596, Miami, FL, USA, 2011.

[2] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José Zapata, and Xavier Serra. Essentia: An Audio Analysis Library for Music Information Retrieval. In *Proc. of the 14th International Society for Music Information Retrieval Conference*, pages 493–498, Curitiba, Brazil, 2013.

[3] Michael J. Bruderer. *Perception and Modeling of Segment Boundaries in Popular Music*. PhD thesis, Universiteitsdrukkerij Technische Universiteit Eindhoven, 2008.

[4] Daniel P. W. Ellis. Beat Tracking by Dynamic Programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[5] Daniel P. W. Ellis and Graham E. Poliner. Identifying 'Cover Songs' with Chroma Features and Dynamic Programming Beat Tracking. In *Proc. of the 32nd IEEE International Conference on Acoustics Speech and Signal Processing*, pages 1429–1432, Honolulu, HI, USA, 2007.

[6] Jonathan Foote. Automatic Audio Segmentation Using a Measure Of Audio Novelty. In *Proc. of the IEEE International Conference of Multimedia and Expo*, pages 452–455, New York City, NY, USA, 2000.

[7] Christopher Harte, Mark Sandler, and Martin Gasser. Detecting Harmonic Change in Musical Audio. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, pages 21–26, Santa Barbara, CA, USA, 2006. ACM Press.

[8] André Holzapfel, Matthew E. P. Davies, José R. Zapata, J. Lobato Oliveira, and Fabien Gouyon. Selective Sampling for Beat Tracking Evaluation. *IEEE Transactions On Audio, Speech, And Language Processing*, 20(9):2539–2548, 2012.

[9] Eric J. Humphrey, Justin Salamon, Oriol Nieto, Jon Forsyth, Rachel M. Bittner, and Juan P. Bello. JAMS: A JSON Annotated Music Specification for Reproducible MIR Research. In *Proc. of the 15th International Society for Music Information Retrieval Conference*, pages 591–596, Taipei, Taiwan, 2014.

[10] Mark Levy and Mark Sandler. Structural Segmentation of Musical Audio by Constrained Clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):318–326, feb 2008.

[11] Hanna Lukashevich. Towards Quantitative Measures of Evaluating Song Segmentation. In *Proc. of the 10th International Society of Music Information Retrieval*, pages 375–380, Philadelphia, PA, USA, 2008.

[12] Brian McFee and Daniel P. W. Ellis. Analyzing Song Structure with Spectral Clustering. In *Proc. of the 15th International Society for Music Information Retrieval Conference*, pages 405–410, Taipei, Taiwan, 2014.

[13] Brian McFee and Daniel P. W. Ellis. Learnign to Segment Songs With Ordinal Linear Discriminant Analysis. In *Proc. of the 39th IEEE International Conference on Acoustics Speech and Signal Processing*, pages 5197–5201, Florence, Italy, 2014.

[14] Brian Mcfee, Colin Raffel, Dawen Liang, Daniel P. W. Ellis, Matt Mcvicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and Music Signal Analysis in Python. In *Proc. of the 14th Python in Science Conference*, pages 1–7, Austin, TX, USA, 2015.

[15] Oriol Nieto and Juan Pablo Bello. Music Segment Similarity Using 2D-Fourier Magnitude Coefficients. In *Proc. of the 39th IEEE International Conference on Acoustics Speech and Signal Processing*, pages 664–668, Florence, Italy, 2014.

[16] Oriol Nieto and Morwaread M. Farbood. Identifying Polyphonic Patterns From Audio Recordings Using Music Segmentation Techniques. In *Proc. of the 15th International Society for Music Information Retrieval Conference*, pages 411–416, Taipei, Taiwan, 2014.

[17] Oriol Nieto, Morwaread M. Farbood, Tristan Jehan, and Juan Pablo Bello. Perceptual Analysis of the F-measure for Evaluating Section Boundaries in Music. In *Proc. of the 15th International Society for Music Information Retrieval Conference*, pages 265–270, Taipei, Taiwan, 2014.

[18] Oriol Nieto and Tristan Jehan. Convex Non-Negative Matrix Factorization For Automatic Music Structure Identification. In *Proc. of the 38th IEEE International Conference on Acoustics Speech and Signal Processing*, pages 236–240, Vancouver, Canada, 2013.

[19] Jouni Paulus, Meinard Müller, and Anssi Klapuri. Audio-Based Music Structure Analysis. In *Proc of the 11th International Society of Music Information Retrieval*, pages 625–636, Utrecht, Netherlands, 2010.

[20] Colin Raffel, Brian Mcfee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis. mir_eval: A Transparent Implementation of Common MIR Metrics. In *Proc. of the 15th International Society for Music Information Retrieval Conference*, pages 367–372, Taipei, Taiwan, 2014.

[21] Joan Serrà, Meinard Müller, Peter Grosche, and Josep Lluís Arcos. Unsupervised Music Structure Annotation by Time Series Structure Features and Segment Similarity. *IEEE Transactions on Multimedia, Special Issue on Music Data Mining*, 16(5):1229 – 1240, 2014.

[22] Jordan B. Smith, J. Ashley Burgoyne, Ichiro Fujinaga, David De Roure, and J. Stephen Downie. Design and Creation of a Large-Scale Database of Structural Annotations. In *Proc. of the 12th International Society of Music Information Retrieval*, pages 555–560, Miami, FL, USA, 2011.

[23] Jordan B. L. Smith and Elaine Chew. A Meta-Analysis of the MIREX Structure Segmentation Task. In *Proc. of the 14th International Society for Music Information Retrieval Conference*, Curitiba, Brazil, 2013.

[24] Douglas Turnbull, Gert Lanckriet, Elias Pampalk, and Masataka Goto. A Supervised Approach for Detecting Boundaries in Music Using Difference Features and Boosting. In *Proc. of the 5th International Society of Music Information Retrieval*, pages 42–49, Vienna, Austria, 2007.

[25] George Tzanetakis and Perry Cook. MARSYAS: a framework for audio analysis. *Organised Sound*, 4(3):169–175, 2000.

[26] Ron Weiss and Juan Pablo Bello. Unsupervised Discovery of Temporal Structure in Music. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1240–1251, 2011.

USING PRIORS TO IMPROVE ESTIMATES OF MUSIC STRUCTURE

Jordan B. L. Smith **Masataka Goto**
National Institute of Advanced Industrial Science and Technology (AIST), Japan
jordan.smith@aist.go.jp, m.goto@aist.go.jp

ABSTRACT

Existing collections of annotations of musical structure possess many strong regularities: for example, the lengths of segments are approximately log-normally distributed, as is the number of segments per annotation; and the lengths of two adjacent segments are highly likely to have an integer ratio. Since many aspects of structural annotations are highly regular, but few of these regularities are taken into account by current algorithms, we propose several methods of improving predictions of musical structure by using their likelihood according to prior distributions. We test the use of priors to improve a committee of basic segmentation algorithms, and to improve a committee of cutting-edge approaches submitted to MIREX. In both cases, we are unable to improve on the best committee member, meaning that our proposed approach is outperformed by simple parameter tuning. The same negative result was found despite incorporating the priors in multiple ways. To explain the result, we show that although there is a correlation overall between output accuracy and prior likelihood, the weakness of the correlation in the high-likelihood region makes the proposed method infeasible. We suggest that to improve on the state of the art using prior likelihoods, these ought to be incorporated at a deeper level of the algorithm.

1. INTRODUCTION

One reason that the perception of structure in music is such a complex and compelling phenomenon is that it is a combination of 'bottom-up' and 'top-down' processes. It is bottom-up in the sense that a listener first performs grouping on short timescales before understanding the grouping at large timescales, but it is top-down in the sense that one has global expectations that can affect the way one perceives the music. For example, when hearing a new pop song for the first time, we expect there to be a chorus; even on our first hearing, we may identify the chorus partway through a song and already expect it to repeat later. After hearing a verse and a chorus, each 32 beats long, we may expect the bridge to be the same length when it starts.

© Jordan B. L. Smith, Masataka Goto. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jordan B. L. Smith, Masataka Goto. "Using Priors To Improve Estimates of Music Structure", 17th International Society for Music Information Retrieval Conference, 2016.

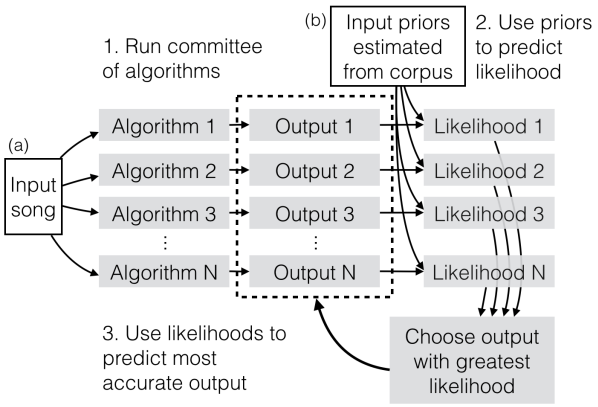


Figure 1. Proposed system overview.

This is important to recognize, since structure is ambiguous: for any piece, there are often multiple ways of interpreting it. As a result, we might never expect a purely bottom-up approach to be 100% correct; we need to also model the top-down influence, or what the listener 'brings' to the analysis.

For instance, consider the following analysis of a piece of music, with the A sections each 10 seconds long, and B 200 seconds long:

```
[-A-|-A-|-A-|------------B------------]
```

Even without knowing the piece of music, we can tell this is an unlikely analysis; it seems wrong to have the segments of the piece sized so asymmetrically. This example hints that the space of *plausible* analyses is limited (even if it is huge), and that listeners' intuitions about these limits inform the annotation process. Is there a way to embed such intuitions into music structure analysis algorithms? Can we employ a kind of 'top-down' critic to assess the likelihood of a given analysis?

We propose a system to accomplish this, illustrated in Figure 1. The inputs to the system are: (a) a song to analyze, and (b) a set of probability density functions (PDFs) estimated from a corpus of annotations. The input song is analyzed by a set of algorithms (step 1); the prior likelihood of each output is computed (step 2); and the estimated description with the highest prior likelihood is chosen (step 3). In contrast to the usual parameter tuning approach, in which a single parameter setting is fixed after evaluating performance over a corpus of songs, in our approach parameters can be tuned for each song, on the basis of prior likelihood.

1.1 Related work

Most algorithms do at least some domain knowledge-based tuning, by putting a lower and/or upper bound on the length of segments, or by filtering features to reduce variations at certain timescales. These are important steps because, although musical structure is hierarchical, algorithms rarely attempt to predict this hierarchy and are evaluated only at a single level. (This status quo has been challenged by [5].)

However, a few algorithms have made greater use of domain knowledge, and their success has been noteworthy. Among the first optimization-based approaches to structure analyis was [7], who explicitly sought to define (in a top-down way) what constitutes a "good" analysis (i.e., one more likely to be in the space of plausible solutions). Later, [10] estimated the median segment length of a piece and used this as the preferred segment length in its search for an optimal segmentation; at the time, their algorithm outperformed the leader at MIREX. The AutoMashUpper system also uses a cost-based approach, rewarding solutions with "good" segment durations of 2, 4, or 8 downbeats, and penalizing ones deemed less likely, like 7 or 9 [2]. In the symbolic domain, [9] also used a cost-minimization approach, with costs increased for segments of unlikely duration or unlikely melodic contour; on one dataset, the approach outperformed a pack of leading melodic-segmentation algorithms.

The most direct way to use domain knowledge is to use supervised learning. Two examples include [14, 15], who each used machine learning to classify short excerpts as boundaries or not based on their resemblance to other short excerpts known to be boundaries. The performance of [15] exceeded the best MIREX result by nearly 10% f-measure for both 0.5- and 3-second thresholds, an enormous achievement.

Our intuition about what constrains the space of plausible analyses, as well as the success of previous algorithms in using domain knowledge and priors learned from corpora, suggest that taking full advantage of this prior knowledge is essential to designing effective algorithms.

In the next section, we survey some of these regularities, and explore the extent to which prior algorithms adhere to them. We detail our proposed algorithms and report our experimental results in Section 3. Alas, despite the solid foundations, no approach will be found to work. The significance of this negative result, and possible explanations for it, are discussed in Section 4.

2. REGULARITIES

In this section we briefly survey some regularities found in the SALAMI corpus of annotations [12], and describe the relationship between these regularities and algorithms that have participated in MIREX campaigns from 2012–14.

Although the time scale of the SALAMI annotations was not explicitly constrained in the Annotator's Guide [1], the length of annotated segments in the SALAMI corpus is roughly log-normally distributed, for both hierarchical levels. Figure 2a shows the PDF of the log segment length ($P(\log L_i)$) for the large and small hierarchical levels in SALAMI; this and all other PDFs in this paper were found using kernel density estimation (KDE). The number of segments within a piece (N) is also log-normally distributed (Figure 2b). If we take the log ratio of each segment's length to the median length of segments within that piece ($\log(L_i/L_{med})$), we obtain a PDF strongly concentrated at $\log(1) = 0$ (see Figure 2c), with additional spikes near ± 0.693, or $\log(2)$ and $\log(1/2)$, for segments of twice or half the median length. There is even more detail if we look at the log length ratio between adjacent segments ($\log(L_i/L_{i+1})$), a histogram of which is shown in Figure 3. Note that all the prominent peaks occur at ratios of small numbers. This makes sense if we consider that segments are usually a whole number of measures long. These properties are not specific to SALAMI annotations; similar distributions were reported by [1] for a completely different corpus of annotations.

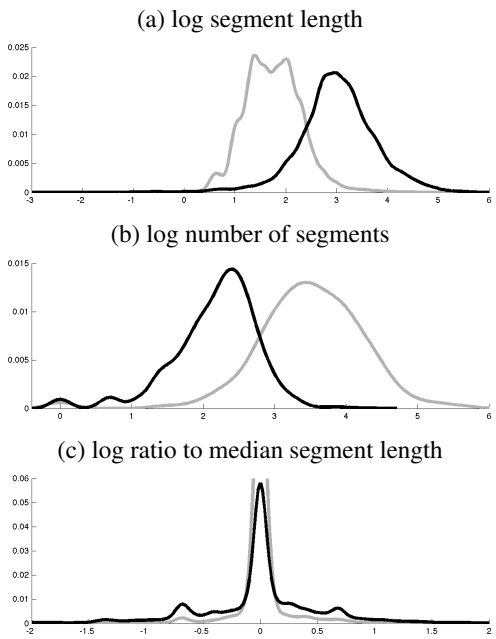


Figure 2. Estimated PDFs of three properties for annotations in SALAMI corpus; x-axis gives the value of the property, y-axis gives its relative probability. PDFs from large-scale segments shown in black, from small-scale segments in gray. In (c), the vertical axis is clipped to show detail; the gray line extends upwards to just over 0.1.

How closely do algorithms model these properties of the annotations? We looked at three years of participants in the MIREX Structural Segmentation task, 2012–14, and estimated PDFs for the same properties. Some examples are shown in Figure 4. Figure 4a shows PDFs for segment length estimated from each algorithm individually: some hew closely to the ground truth, but the majority underestimate the mean segment length. (Since precision is harder to achieve than recall, oversegmentation usually leads to

[1] Available at `http://ddmal.music.mcgill.ca/research/salami/annotations`.

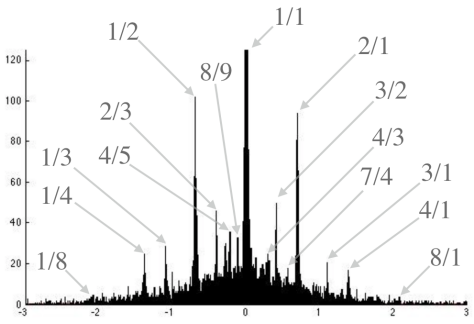


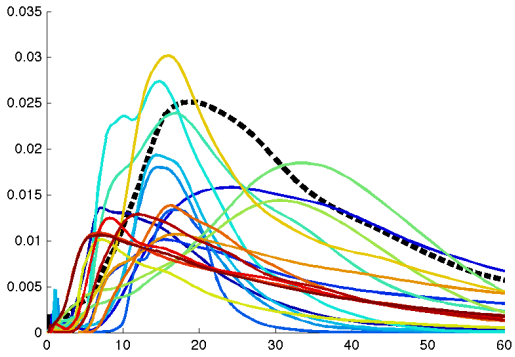
Figure 3. Histogram of $\log(L_i/L_{i+1})$ estimated from SALAMI annotations (y-axis truncated at 1/3 maximum). As shown, nearly every spike represents an integer ratio of segment lengths.

better evaluation scores. [13])

If segment length seems like a weak prior, consider instead Figure 4b, which compares PDFs for the log ratio of adjacent segment length. The characteristic side-lobes representing the high frequency of half- and double-length segments are prominent in only two of the algorithms, RBH1 and RBH3 (2013). This is likely because the algorithm [8] expects boundaries to occur on an 8-measure metrical grid, and snaps estimated boundaries to this grid. Performance (evaluated with f-measure and 3-second threshold) was mixed: RBH1 was close to the state of the art in 2013, while RBH3 was below-average. [2] On the other hand, the next-strongest side-lobes belong to SUG1, the second-best algorithm overall. SUG1 uses a convolutional neural network to classify short excerpts as containing boundaries or not; the method ends by picking peaks from a boundary-likelihood curve, without post-processing [15]. Although SUG1 obviously learns from annotated data, it learns from low-level features (a mel spectrogram) rather than high-level attributes like segment length ratios.

Does the fitness of the algorithms to the SALAMI-derived priors actually have an impact on their performance? We found this to be true by looking at the correlation between algorithm performance and prior likelihood. We took the output of the 18 unique segmentation algorithms that participated in MIREX from 2012–14 [3], and for each algorithm, computed the average log-likelihood of its estimated segments based on the KDE-derived PDFs from SALAMI. We also took the average performance of the algorithms on the three boundary retrieval metrics (f-measure, precision, and recall) with a threshold of 3 seconds. Figure 5 shows the correlation between the mean log-likelihoods (of various segment properties) and the evaluation metrics. There is a weak to moderate correlation between likelihood and f-measure for each of these properties, usually attributable to a strong correlation between likelihood and either precision or recall.

(a) PDFs of segment duration (in seconds)



(b) PDFs of log ratio of length of adjacent segments

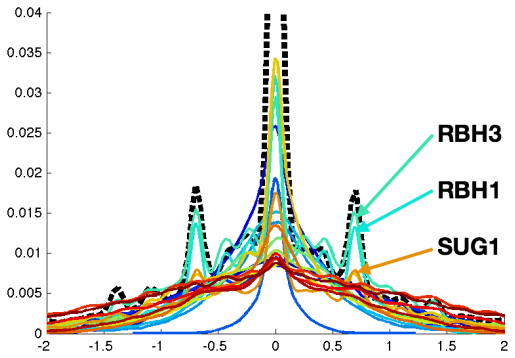


Figure 4. PDFs of properties estimated from SALAMI annotations (black dotted line) and from the output of MIREX algorithms (each algorithm in a different colour).

We have seen that most algorithms deviate substantially from the corpus; the algorithms' descriptions simply don't 'look' like the ground truth. Also, there is some evidence that an algorithm's accuracy is related to the prior likelihood of its output. Hence, it seems reasonable to ask: can we improve on these algorithms, or any set of algorithms, by maximizing their fitness to the priors?

3. USING PRIORS TO IMPROVE A COMMITTEE

Our proposed system is simple: for a single audio file, (1) run several existing structural analysis algorithms, (2) compute the log-likelihood of each prediction with respect to a corpus, and (3) choose the output that maximizes this. (See Figure 1.) For each of these steps, there are many ways to proceed.

3.1 Assembling a committee

We assembled two committees of algorithms: a committee of multiple parameterizations of two basic approaches (Foote [3] and Serra et al. [11]), and a committee of approaches drawn from MIREX. In the first case, we test

[2] Evaluation results in this paper differ from those reported at MIREX, since we re-evaluated the algorithm output with a 5-second 'warm-up' applied: boundaries within the first and final 5 seconds of pieces were ignored. This leads to lower results overall but better differentiation between algorithms.

[3] Of the 24 participants in these years, 5 used the same segmentation algorithm as another, and the data for one (FK2) were posted later than the others, and were excluded.

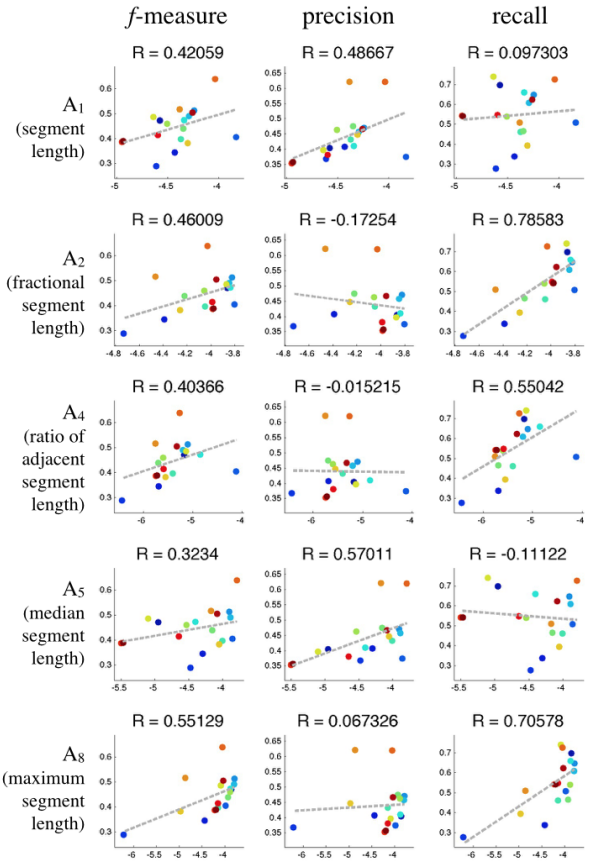


Figure 5. Scatter plots of mean log likelihood of algorithm output (x-axis) and f-measure, precision and recall (y-axis), for 18 MIREX segmentation algorithms, 2012–14. Correlations (Pearson's R) and lines of best fit are shown.

whether a set of more strictly bottom-up segmentation approaches can be improved with the top-down likelihood-maximizing process; in the second case, we test whether a set of already-optimized algorithms can also be improved.

Foote uses a checkerboard kernel to identify discontinuities between homogeneous regions in a self-similarity matrix (SSM), and his remains a classic approach to segmentation. Although surpassed in evaluations such as MIREX, the simplicity and effectiveness of the algorithm means it is still commonly used as a model to improve upon (e.g., see [4]). In contrast to Foote, Serra et al. [11] aim to use both repetition and novelty to locate boundaries. In practice, both algorithms require several design choices: which audio features to use, what amount of smoothing to apply, etc. We ran each algorithm with a small factorial range of settings, including three features (HPCP, MFCCs, and tempograms), for a total of 40 unique settings—hence, 40 committee members. We ran the algorithms on 773 songs within the public SALAMI corpus (version 1.9). Feature extraction and algorithm computation were both easily handled using MSAF [6].

The output of the algorithms that participated in MIREX is publically available, so we simply assembled it, along with the reported algorithm performance, for a MIREX committee of 23 members. We restricted ourselves to the SALAMI portion of the MIREX evaluation, which overlaps significantly with the public half of SALAMI but is not identical.

3.2 Computing likelihoods

We looked at the distribution of several attributes of the SALAMI corpus, listed below. Of these, A_{1-4} are estimated on a per-segment basis and A_{5-9} are global attributes of a description.

- A_1 Segment length (L_i)
- A_2 Fractional segment length (L_i / song length)
- A_3 Ratio of L_i to median segment length
- A_4 Ratio of adjacent segment lengths (L_i/L_{i+1})
- A_5 Median segment length (median of L_i)
- A_6 Number of segments
- A_7 Minimum segment length
- A_8 Maximum segment length
- A_9 Standard deviation of segment length

For attributes A_{1-4}, we took the average across segments. Although log likelihoods are designed to be summed, taking the sum of $\log P(L_i)$ would punish descriptions with more boundaries, regardless of how probable the segments are. (In fact, we did test taking the sum instead of the mean across segments, and the results were generally much poorer.)

3.3 Electing a winner

Once we have computed all of the log likelihoods, how should they be combined, and how should we use these values to elect an answer? Without any *a priori* reason to prefer one over another, we tested multiple approaches:

- choose the description that maximizes the likelihood of attribute A_i;
- choose the description that maximizes a summary statistic over all attributes;
- use a linear model to predict f-measure based on the likelihoods;
- use a linear model with interactions;
- use a quadratic model.

As two summary statistics, we used the sum and the minimum of the log likelihoods of A_i. Using the sum optimizes the general fitness; using the minimum penalizes descriptions with any unlikely attributes.

3.4 Experiment and results

With 5-fold cross-validation, we tested all versions of the algorithm, using both the Foote/Serra and MIREX committees. For each fold, the prior PDFs were estimated only using annotations from the training set.

As a baseline, we used simple parameter tuning: i.e., simply pick the committee member with the greatest average success on the training set. For reference, we also computed the mean f-measure of all committee members, and

Attribute	f (3 sec)	f (0.5 sec)
A_1 (mean)	0.4230	0.1051
A_2 (mean)	0.4156	0.0958
A_3 (mean)	0.4176	0.1140
A_4 (mean)	0.4194	0.1072
A_5	0.3597	0.0863
A_6	0.3781	0.0991
A_7	0.0603	0.0124
A_8	0.3907	0.0961
A_9	0.3956	0.0950
$\sum A_i$	0.4260	0.1093
Min A_i	0.4206	0.1046
Linear model	0.4399	0.0845
Interactions model	0.4451	0.0688
Quadratic model	**0.4494**	0.0739
Baseline	0.4439	**0.1151**
Committee mean	0.2826	0.0691
Theoretical max	0.6015	0.2572

Table 1. Average f-measure (at two thresholds) achieved by different decision criteria for Foote-and-Serra committee.

Attribute	f (3 sec)	f (0.5 sec)
A_1 (mean)	0.6273	0.2733
A_2 (mean)	0.3487	0.0996
A_3 (mean)	0.3487	0.0996
A_4 (mean)	0.3487	0.0996
A_5	0.3916	0.1385
A_6	0.3768	0.1594
A_7	0.3487	0.0996
A_8	0.4662	0.1356
A_9	0.4233	0.1514
$\sum A_i$	0.6273	0.2733
Min A_i	0.6273	0.2733
Linear model	0.5591	0.4005
Interactions model	0.6273	0.4005
Quadratic model	0.6273	0.4005
Baseline	**0.6273**	**0.4005**
Committee mean	0.2826	0.0691
Theoretical max	0.7345	0.5157

Table 2. Average f-measure (at two thresholds) achieved by different decision criteria for MIREX committee.

the theoretical maximum—i.e., the average of the highest-scoring estimates for each song.

The results are shown in Tables 1 and 2. Among all the variations of the proposed method, there were only two instances that surpassed the baseline: the quadratic and interactions models for the Foote-Serra committee, with a 3-second tolerance level. They surpassed the baseline f-measure by 0.0055 and 0.0012, respectively. Given the number of trials conducted, this small amount of success could easily have come by chance.

4. DISCUSSION

Negative results are not normally conclusive: in this case, the reader may suspect that with a small twist, our proposed method may yet succeed. For example, what if we examined subsets by genre, or considered conditional probabilities? In fact, this process of tweaking is how our experiments came about. Our first effort to solve the problem used a small committee of solely Foote-based algorithms, and a set of four log likelihoods. When tests with this initial system gave us a negative result, we tried varying each of the parts of the system—adding more members to the committee, including more PDFs, using increasingly sophisticated regression approaches—until we had assembled the large-scale experiment reported here. And we conducted several more informal tests—looking at subsets of the data, varying the method of characterizing the PDFs (instead of with KDE, they can be modelled with plain histograms, or normal curves can be fitted to some distributions), looking at subcommittees (e.g., removing top-performing and low-performing outlier members) and computing two-dimensional priors (to model, for example, the fact that segment length is not independent of when a segment begins)—all to no avail.

The consistency of the negative result—only two trials out of 56 exceeded the baseline, and only by the slimmest of margins—suggests a dead end. But in order to draw conclusions from this negative result, we must try to understand *why* the approach failed.

Earlier, in Figure 5, we saw that algorithm performance could, over many trials, correlate with the prior likelihood of their output. But what happens when we dig deeper and look at the relationship between each individual output's correctness and its likelihood, as in Figure 6? On the one hand, there is a clear positive trend overall, since there are no examples in the upper-left corner—that is, there are no predictions that have low likelihood but that are close to correct. And the examples with the highest f-measure are also *among* those with the highest likelihoods. Thanks to this relationship, the committees can, despite the noise, generally choose an output that is at least, or slightly better than, the committee's average; that's why, in Tables 1 and 2, nearly all of the algorithms exceeded the average result of the committee.

On the other hand, trying to *find* the high-f-measure predictions based on their prior likelihood is clearly futile when we consider only the rightmost region of the plot, a zoomed-in portion of which is shown in the lower part of Figure 6. Even these predictions, with the greatest fit to the priors, range widely in accuracy: there are plenty above the baseline (0.44), but also plenty below it, including a large number of predictions that contain zero correct boundaries. Figure 6 shows that having a high log-likelihood is a necessary but not sufficient condition to be correct, and it is a condition that most algorithms already achieve.

The uppermost points in Figure 6 represent a few lucky, perfect estimates of the true structure. Their distribution reveals another important point: that although the prior PDFs derive from the ground truth, the prior likelihood of

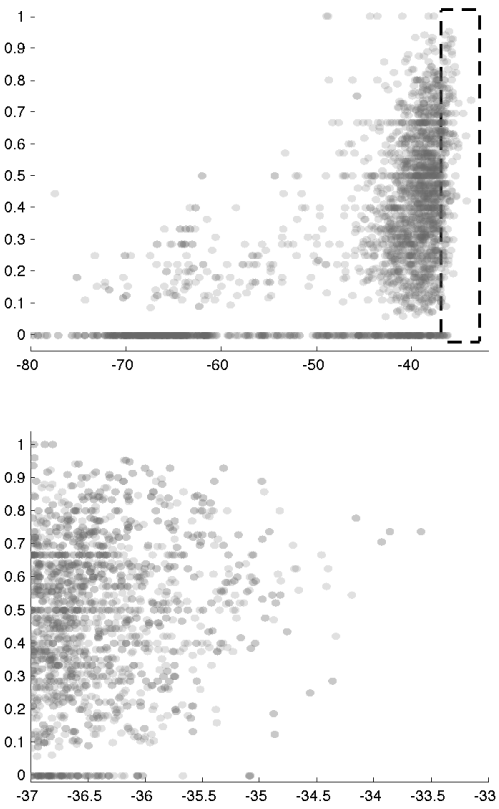


Figure 6. Above: scatter plot of $\sum \log P(A_i)$ (x-axis) vs. f-measure (y-axis, 3-second threshold) of algorithm outputs for Foote/Serra committee on all data. (The heavy horizontal lines are caused by the fact that f-measure is often the product of common fractions.) Below: inset of plot indicated by rectangle.

many annotations is moderate. The fat tails of the PDFs in Figure 2 represent a large set of descriptions that are unlikely to ever be predicted by a prior likelihood-based approach. For example, consider the analyses shown in Figure 7. [4] One algorithm achieved a perfect f-measure (with 3-second threshold), and the likelihood of the description (measured with respect to attribute A_4) was close to that of the annotated description. But a second estimate had a slightly higher prior likelihood thanks to its more consistent segment lengths, and a very poor f-measure.

To sum up the factors that appear to limit the effectiveness of our approach:

1. Although a high f-measure tends to come with a higher prior likelihood, the reverse is not true: plenty of highly probable descriptions are very poor.

2. The moderate correlation between algorithm success and prior likelihood is irrelevant, since we are interested only in the high-likelihood region of estimated descriptions.

[4] Although the MIREX data are anonymized, many songs can be identified by comparing the ground truth to known datasets. [13]

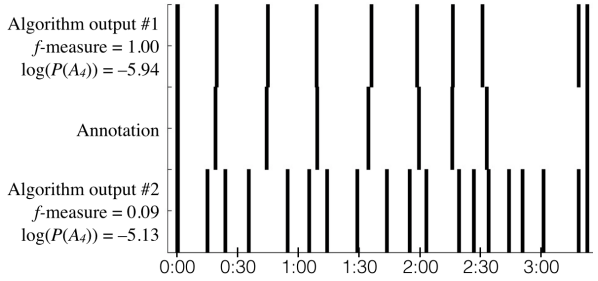


Figure 7. Two algorithmic estimates, compared to the ground truth (middle). The estimates differ somewhat in the likelihood of A_4 (adjacent segment length), but drastically in f-measure. The song is "Rock With You" by Michael Jackson, SALAMI ID 1616.

3. Among high-likelihood descriptions, the correlation between success and likelihood is much weaker: many likely descriptions are poor, and many annotations have low likelihood.

5. CONCLUSION AND FUTURE WORK

We proposed and tested a novel committee-based approach to structural analysis. We motivated the approach by discussing the strong regularities displayed by annotations of music structure. But after a long stretch of negative results, we have concluded that the approach seems unviable: the relationship between a description's prior likelihood and its evaluated score seems to be too weak, especially in the high-likelihood region we are interested in.

We began the article by pointing out some mismatches between the properties of algorithmic estimates of structure and the ground truth, and we suggested that this may be because algorithms do not model top-down factors in perception. For a listener, top-down factors interact with bottom-up factors; in contrast, our algorithm applies bottom-up considerations first (by collecting the committee of estimates), and then applies the top-down considerations *post hoc*. This may be the central weakness of our algorithm. Perhaps, if the top-down influence were modelled earlier on, an estimate like the top one in Figure 7 could be fine-tuned, the boundaries shifted slightly to give a more probable output, rather than rejected early on because of its low likelihood. One algorithm that is ready to test this as future work is the optimization approach of [10]. Although the authors model only a few basic priors, it could be improved by including more.

6. ACKNOWLEDGEMENTS

A version of this work was shown to participants at the Dagstuhl Workshop on Computational Music Structure Analysis. We are very grateful to the attendees for their helpful suggestions, and to the workshop organizers and the Leibniz Center for Informatics for hosting us. This work was supported in part by OngaCREST, CREST, JST.

7. REFERENCES

[1] Frédéric Bimbot, Gabriel Sargent, Emmanuel Deruty, Corentin Guichaoua, and Emmanuel Vincent. Semiotic description of music structure: An introduction to the Quaero/Metiss structural annotations. In *Proceedings of the AES 53rd Conference on Semantic Audio*, 2014.

[2] Matthew E. P. Davies, Philippe Hamel, Kazuyoshi Yoshii, and Masataka Goto. AutoMashUpper: Automatic creation of multi-song music mashups. *IEEE Transactions on Audio, Speech, and Language Processing*, 22(12):1726–1737, 2014.

[3] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of the IEEE International Conference on Multimedia & Expo*, pages 452–455, 2000.

[4] Florian Kaiser and Geoffroy Peeters. A simple fusion method of state and sequence segmentation for music structure discovery. In *Proceedings of ISMIR*, pages 257–262, Curitiba, Brazil, 2013.

[5] Brian McFee, Oriol Nieto, and Juan Pablo Bello. Hierarchical evaluation of segment boundary detection. In *Proceedings of ISMIR*, Málaga, Spain, 2015.

[6] Oriol Nieto and Juan Pablo Bello. Systematic exploration of computational music structure research. In *Proceedings of ISMIR*, New York, NY, USA, 2016.

[7] Jouni Paulus and Anssi Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Transactions on Audio, Speech & Language Processing*, 17(6):1159–1170, 2009.

[8] Bruno Rocha, Niels Bogaards, and Aline Honingh. Segmentation and timbre similarity in electronic dance music. In *Proceedings of the Sound and Music Computing Conference*, pages 754–761, Stockholm, Sweden, 2013.

[9] Marcelo Rodríguez-López, Anja Volk, and Dimitrios Bountouridis. Multi-strategy segmentation of melodies. In *Proceedings of ISMIR*, pages 207–212, Taipei, Taiwan, November 2014.

[10] Gabriel Sargent, Frédéric Bimbot, and Emmanuel Vincent. A regularity-constrained Viterbi algorithm and its application to the structural segmentation of songs. In *Proceedings of ISMIR*, pages 483–488, Miami, FL, USA, 2011.

[11] Joan Serrà, Meinard Müller, Peter Grosche, and Josep Ll. Arcos. Unsupervised detection of music boundaries by time series structure features. In *Proceedings of the AAAI International Conference on Artificial Intelligence*, pages 1613–1619, Toronto, Canada, 2012.

[12] Jordan B. L. Smith, J. Ashley Burgoyne, Ichiro Fujinaga, David De Roure, and J. Stephen Downie. Design and creation of a large-scale database of structural annotations. In *Proceedings of ISMIR*, pages 555–560, Miami, FL, USA, 2011.

[13] Jordan B. L. Smith and Elaine Chew. A meta-analysis of the MIREX Structure Segmentation task. In *Proceedings of ISMIR*, pages 251–256, Curitiba, Brazil, 2013.

[14] Douglas Turnbull, Gert Lanckriet, Elias Pampalk, and Masataka Goto. A supervised approach for detecting boundaries in music using difference features and boosting. In *Proceedings of ISMIR*, pages 51–54, Vienna, Austria, 2007.

[15] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary detection in music structure analysis using convolutional neural networks. In *Proceedings of ISMIR*, pages 417–422, Taipei, Taiwan, November 2014.

MUSIC STRUCTURAL SEGMENTATION ACROSS GENRES WITH GAMMATONE FEATURES

Mi Tian, Mark B. Sandler
Centre for Digital Music, Queen Mary University of London
{m.tian, mark.sandler}@qmul.ac.uk

ABSTRACT

Music structural segmentation (MSS) studies to date mainly employ audio features describing the timbral, harmonic or rhythmic aspects of the music and are evaluated using datasets consisting primarily of Western music. A new dataset of Chinese traditional Jingju music with structural annotations is introduced in this paper to complement the existing evaluation framework. We discuss some statistics of the annotations analysing the inter-annotator agreements. We present two auditory features derived from the Gammatone filters based respectively on the cepstral analysis and the spectral contrast description. The Gammatone features and two commonly used features, Mel-frequency cepstral coefficients (MFCCs) and chromagram, are evaluated on the Jingju dataset as well as two existing used ones using several state-of-the-art algorithms. The investigated Gammatone features outperform MFCCs and chromagram when evaluated on the Jingju dataset and show similar performance with the Western datasets. We identify the presented Gammatone features as effective structure descriptors, especially for music lacking notable timbral or harmonic sectional variations. Results also indicate that the design of audio features and segmentation algorithms should be adapted to specific music genres to interpret individual structural patterns.

1. INTRODUCTION

Music is primarily an event-based phenomenon comprising a series of musical elements such as melody and harmony that unfold in time. Both human listening and analysis activities can identify the musical structure of a piece that divides its contents into sections each featuring their own characteristics. *Music information retrieval* (MIR) is a research field concerning the extraction of meaningful information from the music content for real world purposes. As a popular MIR task, *Music structural segmentation* (MSS) concerns dividing music into structural parts by giving it boundaries, hence generating high-level music descriptions.

© Mi Tian, Mark B. Sandler. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Mi Tian, Mark B. Sandler. "Music Structural Segmentation Across Genres with Gammatone Features", 17th International Society for Music Information Retrieval Conference, 2016.

Datasets used to evaluate MIR systems consist mainly of Western popular or contemporary music. The acquisition of non-Western datasets can be highly valuable to combat the Western bias within current MIR paradigm [4, 18]. Understandings of the music structure can be genre-dependent and a segmentation algorithm designed for one corpus may have vague assumptions for another. Smith studied several segmentation algorithms and suggests that algorithms designed for the structural analysis of Western pop music are widely applicable beyond the Western context [21]. Nonetheless, the evaluation corpora used in [21] are still Western centric and collected on a basis of general structural coherence. One primary motivation of this work is to include more challenging genres to analyse the music structure beyond the Western scenario. One of these genres is Jingju, also known as Beijing Opera or Peking Opera, which is one of the most representative genres of Chinese traditional music. An analytical discovery of its song structure will greatly assist its popularisation and subsequent applications such as browsing and indexing. Although it offers intriguing research topics to challenge the existing MIR tools, little work has been done to understand its content using computational methods until very recent years with its structural analysis largely absent from the literature [1]. It should be noted that the song structure has to be differentiated from the structure of a full Jingju play, where the former relates to only the arias part of the latter [27]. In this paper, we include Jingju as a new genre in the MSS study and address the analysis of its song structure.

Various audio features have been used for the structural description of music capturing mainly its harmonic, timbral or rhythmic aspects [16]. A *chromagram* [7], also denoted *Harmonic pitch class profiles* (HPCP), along with its many variants are the most popular features for the structural analysis of Western pop music. The chromagram is a B-dimensional vector representation denoting the energy of each semitone distributed in a chromatic scale, where B is the number of semitones in an octave. The *Mel frequency cepstral coefficients* (MFCCs) feature is among the most used timbre descriptors for MSS studies. It models the shape of the spectral envelope by describing the frequency spectrum aligned on a Mel scale [10]. Rhythmicity may identify music structure beyond the timbral or harmonic variations. It is however much less employed compared to MFCCs and the chroma features [16]. The classical time-frequency (TF) representations used for timbre research are based on the short-time Fourier transform

(STFT) of the audio signal. Although MSS is considered a high-level task involving human perception, auditory cues are barely incorporated in the commonly used audio features. As the second main motivation of this work, we will explore novel timbre features modelling the frequency resolution of the human auditory system to describe the music structure.

This paper is organised as follows. Section 2 introduces the background of Jingju music and the related work on auditory features. We present the new Jingju dataset in Section 3. The investigated Gammatone features are presented in Section 4 and are evaluated in a MSS experiment introduced in Section 5. Section 6 is devoted to analyse the results. Finally we conclude this work in Section 7.

2. BACKGROUND

Compared to the spectrogram which displays the TF components of an audio signal mapped to their physical intensity levels, auditory representations attempt to emphasise its perceptually salient aspects. The *Gammatone* function has been widely used to derive the TF representation modelling human auditory responses of sound and has various applications in research areas such as auditory scene analysis, speech recognition and audio classification [19, 20, 25]. In [19], features derived from the cepstral analysis of Gammatone filterbank outputs outperform the conventional MFCC and perceptual linear prediction (PLP) features in a speech recognition experiment. In a music and audio genre classification study, features extracted from the temporal envelope of a Gammatone filterbank surpass standard features such as MFCCs [12]. In this paper, we present new features derived from Gammatone filters to describe the music timbre and investigate their applications in music structural segmentation.

Distinct from Western pop music commonly used to evaluate MSS algorithms, Jingju may hold very characteristic music form. Repetitive harmonic structures such as the chorus and verse sections typically found in Western music are hardly present in Jingju. Here we provide some essential background of this genre. The song lyrics are organised in a *couplet* structure which lays the basis of the music structural framework. A couplet contains two *melodic lines* performed by the singer with background accompaniments. Although following certain melodic, rhythmic and instrumentation regularities, each couplet unfolds in a temporal order and is hardly repeated with another. A passage of melodic lines expressing specific music ideas or motifs can be grouped into a *melodic section* which can play a rather integrate role in the overall musical form. Jingju consists mainly of three identifiable musical elements: *mode and modal systems*, *metrical patterns*, and *melodic-phrases*. When composing a Jingju play, modal systems and modes are firstly chosen to set the overall atmosphere. The metrical patterns are then accordingly arranged to portray specific content in each passage of lyrics and signal the sectional. Here a *melodic-phrase* differs from the Western understanding for a *melodic phrase* in the sense that it refers to the melodic progression and the tone for singing a single character from the lyrics [23]. It is considered the smallest meaningful unit in Jingju aesthetics. Jingju songs also have instrumental connectives to bridge the sung parts in the arias. Such connectives can serve as preludes to introduce melodic passages and as interludes to tie together successive couplets. Collectively, these musical elements are hierarchically united into composite organisations and shape the overall temporal music structure. In the next section, we will present the collection of the Jingju dataset.

3. DATASETS

3.1 Existing Corpora

A few MSS datasets are available in the literature. Two are used in this work. The first consists of 174 songs from The Beatles. The annotation was first made at Music Technology Group (MTG), Universisitat Pompeu Fabra (UPF) and corrected at Tampere University of Technology (TUT) [15]. We note this dataset *BeatlesTUT*. The second, SALAMI Internet Archive (S-IA), contains 272 pieces as a publicly available subset of the full SALAMI dataset [22]. The main design consideration of the SALAMI dataset is to cover a wide variety of music genres. S-IA also contains a large set of live recordings hence providing a diversity of audio qualities.

These datasets employ different annotation principles. BeatlesTUT is annotated on a *functional* level, i.e., the music is segmented into structural parts expressing specific musical functions. In contrast, S-IA is annotated incorporating different principles on multiple hierarchies including the *music similarity* level, the *function* level and the highest *lead instrument* level. In this paper, we use the music similarity level annotations for S-IA. The inclusion of these two datasets will provide respectively a standard example of Western pop music and a diversity of styles hence gain us meaningful reference for the structural analysis of Jingju. Additionally, they cover two different annotation principles and can serve as a comprehensive testbed for the investigated segmentation algorithms and audio features.

3.2 Composition of Jingju Dataset

The Jingju corpus presented in this paper consists of 30 excerpts from commercial CDs [2], sampled at 44.1 KHz and 16 bits per sample with a total length of 3.6 hours. The CDs were released in the recent decade with recordings of classical repertoires performed by the most renowned musicians. A full Jingju play can last up to a few hours comprising multiple arias or acts. To fulfil the computational purpose of this study, the 30 excerpts in this dataset are taken from 20 different Jingju plays, with an average length of 432 seconds. The audio samples were chosen on the criteria of repertoire coverage, structural diversity and audio quality. One prerequisite an excerpt can be selected is that various structural parts should be present characterising temporal progressions or changes of sectional units. The selected samples cover the two available modes and various metrical patterns. Half of them are performed by

female singers and half by male singers with different role types. We denote this dataset *CJ* in this paper.

In this work, annotations are made to describe the *musical similarity* setting aside the musical functions of segments, similarly to the lowest level of S-IA. There are mainly two reasons why the similarity level is chosen. First, low-level music similarity is a phenomenon that can be perceived for different genres [3]. Analysing the music structure on the similarity level would therefore grant us fair comparison across genres. However, the instrumentations and the music functions of the sectional units can be highly genre-dependent. Second, the melodic sections are never repeated with each other at a segment-level as the chorus-verse based music forms would do. This could lead to dubious decisions in defining the structural sections based on specific musical functions. Meanwhile, there exists much expressiveness in the performance, which may raise the demand of analysing the ornamentations in parallel to the functional structure units hence introducing uncertainties in locating sectional boundaries.

Three listeners (noted "A1", "A2" and "A3") participated in annotating the music. Another two engaged in verifying their annotations, one is the first author of this paper ("V1"), familiar with this music style as an amateur, the other is a Jingju musician and musicologist ("V2"). All annotators are Chinese and were provided with music scores and lyrics [26]. They were instructed to pay attention to prominent changes in music phenomena such as rhythm, melody, harmony or timbre, and mark the boundaries in places where the similarities break. Within a section, high similarity should present expressing a unified musical idea. When multiple annotators from A1, A2 and A3 have noted a boundary, it is accepted with its final location being the average of those indicated by individual annotators. When a boundary is noted by only one of A1, A2 and A3, its acceptance rests on a conscious discussion of V1 and V2, over whether a boundary should be noted and if yes, its exact position.

The software used for annotation is Sonic Visualiser [1] which displays the annotators the waveform and the spectrogram of the track and allows them to navigate it through, as well as to add time instants and notes to mark a segment boundary. Figure 1 shows respectively the annotations by V1 and V2 and the final accepted annotation for a 60-second excerpt of the recording "Ba wang bie ji" (meaning "Farewell my concubine"), with the lyrics shown on the top. The phrase shown constitutes half a couplet. We can notice that this phrase is sung at a relatively slow tempo with one sung character may last several seconds. This gives the performer lots of freedom for ornamentations in the singing such as vibratos and even intermittence. The final decision is made when agreements have been reached by V1 and V2. Additional annotation information and metadata for this dataset can be found in [24] and online [2].

Here we discuss some properties of the annotations focussing on the *inter-annotator agreement* (IAA) between V1 and V2. In the assessment of each of the two annotations, one is treated as the "ground truth" and the other as the "detection" and then their roles are rotated. We report the average of these two measures. With a variety of existing measures commonly used to compare multiple annotations for music structure [21], the following metrics are used: *precision* (P), *recall* (F) and *F-measure* (F) retrieved at the tolerance of 0.5s ($\pm$0.25s) and 3s ($\pm$1.5s), median of the distance between each annotated segment boundary to its closest detected segment boundary (D_{ad}) and that between each detected segment boundary to its closest annotated segment boundary (D_{da}). Statistics of the investigated metrics are given in Table 1. The IAA measured at 0.5s is relatively comparable to that measured at 3.0s in the corresponding cases. However, choosing different annotation as the ground truth each time yields lower recall than precision and lower precision than recall, hence substantial *false negative* (FN) and *false positive* (FP) respectively. This is mainly because the two annotators noted different numbers of boundaries. This observation shows that the boundary decisions do depend on the annotators' individual understanding of the music. Some statistics of the datasets used in this paper are given in Table 2. We can notice that the average segment lengths of S-IA and CJ are shorter than that of BeatlesTUT mainly due to individual annotation principles.

[1] http://www.sonicvisualiser.org/

[2] http://www.isophonics.net/content/jingju-structural-segmentation-dataset

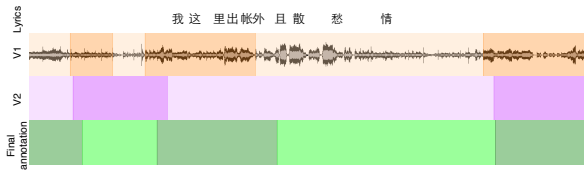


Figure 1: Boundary annotations for a 60-second excerpt of the recording "Ba wang bie ji" from Dataset *CJ*. Panes from top to bottom show respectively the lyrics of the singing (in Chinese), annotations by annotator V1 and V2 and the final annotation.

P 0.5s	R 0.5s	F 0.5s	P 3s	R 3s	F 3s	D_{ad}	D_{da}
0.891	0.675	0.693	0.911	0.689	0.743	0.27	11.88
(0.075)	(0.188)	(0.141)	(0.075)	(0.182)	(0.144)	(0.25)	(48.12)

Table 1: Average agreement between annotator V1 and V2 for recordings in dataset CJ (standard deviations into parenthesis). Statistics include: pairwise precision (P), recall (R) and F-measure (F) measured at 0.5s and 3s, and the median of distances between each annotated segment boundary to its closest detected segment boundary (D_{ad}) and that between each detected segment boundary to its closest annotated segment boundary (D_{da}).

Dataset	No. tracks	Len. track	No. segments	Len. segment
BeatlesTUT	174	159.30 (50.08)	10.21 (2.32)	17.73 (5.45)
S-IA	258	333.09 (130.78)	56.26 (32.07)	7.69 (5.28)
CJ	30	421.38 (219.02)	44.37 (19.18)	9.56 (4.57)

Table 2: Statistics of datasets (standard deviations into parenthesis): number of samples in the dataset, average length of each sample (in second), average number of segments per sample, average length of each segment (in second).

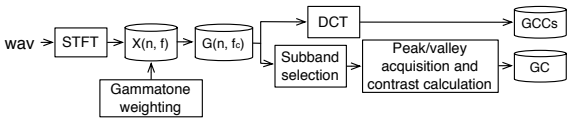


Figure 2: Gammatone feature extraction workflow.

4. GAMMATONE FEATURES

4.1 Gammatone Filters

In the Patterson *Gammatone* model, the cochlea processing is simulated by a Gammatone auditory filterbank with the bandwidth of each filter described by an *Equivalent rectangular bandwidth* (ERB) [14]. The Gammatone function is defined by the impulse response of the signal:

$$GF(t, f_c) = at^{(m-1)}e^{(-2\pi bt)}cos(2\pi f_c t + \phi), \quad (1)$$

where a is the amplitude factor, m is the order of the filter, b is the bandwidth of the filter in Hz which largely determines the duration of the impulse response, f_c in Hz is the centre frequency of the filter and t is the time in s. An efficient implementation is provided by Slaney [20]. It should be noted that $GF(t, f_c)$ keeps the original sample frequency f_s. To derive a spectrogram-like TF representation, noted *Gammatonegram* in this paper, it is necessary to sum up the energy over fixed time windows.

However, to process a signal with a bank of M Gammatone filters can still be computationally expensive. Ellis introduced an alternative method using a *fast Fourier transform* (FFT)-based approximation [5]. In this approach, a conventional fixed-bandwidth spectrogram is first calculated whose frequency bins are then aggregated into Gammatone responses with coarser resolutions via a weighting function. This approximates matches the accurate method by Slaney very closely despite neglecting phase information of each frequency bin when summing them up [5]. Another difference the approximation can introduce is a loss of temporal resolution due to the Fourier transform applied beforehand. However, this is not considered unfavourable in the structural analysis scenario as a relatively coarse temporal resolution is commonly employed aiming at a more musically meaningful scale for the analysis [16]. Many methods propose to use a window size of 0.1 - 1 second, or equal to the beat length [9,16]. In this paper, we use the FFT-based implementation following Ellis [5]. We use M = 64 channels with centre frequencies spaced between 50 Hz to f_s/2 (f_s = 44.1 KHz) on an ERB scale following the default setting of Slaney's and Ellis' toolboxes [5, 20]. The energy of the gammatone filterbank output is aggregated over a 46ms window and shifted by 23ms into the Gammatonegram $G(n, f_c)$.

4.2 Feature Extraction

Two features are extracted from the Gammatonegram capturing different properties of the signal as summarised in Figure 2. Here we describe the feature extraction process. The Discrete cosine transform (DCT) is a commonly used dimensionality reduction technique in feature extraction. It is adopted as the last step in the calculation of the MFCCs which proved highly successful in describing the sound timbre [10]. One motivation of this paper is to find alternative timbral features to describe the music structure incorporating auditory cues. To this end, we introduce a feature called *Gammatone cepstral coefficients* (GCCs) following [19] to describe the average energy distribution of each subband. Specifically, we apply a DCT to $G(n, f_c)$ to de-correlate its components.

$$GCCs(n) = \sum_{i=1}^{M} G(n)cos\left(\frac{\pi}{M}(i + \frac{1}{2})n\right) \quad (2)$$

Shao and his colleagues report that the lowest 30 orders of GCCs contain the majority information of a GF with 128 filterbanks to recover the speech signal [19]. In a sound classification work, the number of filters and GCCs coefficients are set to 48 and 13 with the later identical to MFCCs used in the study [25]. Here we use a 13-coefficient GCCs same as MFCCs to derive a fair comparison of the two. We will discuss the setting of number of coefficients in Section 6. However, a log operation is excluded as applied in common cepstral analyses since initial investigation shows degraded segmentation due to an overemphasis of the lower frequency components when using the logarithmic scale.

Similar to MFCCs, GCCs describe the average energy distribution of each subband in a compact form. Here we are also interested in the extents of flux within the spectra indicating the level of harmonicities associated with different frequency components. To this end, we present a novel feature, *Gammatone contrast* (GC). The extraction of this feature is inspired by the *spectral contrast* (SC) feature which is based on the *octave-scale* filters and is very popular in music genre classification studies [8].

The calculation of the GC feature is as follows. As the first step, the Gammatone filterbank indices $[1, ..., M]$ are grouped into C subbands with linearly equal subdivisions. Since the spectrum is originally laid out on a non-linear ERB scale, the frequency non-linearity is still reserved in the subbands. We use C=6 similar to [8] in this study yielding a subband frequency division of $[50, 363.198, 1028.195, 2440.148, 5438.074, 11803.409, 22050]$. We note $\mathbf{V}$ the Gammatonegram vector of the zth subband $[G_{z,0}(n), G_{z,1}(n), \ldots, G_{z,K-1}(n)]^T$ where $z \in [0, C-1]$, $\mathbf{V}' = \left[G'_{z,0}, G'_{z,1}, \ldots, G'_{z,K-1}\right]^T$ is $\mathbf{V}$ sorted in an ascending order such that $G'_{z,0}(n) < G'_{z,1}(n) < \ldots < G'_{z,K-1}(n)$. We calculate the difference of the strength of peaks and valleys for each subband to derive the C-dimensional GC feature:

$$GC_z(n) = log\left(G'_{z,K-1}(n) - G'_{z,0}(n)\right), \quad (3)$$

GCCs and the vector-wise concatenation of GCCs and GC will be evaluated in comparison with two commonly used features MFCCs and chromagram in the MSS scenario as introduced shortly. The reason why GC is not

evaluated individually is that it measures only the relative contrasts within subband energies hence may lack complementary spectral information. We use the LibROSA music and audio analysis library which provides feature extraction modules for MFCCs and chromagram [11]. We implement the Gammatone module into this library to obtain a uniform feature extraction environment. The extracted GCCs, GC, MFCCs and chromagram are respectively 13-, 6-, 13- and 12-dimensional features. The window and step size for feature extraction are respectively 46ms and 23ms.

5. SEGMENTATION EXPERIMENT

The investigated Gammatone features are evaluated on the presented datasets in a segmentation context using *Music Structure Analysis Framework* (MSAF) which relies on LibROSA [11] for feature extraction and includes a list of recently published segmentation algorithms [13]. Three are used in this paper covering the novelty-, homogeneity- and repetition-based segmentation principles [16]. The first one is included into MSAF by the author of this paper and the rest two are provided by MSAF.

The first method is a novelty-based one presented in a recent work [24] following Foote [6]. A Self-similarity matrix (SSM) is constructed by calculating the pairwise Euclidean distance between vectors of the feature matrix. A Gaussian-tapered "checkerboard" kernel is correlated along the main diagonal of the SSM yielding a novelty curve. Given a list of local maxima detected by the adaptive thresholding from the smoothed novelty curve, a second-degree polynomial $y = ax^2 + bx + c$ is fitted on the novelty curve centred around each local maximum with a window of five samples. In this quadratic model a and c control respectively the sharpness and amplitude of each peak. Assessing these two parameters hence allows us to assess the sharpness and the magnitude of a peak independently where it will only be selected as a segment boundary when both meet set conditions. This method is denoted *Quadratic novelty* (QN) in this paper. The second is a homogeneity-based approach which attempts to segment the music by clustering the frames into different section types [9]. First, audio frames are labelled into hidden Markov model (HMM) states derived from trained features. Then histograms of neighbouring frames are clustered into segment types where the temporal continuity on cluster assignments is obtained from the HMMs. Segment boundaries are retrieved by locating changing of segment types. We note this algorithm *Constrained clustering* (CC). The third method, SF, uses features called *structure features* which incorporate both local and global properties accounting for structural information in the recent past [17]. To construct the structure features, a multi-dimensional time series is firstly obtained by accumulating vectors of a standard audio feature ranging across a time span. A *recurrence plot* (RP) is then computed containing the pairwise resemblance $P_{i,j}$ between time series centred at different time locations i and j. Here, an *RP* differs from an SSM typically used to describe music structure in the sense that $P_{i,j}$ is calculated between feature vectors embedded with time-shifts, i.e., between multi-dimensional time series instead of static vectors. This recurrence nature enables encapsulating both homogeneity and repetition properties in the feature space. The structure features are obtained by estimating Gaussian probability density of the time lag matrix of the RP. Finally, a novelty curve is computed where segment boundaries are detected following the standard novelty approach [6]. In this way, all three basic principles – novelty, homogeneity, and repetition, are combined in the segmentation process.

While QN is newly included into MSAF along with the research presented in this paper, CC and SF are provided by the original MSAF system, with CC forked from its open source software by Levy [9] and SF reimplemented from [17] by Nieto [13].

6. RESULTS AND DISCUSSION

The segmentation boundary retrieval results are evaluated with the *precision* (P), *recall* (R) and *F-measure* (F) measured at 3s [9] using the MSAF framework [13]. Results are shown in Table 3 obtained with system configurations parameterised both globally and on each dataset individually. By doing this, we are aiming to investigate how dependent each algorithm is on parameter configurations in the context of a specific dataset. To avoid a potential overfitting, the discussions made in the remainder of this paper are based on the results obtained with the globally uniform configurations, unless noted otherwise.

Here we analyse presented Gammatone features as structural descriptors. We first compare GCCs to MFCCs, both are based on cepstral analysis of the spectra and related to the music timbre. When assessed on individual datasets, GCCs outperform MFCCs on CJ using all investigated segmentation algorithms, with statistical significance observed when using CC ($p = 0.035$) and QN ($p = 0.019$), while the two strike a tie on BeatlesTUT and S-IA.

Figure 3 shows the SSMs derived from the MFCCs and GCCs on an excerpt of the Jingju song "Ba wang bie ji" from CJ (only the first 60 seconds are shown for visualisation purposes). It can be noticed that GCCs yield more distinguished sectional variations than MFCCs. Singing-based musical works such as Jingju or Western opera may present less notable repetitive harmonic or rhythmic patterns than the popular music. However, the vocal-driven nature makes the singing voice an important discriminator of the music structure with its salient presence in the overall instrumentation. In the case of Jingju, new structural units can emerge in the same melodic passage with subtle timbral variations, as shown in Figure 3(a). The dynamics introduced by the singing voice are mainly present in the lower frequency part of the spectrum, which can be better captured by using the ERB scale than Mel. Meanwhile, emphasising the lower sound levels, the ERB warping can be robust against high-frequency transients which may interfere with the analysis. When the music presents more distinguishable timbral variations, GCCs summarise the structure equally effectively as MFCCs, as indicated by their comparable performances on BeatlesTUT and S-

	GCC						GCC+GC						MFCC						Chromagram					
	Individual config			Global config			Individual config			Global config			Individual config			Global config			Individual config			Global config		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
BeatlesTUT	0.601	0.647	0.614	0.557	0.706	0.612	0.600	0.650	0.615	0.568	0.706	0.618	0.599	0.634	0.606	0.568	0.706	0.618	0.588	0.636	0.600	0.527	0.668	0.579
CJ	0.684	0.486	0.550	0.732	0.448	0.538	0.701	0.465	0.552	0.688	0.436	0.522	0.701	0.483	0.555	0.689	0.441	0.525	0.651	0.509	0.540	0.645	0.447	0.514
S-IA	0.550	0.525	0.524	0.500	0.562	0.515	0.555	0.536	0.535	0.514	0.586	0.533	0.572	0.559	**0.551**	0.517	0.565	0.526	0.558	0.544	0.535	0.514	0.596	0.535

(a) CC

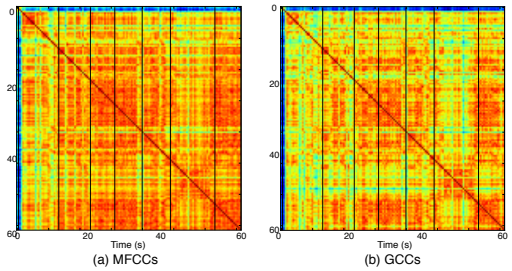
BeatlesTUT	0.564	0.643	0.588	0.523	0.687	0.580	0.565	0.667	0.598	0.523	0.710	0.587	0.638	0.589	0.596	0.584	0.635	0.580	0.468	0.691	0.544	0.435	0..726	0.530
CJ	0.619	0.715	0.639	0.685	0.475	0.543	0.599	0.761	**0.654**	0.673	0.521	**0.574**	0.588	0.715	0.625	0.706	0.439	0.521	0.520	0.798	0.616	0.557	0.593	0.562
S-IA	0.463	0.599	0.500	0.430	0.639	0.492	0.471	0.623	0.516	0.438	0.666	0.508	0.526	0.572	0.525	0.478	0.610	0.513	0.413	0.663	0.486	0.394	0.704	0.480

(b) QN

BeatlesTUT	0.625	0.739	0.667	0.594	0.755	0.654	0.630	0.743	0.673	0.603	0.761	0.663	0.644	0.743	0.678	0.621	0.772	0.671	0.644	0.751	**0.683**	0.612	0.777	**0.679**
CJ	0.559	0.799	0.631	0.688	0.461	0.534	0.536	0.807	0.628	0.664	0.471	0.540	0.554	0.792	0.627	0.677	0.482	0.530	0.542	0.759	0.617	0.668	0.439	0.514
S-IA	0.497	0.577	0.515	0.442	0.649	**0.545**	0.502	0.586	0.520	0.433	0.635	0.493	0.504	0.588	0.523	0.443	0.635	0.497	0.494	0.588	0.524	0.438	0.630	0.500

(c) SF

Table 3: Segmentation results using investigated features on the *BeatlesTUT*, *CJ* and *S-IA* datasets with algorithm *CC*, *QN* and *SF*. Highest F-measure obtained for each dataset is shown in bold.



(a) MFCCs (b) GCCs

Figure 3: SSMs computed using MFCCs and GCCs on the first 60 seconds excerpt of "Ba wang bie ji" from CJ. Vertical lines indicate segment boundaries.

IA shown in Table 3. This hence suggests the GCCs as a competitive alternative to the commonly used features for music structural description.

Combining GC with GCCs by matrices concatenation has introduced improvements over using GCCs alone for most cases on the investigated datasets with each algorithm tested. However, a statistical significance is not always present as suggested by Student's t-test with related samples when comparing $GCCs + GC$ to $GCCs$. The main effect of using the additional GC feature is a more pronounced within-SSM variance. This has led to the retrieval of more boundaries as indicated by a higher recall in the general case yet an occasional degrading precision.

Investigated features and algorithms perform differently on Western and Jingju music. Chromagram feature and MFCCs work reliably on BeatlesTUT and S-IA. For Jingju, timbre features capture its structural characteristics better than the chroma feature, with auditory inspired Gammatone features outperforming MFCCs. When using the same features and algorithms, similar segmentation results in terms of F-measures tend to emerge from CJ and S-IA, both use the annotation at the music similarity level. However, it can be noticed that algorithms are more dependent on parameter configurations when evaluated on CJ than on S-IA and BeatlesTUT, reflected by the substantial degradation of the F-measures observed when changing the parameter configuration tuned for the individual dataset to the global setting. This suggests the need of designing new segmentation methods to bridge the gaps between genres. It also implies that contextual knowledge, such as the genre and the level of music structure to analyse, can assist a segmentation system to obtain better performance.

It is also noted that SF appears less effective than QN on the CJ dataset when using the chromagram feature, as shown in Table 3. This is in contrast with the many observations for Western pop music, where repetition-based segmentation algorithms are identified as useful interpreters of structural characteristics reflected by chroma features. We find that for Jingju, the chromagram feature forms mainly *block* structures as do the MFCCs instead of *stripes* in the sub-diagonals of the SSMs. This somehow contradicts with many established observations for Western pop music. As introduced in Section 2, the repetitive chord structure is lacking in Jingju in the sense of chorus and verse. The chroma feature in the Jingju scenario functions mainly to capture its low-level homogeneity in the vicinity. Therefore, the same audio feature may exhibit different structural characteristics for specific music genres and the selection of segmentation algorithms should be adapted accordingly to interpret such patterns.

7. CONCLUSION

This paper investigated novel features derived from Gammatone filters for music structural segmentation beyond the commonly studied music corpora. A new dataset with Chinese traditional Jingju music is presented to complement the existing evaluation corpora. In the music structural segmentation experiment, GCCs surpass MFCCs notably on the Jingju dataset comprising vocal-driven music. The fact that the Gammatone features also obtain comparable segmentation results to MFCCs and chromagram on the Beatles and S-IA datasets indicate them to be competitive alternatives to existing audio features for music structural analysis. Different patterns emerge for different music genres from existing algorithms and audio features, shedding new perspectives on music structural segmentation research.

8. REFERENCES

[1] R. Caro Repetto, A. Srinivasamurthy, S. Gulati, and X. Serra. Jingju music: concepts and computational tools for its analysis. Technical report, Tutorial session,

International Society for Music Information Retrieval Conference (ISMIR), 2014.

[2] China Music Group (CMG). Peking opera box set, limited edition. Audio CD, http://chinamusicgroup.com/get_music.php, 2010.

[3] D. Deutsch, editor. *The psychology of music*. Academic Press, 3rd edition, 2012.

[4] J. S. Downie. Toward the scientific evaluation of music information retrieval systems. In *Proceedings of the 4th International Society for Music Information Retrieval Conference (ISMIR)*, 2003.

[5] D. P. W. Ellis. Gammatone-like spectrograms. http://www.ee.columbia.edu/ dpwe/resources/matlab/, 2009.

[6] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, 2000.

[7] T. Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proceedings of the International Computer Music Conference (ICMC)*, 1999.

[8] D. Jiang, L. Lu, H. Zhang, J. Tao, and L. Cai. Music type classification by spectral contrast feature. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2002.

[9] M. Levy and M. B. Sandler. Structural segmentation of musical audio by constrained clustering. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):318–326, 2008.

[10] B. Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the 1st International Society for Music Information Retrieval Conference (ISMIR)*, 2000.

[11] B. McFee, M. McVicar, C. Raffel, D. Liang, O. Nieto, E. Battenberg, J. Moore, D. Ellis, R. Yamamoto, R. Bittner, D. Repetto, P. Viktorin, J. F. Santos, and A. Holovaty. Librosa: Python library for audio and music analysis. In *Proceedings of the 14th Python in Science Conference*, 2015.

[12] M. F. McKinney and J. Breebaart. Features for audio and music classification. In *Proceedings of the 4th International Society for Music Information Retrieval Conference (ISMIR)*, 2003.

[13] O. Nieto and J. P. Bello. Msaf: Music structure analytis framework. In *Late breaking session, 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[14] R. D. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice. An efficient auditory filterbank based on the gammatone function. Applied psychology unit (apu), report 2341, Cambrige, UK, 1987.

[15] J. Paulus and A. Klapuri. Labelling the structural parts of a music piece with markov models. In *Proceedings of Computers in Music Modeling and Retrieval Conference (CMMR)*, 2008.

[16] J. Paulus, M. Müller, and A. Klapuri. State of the art report: audio-based music structure analysis. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, 2010.

[17] J. Serrà, M. Müller, P. Grosche, and J. L. Arcos. Unsupervised detection of music boundaries by time series structure features. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, 2012.

[18] X. Serra. Creating research corpora creating research corpora for the computational study of music: the case of the compmusic project. In *Proceedings of International Audio Engineering Society (AES) Conference*, 2014.

[19] Y. Shao, Z. Jin, D. Wang, and S. Srinivasan. An auditory-based feature for robust speech recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009.

[20] M. Slaney. An efficient implementation of the patterson-holdsworth auditory filter bank. Technical report, Apple Technical Report, 1993.

[21] J. B. L. Smith. A comparison and evaluation of approaches to the automatic formal analysis of musical audio. Master's thesis, McGill University, 2010.

[22] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. De Roure, and J. S. Downie. Design and creation of a large-scale database of structural annotations. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, 2011.

[23] J. P. J Stock. A reassessment of the relationship between text, speech tone, melody, and aria structure in beijing opera. *Journal of Musicological Research*, 18(3):183–206, 1999.

[24] M. Tian and M. B. Sandler. Towards music structural segmentation across genres: features, structural hypotheses and annotation principles. *Special Issue on Intelligent Music Systems and Applications, Intelligent Systems and Technology, ACM Transactions on (ACM TIST)*, In press.

[25] X. Valero and F. Alías. Gammatone cepstral coefficients: biologically inspired features for non-speech audio classification. *Multimedia, IEEE Transactions on*, 14(6):1684–1689, 2012.

[26] Shanghai wenyi chubanshe. *Collection of jingju scores ("Jingju qupu jicheng")*. 1992.

[27] E. Wichmann. *Listening to theatre: the aural dimension of Beijing opera*. University of Hawaii Press, 1991.

Poster Session 3

A COMPARISON OF MELODY EXTRACTION METHODS BASED ON SOURCE-FILTER MODELLING

Juan J. Bosch[1] **Rachel M. Bittner**[2] **Justin Salamon**[2] **Emilia Gómez**[1]
[1] Music Technology Group, Universitat Pompeu Fabra, Spain
[2] Music and Audio Research Laboratory, New York University, USA
{juan.bosch,emilia.gomez}@upf.edu, {rachel.bittner,justin.salamon}@nyu.edu

ABSTRACT

This work explores the use of source-filter models for pitch salience estimation and their combination with different pitch tracking and voicing estimation methods for automatic melody extraction. Source-filter models are used to create a mid-level representation of pitch that implicitly incorporates timbre information. The spectrogram of a musical audio signal is modelled as the sum of the leading voice (produced by human voice or pitched musical instruments) and accompaniment. The leading voice is then modelled with a Smoothed Instantaneous Mixture Model (SIMM) based on a source-filter model. The main advantage of such a pitch salience function is that it enhances the leading voice even without explicitly separating it from the rest of the signal. We show that this is beneficial for melody extraction, increasing pitch estimation accuracy and reducing octave errors in comparison with simpler pitch salience functions. The adequate combination with voicing detection techniques based on pitch contour characterisation leads to significant improvements over state-of-the-art methods, for both vocal and instrumental music.

1. INTRODUCTION

Melody is regarded as one of the most relevant aspects of music, and melody extraction is an important task in Music Information Retrieval (MIR). Salamon et al. [21] define melody extraction as the estimation of the sequence of fundamental frequency (f_0) values representing the pitch of the lead voice or instrument, and this definition is the one employed by the Music Information Retrieval Evaluation eXchange (MIREX) [7]. While this definition provides an objective and clear task for researches and engineers, it is also very specific to certain types of music data. Recently proposed datasets consider broader definitions of melody, which are not restricted to a single instrument [2, 4, 6].

Composers and performers use several cues to make melodies perceptually salient, including loudness, timbre, frequency variation or note onset rate. Melody extraction methods commonly use cues such as pitch continuity and pitch salience, and some of them group pitches into higher level objects (such as tones or contours), using principles from Auditory Scene Analysis [8, 13, 16, 18, 20]. Some approaches have also considered timbre, either within a source separation framework [10, 17], with a machine learning approach [11], or in a salience based approach [14, 16].

One of the best performing methods so far in MIREX in terms of overall accuracy [20] (evaluated in 2011) is based on the creation and characterisation of pitch contours. This method uses a fairly simple salience function based on harmonic summation [15] and then creates and characterises pitch contours for melody tracking. Voicing detection (determining if a frame contains a melody pitch or not) is one of the strong aspects of this method, even though it might be improved further by incorporating timbre information. In contrast, alternative approaches employ more sophisticated salience functions, but the pitch tracking and voicing estimation components are less complex [10, 12]. Voicing detection has in fact been identified in the literature as a crucial task for improving melody extraction systems [10, 20].

While these approaches work especially well for vocal music, their performance decreases for instrumental pieces, as shown in [6] and [2], where a drop of 19 percentage points in overall accuracy was observed for instrumental pieces compared to vocal pieces. A main challenge for melody extraction methods is thus to cope with more complex and varied music material, with melodies played by different instruments, or with harmonised melodic lines [21]. A key step towards the development of more advanced algorithms and a more realistic evaluation is the availability of large and open annotated datasets. In [4, 6] the authors presented a dataset for melody extraction in orchestral music with such characteristics, and MedleyDB [2] also includes a variety of instrumentation and genres. Results on both datasets generally drop significantly in comparison to results on datasets used in MIREX [7].

Based on results obtained in previous work [5, 6], we hypothesise that a key ingredient for improving salience-based melody extraction in relatively complex music data is the salience function itself. In particular, we propose combining strong components of recently proposed algorithms: (1) a semantically rich salience function based on a

© First author, Second author, Third author, Fourth author, Fifth author, Sixth author. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** First author, Second author, Third author, Fourth author, Fifth author, Sixth author. "A comparison of melody extraction methods based on source-filter modelling", 17th International Society for Music Information Retrieval Conference, 2016.

source-filter model, which proved to work especially well in pitch estimation [6, 9, 10], and (2) pitch-contour-based tracking [2, 20], which presents numerous benefits including high-performance voicing detection.

2. RELATED METHODS

This section describes the pitch salience functions and melody tracking methods used as building blocks for the proposed combinations.

2.1 Salience functions

Most melody extraction methods are based on the estimation of pitch salience - we focus here on the ones proposed by Salamon and Gómez [20], and Durrieu et al. [9].

Durrieu et al. [9] propose a salience function within a separation-based approach using a Smoothed Instantaneous Mixture Model (SIMM). They model the spectrum X of the signal as the lead instrument plus accompaniment $\hat{X} = \hat{X}_v + \hat{X}_m$. The lead instrument is modelled as: $\hat{X}_v = X_\Phi \circ X_{f_0}$, where X_{f_0} corresponds to the source, X_Φ to the filter, and the symbol $\circ$ denotes the Hadamard product. Both source and filter are decomposed into basis and gains matrices as $X_{f_0} = W_{f_0} H_{f_0}$ and $X_\Phi = W_\Gamma H_\Gamma H_\Phi$ respectively. H_{f_0} corresponds to the pitch activations of the source, and can also be understood as a representation of pitch salience [9]. The accompaniment is modelled as a standard non negative matrix factorization (NMF): $\hat{X}_m = \hat{W}_m \hat{H}_m$. Parameter estimation is based on Maximum-Likelihood, with a multiplicative gradient method [10], updating parameters in the following order for each iteration: H_{f_0}, H_Φ, H_m, W_Φ and W_m. Even though this model was designed for singing voice, it can be successfully used for music instruments, since the filter part is related to the timbre of the sound, and the source part represents a harmonic signal driven by the fundamental frequency.

Salamon and Gómez [20] proposed a salience function based on harmonic summation: a time-domain Equal-Loudness Filter (ELF) is applied to the signal, followed by the Short-Time Fourier Transform (STFT). Next, sinusoidal peaks are detected and their frequency/amplitude values are refined using an estimate of the peaks' instantaneous frequency. The salience function is obtained by mapping each peak's energy to all harmonically related f_0 candidates with exponentially decaying weights.

2.2 Tracking

The estimated pitch salience is then used to perform pitch tracking, commonly relying on the predominance of melody pitches in terms of loudness, and on the melody contour smoothness [1, 10, 12, 20].

Some methods have used pitch contour characteristics for melody tracking [1, 20, 22]. Salamon and Gómez [20] create pitch contours from the salience function by grouping sequences of salience peaks which are continuous in time and pitch. Several parameters need to be set in this process, which determine the amount of extracted contours. Created contours are then characterised by a set of features: pitch (mean and deviation), salience (mean, standard deviation), total salience, length and vibrato related features.

The last step deals with the selection of melody contours. Salamon [20] first proposed a pitch contour selection (PCS) stage using a set of heuristic rules based on the contour features. Salamon [22] and Bittner [1] later proposed a pitch contour classification (PCC) method based on contour features. The former uses a generative model based on multi-variate Gaussians to distinguish melody from non-melody contours, and the latter uses a discriminative classifier (a binary random forest) to perform melody contour selection. The latter also adds Viterbi decoding over the predicted melodic-contour probabilities for the final melody selection. However, these classification-based approaches did not outperform the rule-based approach on MedleyDB. One of the important conclusions of both papers was that the sub-optimal performance of the contour creation stage (which was the same in both approaches) was a significant limiting factor in their performance.

Durrieu et al. [10] similarly use an HMM in which each state corresponds to one of the bins of the salience function, and the probability of each state corresponds to the estimated salience of the source (H_{f_0}).

2.3 Voicing estimation

Melody extraction algorithms have to classify frames as voiced or unvoiced (containing a melody pitch or not, respectively). Most approaches use static or dynamic thresholds [8, 10, 12], while Salamon and Gómez exploit pitch contour salience distributions [20]. Bittner et al. [1] determine voicing by setting a threshold on the contour probabilities produced by the discriminative model. The threshold is selected by maximizing the F-measure of the predicted contour labels over a training set.

Durrieu et al. [10] estimate the energy of the melody signal frame by frame. Frames whose energy falls below the threshold are set as unvoiced and vice versa. The threshold is empirically chosen, such that voiced frames represent more than 99.95% of the leading instrument energy.

3. PROPOSED METHODS

We propose and compare three melody extraction methods which combine different pitch tracking and voicing estimation techniques with pitch salience computation based on source-filter modelling and harmonic summation. These approaches have been implemented in python and are available online [1]. We reuse parts of code from Durrieu's method [2], Bittner et al. [3], and Essentia [4] [3], an open source library for audio analysis, which includes an implementation of [20] which has relatively small deviations

[1] https://github.com/juanjobosch/SourceFilterContoursMelody
[2] https://github.com/wslihgt/separateLeadStereo
[3] https://github.com/rabitt/contour_classification
[4] https://github.com/MTG/essentia

in performance from the authors' original implementation MELODIA [5] . We refer to the original implementation of MELODIA as SAL, and to the implementation in the Essentia library as ESS.

3.1 Pitch Salience Adaptation

There are important differences between the characteristics of salience functions obtained with SIMM (H_{f_0}) and harmonic summation (HS). For instance, H_{f_0} is considerably more sparse, and the range of salience values is much larger than in HS since the NMF-based method does not prevent values (weights) from being very high or very low. This is illustrated in Figure 1: (a) shows the pitch salience function obtained with the source filter model, H_{f_0}. Given the large dynamic range of H_{f_0} we display its energy on a logarithmic scale, whereas plots (b)–(d) use a linear scale. (b) corresponds to HS which is denser and results in complex patterns even for monophonic signals. Some benefits of this salience function with respect to H_{f_0} (SIMM) is that it is smoother, and the range of possible values is much smaller.

Given the characteristics of H_{f_0}, it is necessary to reduce the dynamic range of its salience values in order to use it as input to the pitch contour tracking framework, which is tuned for the characteristics of HS. To do so, we propose the combination of both salience functions $HS(k,i)$ and $H_{f_0}(k,i)$, where k indicates the frequency bin $k = 1 \dots K$ and i the frame index $i = 1 \dots I$. The combination process is illustrated in Figure 1: (1) **Global normalization** (Gn) of HS, dividing all elements by their maximum value $max_{k,i}(HS(k,i))$. (2) **Frame-wise normalization** (Fn) of H_{f_0}. For each frame *i*, divide $H_{f_0}(k,i)$ by $max_k(H_{f_0}(k,i))$. (3) **Convolution in the frequency axis** k of H_{f_0} with a Gaussian filter to smooth estimated activations. The filter has a standard deviation of .2 semitones. (4) **Global normalization** (Gn), whose output is $\widetilde{H_{f_0}}$ (see Figure 1 (c)). (5) **Combination** by means of an element-wise product: $S_c = \widetilde{H_{f_0}} \circ HS$ (see Figure 1 (d)).

3.2 Combinations

We propose three different combination methods. The first (C1) combines the output of two algorithms: estimated pitches from DUR and voicing estimation from SAL. The second (C2) is based on S_c, which combines harmonic summation HS computed with ESS with $\widetilde{H_{f_0}}$, and employs pitch contour creation and selection as the tracking method. The last method (C3) combines S_c with pitch contour creation from [20] and the contour classification strategy from [1]. C2 and C3 correspond to Figure 1, where the contour filtering stage is based on pitch contour selection or pitch contour classification, respectively.

4. EVALUATION

Evaluation was carried out using the MedleyDB and Orchset datasets, following the standard MIREX evaluation

[5] http://mtg.upf.edu/technologies/melodia

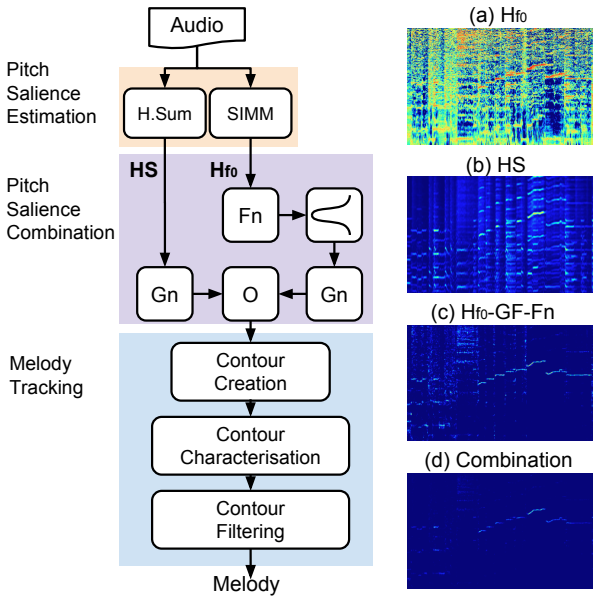


Figure 1. **Left**: Schema for C2 and C3. H.Sum: Harmonic Summation (outputs HS); SIMM: Smoothed Instantaneous Mixture Model (outputs H_{f_0}); Fn: Frame-wise normalisation; Gn: Global normalisation; o: Hadamard product; Gaussian symbol: Gaussian filtering. **Right**: Time-frequency pitch salience representation of an excerpt from "MusicDelta_FunkJazz.wav" (MedleyDB) with (a) SIMM: $log_{10}(H_{f_0})$ is represented for visualisation purposes) (b) Harmonic Summation: HS (c) H_{f_0} normalised per frame, Gaussian filtered and globally normalized ($\widetilde{H_{f_0}}$) (d) Combination (S_c).

methodology. We evaluate the proposed methods (C1–C3) and the original algorithms by Durrieu (DUR), Bittner (BIT) and Salamon. Table 1 provides and overview of their main building blocks. In the case of Salamon's approach, we include the original implementation MELODIA (SAL), and the implementation in the Essentia library (ESS). The latter can be viewed as a baseline for the proposed combination methods (C2, C3), since all three share the same contour creation implementation.

For the evaluation of classification-based methods, we followed [1], and created train/test splits using an "artist-conditional" random partition on MedleyDB. For Orchset we created a "movement-conditional" random partition, meaning all excerpts from the same movement must be used in the same subset: either for training or for testing. Datasets are split randomly into a training, validation and test sets with roughly 63%, 12%, and 25% of the songs/excerpts in the dataset, respectively. This partitioning was chosen so as to have a training set that is as large as possible while retaining enough data in the validation and test sets for results to be meaningful. In order to account for the variance of the results, we repeat each experiment with four different randomized splits.

We set the same frequency limit for all algorithms: $f_{min} = 55$ Hz and $f_{max} = 1760$ Hz. The number of

	(Pre Proc.)+Transform	Salience/Multi f_0 Estim.	Tracking	Voicing
DUR [10]	STFT	SIMM	Vit(S)	Energy thd.
SAL [20]	(ELF)+STFT+IF	H.Sum.	PCS	Salience-based
BIT [1]	(ELF)+STFT+IF	H.Sum.	PCC+Vit(C)	Probability-based
C1	(ELF)+STFT+IF	H.Sum + SIMM	PCS+Vit(S)	Salience-based
C2	(ELF)+STFT+IF	H.Sum + SIMM	PCS	Salience-based
C3	(ELF)+STFT+IF	H.Sum + SIMM	PCC+Vit(C)	Probability-based

Table 1. Overview of the methods. STFT: Short Time Fourier Transform, IF: Instantaneous Frequency estimation, ELF: Equal-Loudness Filters, SIMM: Smoothed Instantaneous Mixture Model, using a Source-Filter model, H.Sum: Harmonic Summation, HMM: Hidden Markov Model, Vit(S): Viterbi on salience function, Vit(C): Viterbi on contours, PCS: Pitch Contour Selection, PCC: Pitch Contour Classification.

bins per semitone was set to 10, and the hop size was 256 samples (5.8 ms), except for SAL which is fixed to 128 samples (2.9 ms) given a sampling rate of 44100 Hz.

4.1 Datasets

The evaluation is conducted on two different datasets: MedleyDB and Orchset, converted to mono using (left+right)/2. MedleyDB contains 108 melody annotated files (most between 3 and 5 minutes long), with a variety of instrumentation and genres. We consider two different definitions of melody, **MEL1**: the f_0 curve of the predominant melodic line drawn from a single source (MIREX definition), and **MEL2**: the f_0 curve of the predominant melodic line drawn from multiple sources. We did not use the third type of melody annotation included in the dataset, since it requires algorithms to estimate more than one melody line (i.e. multiple concurrent lines). Orchset contains 64 excerpts from symphonies, ballet suites and other musical forms interpreted by symphonic orchestras. The definition of melody in this dataset is not restricted to a single instrument, with all (four) annotators agreeing on the melody notes [4, 6]. The focus is pitch estimation, while voicing detection is less important: the proportion of voiced and unvoiced frames is 93.7/6.3%.

Following MIREX methodology [6], the output of each algorithm is compared against a ground truth sequence of melody pitches. Five standard melody extraction metrics are computed using `mir_eval` [19]: Voicing Recall Rate (VR), Voicing False Alarm Rate (VFA), Raw Pitch Accuracy (RPA), Raw Chroma Accuracy (RCA) and Overall Accuracy (OA). See [21] for a definition of each metric.

4.2 Contour creation results

Before evaluating complete melody extraction systems, we examine the initial step, by computing the recall of the pitch contour extraction stage as performed in [1]. We measure the amount of the reference melody that is covered by the extracted contours, by selecting the best possible f_0 curve from them. For the MEL1 definition in MedleyDB the oracle output yielded an average RPA of .66 ($\sigma = .22$) for HS and .64 ($\sigma = .20$) for S_c. In the case of MEL2: .64 ($\sigma = .20$) for HS and .62 ($\sigma = .18$) for S_c. For Orchset we obtain .45 ($\sigma = .21$) for HS and .58 ($\sigma = .18$) for S_c. These results represent the highest raw pitch accuracy that could be obtained by any of the melody extraction methods using contours created from HS and S_c. Note however that these values are dependent on the parametrization of the contour creation stage, as described in [20].

4.3 Melody extraction results

Results for all evaluated algorithms and proposed combinations are presented in Table 2 for MedleyDB (MEL1 and MEL2) and in Table 3 for Orchset. The first remark is that the three proposed combination methods yield a statistically significantly (t-test, significance level $\alpha = .01$) higher overall accuracy (OA) than the baseline (ESS) for both datasets and both melody definitions. The OA of C2 and C3 is also significantly higher than the OA of all other evaluated approaches on MedleyDB (MEL1), with the exception of SAL* (SAL with a voicing threshold optimized for MedleyDB/MEL1): C2-SAL* (p = .10), C3-SAL* (p = .27). For the MEL2 definition C2 and C3 yield an OA that is significantly higher than all compared approaches. In the case of Orchset, C3 is significantly better than C1 and C2 except when increasing the voicing threshold on C2* (p = .78), and outperforms all compared approaches but DUR. As expected, pitch related metrics (RPA, RCA) are the same for C1 and DUR (they output the same pitches), and voicing detection metrics (VR, VFA) are the same for C1 and SAL. This simple combination is already able to significantly improve overall accuracy results on MedleyDB in comparison to all evaluated state-of-the-art approaches except SAL, thanks to the highest pitch estimation accuracy obtained by DUR, and the lowest VFA yielded by SAL. However, OA results are not as high as with DUR on Orchset, due to the lower recall of SAL. An important remark is that DUR always obtains almost perfect recall, since this method outputs almost all frames as voiced. This has a huge influence on the overall accuracy on Orchset, since this dataset contains 93.7% of voiced frames. However, the false alarm rate is also very high, which lowers OA results on MedleyDB, since it contains full songs with large unvoiced portions.

SAL and BIT perform similarly on MedleyDB, but the usefulness of Bittner's method becomes evident on Orch-

[6] http://www.music-ir.org/mirex/wiki/2014:Audio_Melody_Extraction

		MedleyDB-MEL1					MedleyDB-MEL2				
Method	ν	VR	VFA	RPA	RCA	OA	VR	VFA	RPA	RCA	OA
DUR	-	1.0 (.01)	.96 (.05)	.66 (.21)	.73 (.16)	.36 (.16)	1.0 (.01)	.95 (.06)	.65 (.18)	.73 (.14)	.42 (.14)
SAL	.2	.78 (.13)	.38 (.14)	.54 (.27)	.68 (.19)	.54 (.17)	.76 (.12)	.33 (.12)	.52 (.24)	.66 (.17)	.53 (.17)
SAL*	-1	.57 (.21)	.20 (.12)	.52 (.26)	.68 (.19)	.57 (.18)	.54 (.19)	.17 (.09)	.49 (.23)	.66 (.17)	.53 (.18)
BIT	-	.80 (.13)	.48 (.13)	.51 (.23)	.61 (.19)	.50(.15)	.79 (.10)	.44 (.13)	.50 (.20)	.60 (.16)	.50 (.14)
ESS	.2	.79 (.13)	.44 (.15)	.55 (.26)	.68 (.19)	.50 (.17)	.77 (.12)	.39 (.14)	.53 (.23)	.66 (.17)	.50 (.17)
C1	.2	.78 (.13)	.38 (.14)	.66 (.21)	.73 (.16)	.56 (.14)	.76 (.12)	.33 (.12)	.65 (.18)	.73 (.14)	.57 (.13)
C2	.2	.65 (.15)	.26 (.11)	.63 (.21)	.70 (.16)	.61 (.15)	.62 (.14)	.21 (.08)	.61 (.19)	.69 (.14)	.60 (.15)
C3	-	.75 (.15)	.38 (.16)	.58 (.23)	.64 (.19)	.59 (.16)	.74 (.13)	.34 (.13)	.58 (.19)	.64 (.17)	.60 (.14)

Table 2. Mean results (and standard deviation) over all excerpts for the five considered metrics, on MedleyDB with MEL1 and MEL2 definition. Parameter ν refers to the voicing threshold used in the methods based on pitch-contour selection [20]. In the case of classification-based methods (BIT and C3), this parameter is learnt from data. SAL* refers to the results obtained with the best ν for MedleyDB/MEL1.

set: with the same candidate contours, the RPA increases with respect to SAL. This classification-based method is thus partially able to learn the characteristics of melody contours in orchestral music. Orchset is characterized by a higher melodic pitch range compared to most melody extraction datasets which often focus on sung melodies [4].

5. DISCUSSION

5.1 Salience function and contour creation

By comparing the results obtained by SAL and C2 we can assess the influence of the salience function on methods based on pitch contour selection [20]. SAL obtains lower pitch related accuracies (RPA, RCA) than C2, especially for orchestral music. The difference between RPA and RCA is also greater in SAL than compared to C2, indicating SAL makes a larger amount of octave errors, especially for Orchset. This indicates that the signal representation yielded by the proposed pitch salience function S_c is effective at reducing octave errors, in concurrence with the observations made in [9]. C3 also provides a significantly higher accuracy in comparison to BIT, showing that the proposed salience function helps to improve melody extraction results also when combined with a pitch contour classification based method. Once again, this is particularly evident in orchestral music.

Note that even if the performance ceiling when creating the pitch contours from HS on MedleyDB is 2 percentage points higher than with S_c (see section 4.2), melody extraction results are better with S_c. This is due to the fact that the precision of the contour creation process with the proposed salience function is higher than with HS.

5.2 Pitch tracking method

By comparing the results of C2 and C3 we can assess the influence of the pitch tracking strategy, as both methods use the same contours as input. In MedleyDB, there is no significant difference between both methods in terms of overall accuracy, but the contour classification based method (C3) has a higher voicing recall for both melody definitions, while the contour selection method (C2) has a better RPA, RCA and VFA. This agrees with the findings from Bittner et al. [1] who also compared between both pitch tracking strategies using HS as the salience function. In the case of Orchset, the difference in OA is evident between C2-C3 (p = .004), since the classification based approach tends to classify most frames as voiced, which is beneficial when evaluating on this dataset. However, increasing the tolerance in C2 (C2*, $\nu = 1.4$) provides similar OA results: C2*-C3 (p = .78).

An analysis of feature importance for pitch contour classification (using S_c) revealed that salience features are the most discriminative in both datasets, especially mean salience. This suggests that the proposed salience function S_c is successful at assigning melody contours a higher salience compared to non-melody contours.

The most important difference between C2 and C3 is that C3 allows the model to be trained to fit the characteristics of a dataset, avoiding the parameter tuning necessary in rule-based approaches like [20]. The set of rules from [20] used in C2 are not tuned to orchestral music, which also explains why C2 obtains a lower OA on Orchset with the default parameters. Careful tuning could considerably improve the results.

5.3 Influence of parameters

We ran some additional experiments with C2 in order to investigate the influence of the parameters used to compute the pitch salience function and contour creation step. Several parameters affect the creation of the salience function [9], here we focus on the number of iterations used for the source-filter decomposition and how it affects the results obtained with the proposed salience function S_c. We found that on Orchset the drop in OA when reducing the number of iterations from 50 to 10 is less than 4%. On MedleyDB the change in OA is less than 1% when varying from 50 to 10 iterations. We also found that DUR is generally more sensitive to the decrease in number of iterations, which is a positive aspect of our proposed approach, given the high computational cost of the pitch salience estimation algorithm. For instance, DUR experiments a relative decrease in OA of around 7% when going from 50 to 10 iterations (on MedleyDB with MEL1 definition). The relative decrease in the case of C2 is less than 3%. The results reported in this study are based on 30 iterations.

Method	ν	VR	VFA	RPA	RCA	OA
DUR	-	1.0 (.00)	.99 (.09)	.68 (.20)	.80 (.12)	.63 (.20)
SAL	.2	.60 (.09)	.40 (.23)	.28 (.25)	.57 (.21)	.23 (.19)
SAL*	1.4	.81 (.07)	.57 (.25)	.30 (.26)	.57 (.21)	.29 (.23)
BIT	-	.69 (.14)	.45 (.25)	.35 (.17)	.53 (.15)	.37 (.16)
ESS	.2	.59 (.10)	.38 (.22)	.29 (.24)	.55 (.20)	.22 (.19)
C1	.2	.60 (.09)	.40 (.22)	.68 (.20)	.80 (.12)	.42 (.14)
C2	.2	.49 (.11)	.28 (.16)	.57 (.20)	.69 (.14)	.39 (.16)
C2*	1.4	.70 (.11)	.44 (.21)	.57 (.20)	.70 (.14)	.52 (.19)
C3	-	.73 (.12)	.46 (.23)	.53 (.19)	.65 (.14)	.53 (.18)

Table 3. Mean results (and standard deviation) over all excerpts for the five considered metrics, on Orchset. Parameter ν refers to the voicing threshold used in the methods based on pitch-contour selection. In the case of the classification-based methods (BIT and C3), this parameter is learnt from data. The sign * refers to the results obtained with the best v.

We also analysed the influence of Gaussian filtering (see Figure 1), by suppressing it from the salience function creation process. The effect is quite small on MedleyDB, but is more noticeable on Orchset where it results in a 4% point drop in OA. A possible explanation is that in symphonic music many instruments contribute to the melody but are not perfectly in tune. By smoothing the salience function we are able to increase the pitch salience of notes played by orchestral sections in unison. Pitch contour extraction and voicing detection parameters are more relevant, however. Overall accuracy generally increases on MedleyDB when the maximum allowed gap between pitches in a contour is decreased from 100 ms to 50 ms (50 ms is used in the reported experiments). Since SIMM can add noise to unvoiced frames, using the stricter threshold of 50 ms in the contour creation step can help filter some of this noise by preventing it from being tracked as part of a contour.

We also conducted a study of the effect of the voicing parameter (ν) on both C2 and SAL. A higher value results in less contours being filtered as unvoiced, which is beneficial on Orchset. A lower value (heavier filtering) is beneficial when evaluating against the MEL1 definition, since the melody is restricted to a single instrument. Varying ν from -1.4 to 1.0, the OA results with SAL range from .46 to .57 on MedleyDB MEL1, while with C2 they only range from .56 to .61. In the case of MEL2, the OA of SAL ranges from .46 to .54, while in the case of C2 the range is also smaller, from .57 to .60. This shows that the proposed method is more robust to the selection of the voicing parameter. While default contour creation parameters in ESS already provided satisfying results for C2 on MedleyDB, further tests on Orchset showed that they could be tuned to go up to 0.60 overall accuracy. In fact, just modifying the voicing parameter to $\nu = 1.4$ already increases the OA of C2 to 0.52. The highest overall accuracy obtained by SAL with the best parameter configuration on Orchset is 0.29 (see Table 3). This again shows that the same pitch contour selection based method can be improved with the proposed salience function, especially on orchestral music.

5.4 Pitch salience integration in contour creation

The benefits of combining a source-filter model and a pitch contour based tracking method have become evident by now, and each of the proposed combination approaches has its advantages and disadvantages. The main advantage of C1 is its simplicity, and that it always yields the same RPA as DUR, which is always the best in all datasets. The main disadvantage is that the contour creation process from SAL does not take advantage of the benefits of the pitch salience from DUR. This is the reason why it becomes important to integrate the source-filter model into the pitch contour creation process, as performed in C2 and C3. One difficulty of the integration is that the salience function from DUR needs to be adapted to the pitch contour creation framework. However, this improves overall accuracy in both MedleyDB and Orchset.

6. CONCLUSIONS AND FUTURE WORK

This paper presents a comparison of melody extraction methods based on source-filter models within a pitch contour based melody extraction framework. We propose three different combination methods, based on a melody oriented pitch salience function which adapts a source-filter model to the characteristics of the tracking algorithm. The adaptation is based on the combination with a salience function based on harmonic summation. We have shown that the proposed salience function helps improve pitch estimation accuracy and reduce octave errors in comparison to harmonic summation. This salience function consistently improves the mean overall accuracy results when it substitutes harmonic summation in pitch contour based tracking methods. This is true for both heuristic and machine-learning-based approaches, when evaluated on a large and varied set of data. Future work deals with improving the proposed salience function, in order to further reduce the amount of noise in unvoiced parts.

7. ACKNOWLEDGEMENTS

This work is partially supported by the European Union under the PHENICX project (FP7-ICT-601166) and the Spanish Ministry of Economy and Competitiveness under CASAS project (TIN2015-70816-R) and Maria de Maeztu Units of Excellence Programme (MDM-2015-0502).

8. REFERENCES

[1] R. Bittner, J. Salamon, S. Essid, and J. Bello. Melody extraction by contour classification. In *Proc. ISMIR*, pages 500–506, Málaga, Spain, Oct. 2015.

[2] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. Bello. Medleydb: a multitrack dataset for annotation-intensive mir research. In *Proc. ISMIR*, pages 155–160, Taipei, Taiwan, Oct. 2014.

[3] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X Serra. Essentia: an open source library for audio analysis. *ACM SIGMM Records*, 6, 2014.

[4] J. Bosch and E. Gómez. Melody extraction in symphonic classical music: a comparative study of mutual agreement between humans and algorithms. In *Proc. 9th Conference on Interdisciplinary Musicology – CIM14*, Berlin, Germany, Dec. 2014.

[5] J. Bosch and E. Gómez. Melody extraction by means of a source-filter model and pitch contour characterization (MIREX 2015). In *11th Music Information Retrieval Evaluation eXchange (MIREX), extended abstract*, Málaga, Spain, Oct. 2015.

[6] J. Bosch, R Marxer, and E. Gómez. Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music. *Journal of New Music Research*, DOI: 10.1080/09298215.2016.1182191, 2016.

[7] J Stephen Downie. The music information retrieval evaluation exchange (2005-2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[8] K. Dressler. Towards Computational Auditory Scene Analysis: Melody Extraction from Polyphonic Music. In *Proc. CMMR*, pages 319–334, London, UK, 2012.

[9] J. Durrieu, B. David, and G. Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *Sel. Top. Signal Process. IEEE J.*, 5(6):1180–1191, 2011.

[10] J. Durrieu, G. Richard, B. David, and C. Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *Audio, Speech, Lang. Process. IEEE Trans.*, 18(3):564–575, 2010.

[11] D. Ellis and G. Poliner. Classification-based melody transcription. *Machine Learning*, 65(2-3):439–456, 2006.

[12] B. Fuentes, A. Liutkus, R. Badeau, and G. Richard. Probabilistic model for main melody extraction using constant-Q transform. In *Proc. IEEE ICASSP*, pages 5357–5360, Kyoto, Japan, Mar. 2012. IEEE.

[13] M. Goto. A Real-Time Music-Scene-Description System: Predominant-F0 Estimation for Detecting Melody and Bass Lines in Real-World Audio Signals. *Speech Communication*, 43(4):311–329, September 2004.

[14] C. Hsu and J. Jang. Singing pitch extraction by voice vibrato/tremolo estimation and instrument partial deletion. In *Proc. ISMIR*, pages 525–530, Utrecht, Netherlands, Aug. 2010.

[15] A. Klapuri. Multiple fundamental frequency estimation by summing harmonic amplitudes. In *Proc. ISMIR*, pages 216–221, Victoria, Canada, Oct. 2006.

[16] M. Marolt. Audio melody extraction based on timbral similarity of melodic fragments. In *EUROCON 2005*, volume 2, pages 1288–1291. IEEE, 2005.

[17] A. Ozerov, P. Philippe, F. Bimbot, and R. Gribonval. Adaptation of bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(5):1564–1578, 2007.

[18] R. Paiva, T. Mendes, and A. Cardoso. Melody detection in polyphonic musical signals: Exploiting perceptual rules, note salience, and melodic smoothness. *Computer Music Journal*, 30(4):80–98, 2006.

[19] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis. mir_eval: A transparent implementation of common MIR metrics. In *Proc. ISMIR*, pages 367–372, Taipei, Taiwan, Oct. 2014.

[20] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Trans. Audio. Speech. Lang. Processing*, 20(6):1759–1770, 2012.

[21] J. Salamon, E. Gómez, D. Ellis, and G. Richard. Melody Extraction from Polyphonic Music Signals: Approaches, applications, and challenges. *IEEE Signal Process. Mag.*, 31:118–134, 2014.

[22] J. Salamon, G. Peeters, and A. Röbel. Statistical characterisation of melodic pitch contours and its application for melody extraction. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pages 187–192, Porto, Portugal, Oct. 2012.

AN ANALYSIS OF AGREEMENT IN CLASSICAL MUSIC PERCEPTION AND ITS RELATIONSHIP TO LISTENER CHARACTERISTICS

Markus Schedl, Hamid Eghbal-Zadeh
Johannes Kepler University
Linz, Austria
firstname.lastname@jku.at

Emilia Gómez
Universitat Pompeu Fabra
Barcelona, Spain
emilia.gomez@upf.edu

Marko Tkalčič
Free University of Bozen–Bolzano
Italy
marko.tkalcic@unibz.it

ABSTRACT

We present a study, carried out on 241 participants, which investigates on classical music material the agreement of listeners on perceptual music aspects (related to emotion, tempo, complexity, and instrumentation) and the relationship between listener characteristics and these aspects. For the currently popular task of music emotion recognition, the former question is particularly important when defining a ground truth of emotions perceived in a given music collection. We characterize listeners via a range of factors, including demographics, musical inclination, experience, and education, and personality traits. Participants rate the music material under investigation, i.e., 15 expert-defined segments of Beethoven's 3rd symphony, "Eroica", in terms of 10 emotions, perceived tempo, complexity, and number of instrument groups. Our study indicates only slight agreement on most perceptual aspects, but significant correlations between several listener characteristics and perceptual qualities.

1. INTRODUCTION

Music has always been closely related to human emotion. It can express emotions and humans can perceive and experience emotions when listening to music, e.g., [10, 22, 29]. In a uses and gratification analysis of why people listen to music [20], Lonsdale and North even identify emotion regulation as the main reason why people actively listen to music.

However, little is known about the influence of individual listener characteristics on music perception (emotion and other aspects) and whether listeners agree on such aspects at all. The aim of this paper is therefore to gain a better understanding of the *agreement on perceptual music aspects* and the *relationship between perceptual music aspects and personal characteristics*. To approach these two questions, we present and analyze results of a web-based user study involving 241 participants. We characterize listeners by demographics, music knowledge and experience, and personality. For our study, we focus on classical music, the repertoire under investigation being Beethoven's 3rd symphony, "Eroica". Responses of the listeners to the music are recorded via ratings of perceived emotions, tempo, complexity, and instrumentation.

In Section 2, we position our contribution within existing literature. Details on data acquisition and setup of the user study are provided in Section 3. Subsequently, we present and discuss the findings of our analysis on the agreement on perceptual aspects (Section 4) and on the relationship between these aspects and listener characteristics (Section 5). We round off by concluding remarks and a brief outlook to future research directions in Section 6.

2. RELATED WORK

This work connects to other investigations of music perception, to studies on personality in music, and to music emotion recognition.

Previous analyses on *music perception* have suggested that certain musical parameters especially influence the content of emotional responses, notably timbre, orchestration, acoustics, rhythm, melody, harmony, and structure [14]. For instance, Laurier created mappings between musical descriptors and emotion categories [19], but these emotion categories are limited to the five emotions happiness, sadness, anger, fear, and tenderness [3]. Rentfrow et al. identified five genre-free latent factors that reflect the affective response of listeners to music [25]. They named them "mellow", "urban", "sophisticated", "intense", and "campestral" music preference factors, yielding the acronym *MUSIC*. Not much research has been devoted to how listeners of different demographic, personality, and musical background experience different perceptual aspects of the same music. While there do exist several cross-cultural studies on music and perceived emotion [1, 8, 12, 15, 28], these studies tend to focus on greatly different cultures, rather than on more subtle differences such as age, gender, and musical experience or exposure.

Personality has been related to music preferences in a number of studies. Rentfrow and Gosling showed that personality traits are related to four preference dimensions: reflective and complex, intense and rebellious, upbeat and conventional, and energetic and rhythmic [26]. Furthermore, they found that personality-based stereotypes are

© Markus Schedl, Hamid Eghbal-Zadeh, Emilia Gómez, Marko Tkalčič. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Markus Schedl, Hamid Eghbal-Zadeh, Emilia Gómez, Marko Tkalčič. "An Analysis of Agreement in Classical Music Perception and Its Relationship to Listener Characteristics", 17th International Society for Music Information Retrieval Conference, 2016.

strongly correlated with music genre preferences [24]. Perhaps the most commonly used model of personality is the five factor model (FFM), which is composed of the factors openness, conscientiousness, extraversion, agreeableness, and neuroticism [21]. Employing this model in their study, Chamorro-Premuzic and Furnham found that people who score high on openness tend to consume music in a more rational way, while people who score high on neuroticism and those who score low on extraversion and conscientiousness tend to consume music to regulate their emotions [2]. Similarly, Ferwerda et al. showed that personality accounts for individual differences in mood regulation [6]. Personality has also been linked to how users tend to perceive and organize music [7].

Our work also connects to *music emotion recognition* (MER) at large, which has lately become a hot research topic [4, 11, 13, 18, 27, 30, 32]. It aims at automatically learning relationships between music audio or web features and emotion terms. However, common MER approaches assume that such a relationship exists, irrespective of a particular listener. In the study at hand, we take one step back and approach the question of whether listeners at all agree on certain emotions and other perceptive aspects when listening to classical music.

3. MATERIALS AND USER STUDY

3.1 Music Material

In our study, we focused on classical music and selected one particular piece, namely *Beethoven's 3rd symphony, "Eroica"*, from which we extracted 15 coherent excerpts [31]. This symphony is a well-known piece, also to many who are not much into classical music. We had to make this restriction to one piece to compare results between participants and keep them engaged throughout the questionnaire. Furthermore, this symphony was selected because of its focus in the PHENICX project, [1] the work at hand emerged from. Beethoven's "Eroica" is generally agreed on as a key composition of the symphonic repertoire, constituting a paradigm of formal complexity, as evidenced by the vast literature analyzing the symphony. We considered a performance by the Royal Concertgebouw Orchestra, Amsterdam. The 15 excerpts we used in the study were carefully selected by the authors, some of which are trained in music theory and performance, then reviewed by a musicologist. To this end, every section was first labeled with one of the nine emotions according to the Geneva Emotional Music Scale (GEMS) [33], judged based on musical elements. Then, the six emotions that appeared most frequently among the labels were identified. Three excerpts each for peacefulness, power, and tension, and two excerpts each for transcendence, joyful activation, and sadness, were finally selected. In this final selection step, we ensured that the segments covered a variety of musical characteristics, lasted the duration of a complete musical phrase, and strongly represented one of the above six emotions.

[1] http://phenicx.upf.edu

For the sake of reproducibility, interested readers can download the excerpts from http://mtg.upf.edu/download/datasets/phenicx-emotion.

3.2 Study Design

The study was conducted as online survey, accessible via a web interface. Participants were recruited by mass mail to all students of the Johannes Kepler University Linz and by posting to several research mailing lists. Announcements were also made on various social media platforms the authors are active on. In the survey, we first asked participants a range of questions, related to demographics, music education and experience, inclination to music and to classical music in particular, and familiarity with Beethoven's "Eroica". Subsequently, participants had to fill in a personality questionnaire, i.e., the standardized Ten Item Personality Instrument (TIPI) [9]. After having provided this personal information, we asked participants to listen to each of the 15 excerpts and provide ratings of perceptual qualities (emotions, tempo, complexity, and instrumentation). We ensured that participants actually listened to the excerpts by measuring the time they played each piece in the web browser. To describe emotions, we used the six emotions of the GEMS model most dominant in the music material (see above) and added five basic human emotions identified in psychological literature [5, 23]: transcendence, peacefulness, power, joyful activation, tension, sadness; anger, disgust, fear, surprise, tenderness. We added these additional emotions to complement the GEMS model with basic emotions not specifically targeted at music perception. The options available to participants for each answer, as well as their numeric coding for the following analysis, are provided in Table 1. Note that we are interested in *perceived* music qualities. Therefore, the questions were formulated according to the scheme "I perceive the music as ...".

3.3 Statistics of Participants

The survey was completed by 241 participants, taking them around 40 minutes on average. We had 123 male and 118 female participants. The vast majority of 217 participants were Austrians; other participants were Germans, Italians, Russians, Englishmen, and Spaniards. A limitation of the study is that participation was biased towards younger people, the median age of participants being 25 years. This can be explained by the large number of students among participants. However, the youngest participants were only 16, while the eldest one was 67. As for participants' music taste and listening frequency, on average subjects listen to classical music 2.6 hours per week, and to other genres 11 hours per week. Interestingly, the median for listening to classical music (1 hour per week) is much lower than the median of listening to other genres (8 hours per week). It thus seems that participants either love classical music and devote a lot of time to it, or do not listen to it at all. Less than half of the participants play an instrument (median of 0 hours per week), but most had some form of musical education, on average 6.8

Aspect	Options	Numeric encoding
Age	free form	years
Gender	male or female	—
Country	list selection from 193 countries	—
Listening classical	free form	hours per week
Listening non-classical	free form	hours per week
Playing instrument	free form	hours per week
Musical education	free form	years
Concerts classical	free form	attendances per year
Concerts non-classical	free form	attendances per year
Familiar with "Eroica"	unfamiliar, somewhat familiar, very familiar	0–2
All personality traits	strongly disagree–strongly agree	1–7
All emotions	strongly disagree, disagree, neither agree nor disagree, agree, strongly agree, don't know	0–4, -1
Perceived tempo	slow, fast, don't know	0, 1, -1
Perceived complexity	very low–very high, don't know	0–4, -1
Kinds of instruments	1, 2, 3, 4, more, don't know	1, 2, 3, 4, 5, -1
Description of the excerpt	free form	—

Table 1. Options available to participants and numerical encoding of the answers using for analysis.

Aspect	μ	σ	med	min	max
Age	27.35	8.47	25	16	67
Listening classical (hrs/week)	2.56	5.20	1	0	40
Listening non-classical	11.16	11.86	8	0	70
Playing instrument	1.93	4.23	0	0	40
Musical education	6.77	6.39	5	0	33
Concerts classical	2.43	5.28	1	0	40
Concerts non-classical	3.93	6.70	2	0	70
Familiar with "Eroica"	0.83	0.64	1	0	2

Table 2. Basic statistics of the participants. μ = mean, σ = standard deviation, med = median

Personality trait	μ	σ	med	min	max
Extraverted	4.27	1.88	5	1	7
Critical	4.54	1.68	5	1	7
Dependable	5.27	1.43	6	1	7
Anxious	3.17	1.64	3	1	7
Open to new experiences	5.59	1.27	6	2	7
Reserved	4.41	1.81	5	1	7
Sympathetic	5.39	1.32	6	1	7
Disorganized	2.83	1.69	2	1	7
Calm	5.01	1.56	6	1	7
Conventional	2.84	1.63	2	1	7

Table 3. Personality statistics of participants. μ = mean, σ = standard deviation, med = median

years. Participants attend on average 2 classical and 4 non-classical concerts per year, but the median values are again smaller (1 and 2 concerts, respectively). Many participants do not attend concerts at all: 39% do not attend a single classical concert, 22% do not attend a single concert of another genre per year. Most participants were not (72 or 30%) or somewhat (137 or 57%) familiar with Beethoven's "Eroica". Only 32 (14%) indicated to be very familiar with the piece. Analyzing the personality traits, shown in Table 3, we observe that subjects tend to regard themselves as open to new experiences, sympathetic, calm, but also dependable (average and median ratings are at least "agree a little"). On the other hand, they negate being disorganized, conventional, and anxious (average and median ratings are at most "disagree a little").

4. LISTENER AGREEMENT

We compute the agreement on all perceptive aspects under investigation. To this end, we use *Krippendorff's* α score for inter-rater agreement [16], computed on the ratings given by participants for each segment separately and subsequently averaged. We excluded from the calculations "don't know" answers, i.e., treated them as missing values.

Table 4 shows the overall mean ratings, standard deviations, and agreement scores among participants for each investigated aspect, macro-averaged over all segments. We observe that participants give highest average ratings (column μ in Table 4) to the aspects of power and tension, followed by transcendence and joyful activation. Lowest ratings are given to fear, sadness, anger, and — much below — disgust. Overall, it seems that the aspects ranging in the lower arousal range (sadness, peacefulness, etc.) are perceived to a smaller degree in the music material under consideration. Tempo is, on average, neither perceived as particularly low nor high. So is complexity. As for instrumentation, overall, most participants could distinguish 4 kinds of instruments.

As for agreement, the study evidences a low to moderate agreement for most aspects, according to Krippendorff's α. Participants do not (0.00–0.20) or at most slightly (0.21–0.40) agree on most perceptual aspects.

Aspect	Scale	μ	σ	α
Transcendence	0–4	2.215	1.095	0.010
Peacefulness	0–4	1.812	0.986	**0.450**
Power	0–4	2.477	0.937	**0.450**
Joyful activation	0–4	2.048	1.059	*0.320*
Tension	0–4	2.318	1.121	*0.222*
Sadness	0–4	1.233	0.979	*0.298*
Anger	0–4	1.204	1.008	*0.300*
Disgust	0–4	0.808	0.941	0.128
Fear	0–4	1.292	1.084	*0.276*
Surprise	0–4	1.790	1.162	0.054
Tenderness	0–4	1.687	1.046	*0.366*
Tempo	0–1	0.460	0.337	**0.513**
Complexity	0–4	2.240	0.864	0.116
Instrument kinds	1–5	3.899	0.980	0.077

Table 4. Mean μ, standard deviation σ, and agreement score (Krippendorff's α) for investigated aspects of music perception. Italic font is used to indicate slight agreement. Bold face is used to denote moderate agreement.

The values indicating moderate agreement (0.41–0.60) according to [17] are printed in bold in Table 4, whereas slight agreement is indicated by italics. Highest agreement among the emotion aspects is found for peacefulness and power, while tempo shows the highest agreement among all investigated aspects. Slight agreement can be observed for joyful activation, tension, sadness, anger, fear, and tenderness. No relevant agreement is observed for transcendence, disgust, surprise, as well as perceived complexity and number of instrument groups. Perceived complexity is presumably a highly subjective aspect. Furthermore, it seems that there is a discrepancy between listeners with regard to their ability to distinguish different instrumentations, which is presumably due to different music knowledge and expertise levels.

5. LISTENER CHARACTERISTICS AND PERCEPTUAL ASPECTS

We investigate whether there exists a significant relationship between listener characteristics and the perceptual aspects under investigation. To this end, we calculate Pearson's correlation coefficient between the respective numerically encoded factors, according to Table 1, treating each user–segment pair as one observation. Table 5 shows the correlation values for all listener characteristics (rows) and perceptual aspects (columns). While most correlations are not very pronounced, several are significant (at $p < 0.05$), where p values are the probability of observing by chance a correlation as large as the observed one, when the true correlation is 0.

Reviewing the results, a remarkable observation is the significant correlations between factors of musical background and knowledge (listening to classical music, playing an instrument, musical education, concert attendances, familiarity with the piece) and perceived number of instrument groups. Participants with a stronger musical background therefore seem to be able to distinguish more instruments. While participants scoring high on conventionalism show negative correlation with the perceived number of instrument groups, the opposite it true for people who are open to new experiences. As for participants' age, older people tend to perceive the music as more joyful and less fearsome. Frequent listeners of classical music tend to perceive the piece as more powerful, transcendent, and tender, but less fearsome. On the other hand, listeners of other genres perceive more anger and surprise. Playing an instrument, extensive musical education, and frequent classical concert attendances show a positive correlation with perceived power and tension, while attending non-classical concerts seem to have no influence on emotion perception. Participants who are familiar with the "Eroica" overall tend to perceive it as more transcendent, powerful, joyful, and tender.

Among the personality traits, most show little correlation with the perceptual aspects. However, openness to new experiences is significantly correlated with the emotion categories transcendence, peacefulness, joyful activation, and tenderness, as well as tempo and instrumentation. Disorganized people tend to rate the piece higher on sadness, anger, disgust, but also on tenderness. Sympathetic subjects on average give higher ratings to aspects of peacefulness, tenderness, tempo, and number of instrument groups. Calm participants perceive the music as more peaceful, joyful, and tender than others, while conventionalists perceive it as less transcendent and tense.

6. CONCLUSIONS AND FUTURE DIRECTIONS

In the presented study, we addressed two research questions. First, we investigated whether listeners agree on a range of perceptual aspects (emotions, tempo, complexity, and instrumentation) in classical music material, represented by excerpts from Beethoven's 3rd symphony, "Eroica". Only for the perceived emotions peacefulness and power as well as for the perceived tempo, moderate agreement was found. On other aspects, participants agreed only slightly or not at all. The second question we approached in the study is the relationship between listener characteristics (demographics, musical background, and personality traits) and ratings given to the perceptual aspects. Among others, we found significant correlations between musical knowledge and perceived number of instrument groups, which might not be too surprising. We further identified slight, but significant positive correlations between aspects of musical inclination or knowledge and perceived power and tension. Several correlations were also found for the personality traits open to new experiences, sympathetic, disorganized, calm, and conventional.

As part of future work, we plan to assess to which extent the investigated perceptual aspects can be related to music audio descriptors. We would further like to analyze the impact of listener characteristics on the agreement scores of perceptual aspects. Furthermore, a cross-correlation analysis between ratings of the perceived qualities could reveal which emotions (or other investigated aspects) are

	Trans.	Peace.	Power	Joyful.	Tension	Sadness	Anger	Disgust	Fear	Surprise	Tender	Tempo	Compl.	Instr.
Age	**0.155**	0.040	0.102	**0.261**	0.075	-0.081	-0.110	-0.002	**-0.186**	-0.015	0.104	-0.031	-0.019	-0.026
Listening classical	**0.203**	0.112	**0.212**	0.078	0.019	-0.082	-0.090	-0.105	**-0.190**	-0.029	**0.148**	0.028	0.123	**0.192**
Listening non-classical	0.085	0.092	0.121	0.007	0.033	0.028	**0.139**	0.042	0.078	**0.149**	0.054	0.122	0.064	-0.036
Playing instrument	0.085	-0.016	**0.133**	0.010	**0.190**	0.077	0.113	0.073	0.050	0.042	0.014	0.061	0.012	**0.259**
Musical education	**0.140**	-0.073	**0.143**	0.007	**0.170**	0.029	0.101	0.085	0.008	-0.064	0.007	0.077	0.076	**0.418**
Concerts classical	**0.170**	0.065	**0.175**	0.108	**0.192**	-0.015	-0.033	-0.028	-0.065	-0.046	0.076	0.017	0.086	**0.243**
Concerts non-classical	0.114	-0.004	0.048	-0.008	0.099	0.080	0.079	0.061	0.091	0.069	-0.003	0.106	0.045	**0.153**
Familiar with “Eroica”	**0.141**	0.118	**0.211**	**0.184**	0.116	-0.045	0.057	0.026	-0.018	0.004	**0.149**	0.056	0.096	**0.242**
Extraverted	0.045	0.024	0.120	0.065	0.022	0.031	-0.014	-0.027	0.007	0.041	**0.166**	0.112	0.059	0.066
Critical	0.010	0.031	0.094	0.081	0.049	0.037	-0.035	-0.041	-0.011	**-0.141**	0.043	0.066	0.075	0.049
Dependable	0.054	-0.098	-0.074	-0.098	0.009	-0.049	-0.065	-0.035	0.011	-0.018	0.007	-0.023	-0.075	0.033
Anxious	-0.084	-0.054	-0.108	-0.114	-0.108	-0.003	0.017	0.064	0.055	0.023	-0.089	-0.072	-0.054	-0.087
Open to new exp.	**0.159**	**0.139**	0.108	**0.181**	0.054	0.053	0.010	0.005	-0.003	0.009	**0.222**	**0.173**	0.006	**0.201**
Reserved	-0.049	0.033	-0.112	-0.057	-0.095	-0.038	-0.033	-0.014	-0.045	-0.042	-0.084	-0.026	-0.054	-0.061
Sympathetic	0.077	**0.147**	0.098	0.107	0.059	-0.031	-0.012	0.020	0.026	0.078	**0.166**	**0.148**	0.015	**0.134**
Disorganized	0.076	0.120	0.032	0.083	0.114	**0.167**	**0.157**	**0.146**	0.116	0.111	**0.129**	**0.130**	-0.014	-0.069
Calm	0.076	**0.142**	-0.002	**0.153**	-0.032	-0.023	-0.044	-0.060	0.031	-0.063	**0.132**	0.069	**0.153**	**0.135**
Conventional	**-0.145**	0.099	-0.048	0.012	**-0.135**	0.050	0.087	0.070	0.102	0.008	-0.058	-0.040	-0.002	**-0.129**

Table 5. Correlation between demographics, music expertise, and personality traits on the one hand, and aspects of music perception on the other. Significant results ($p < 0.05$) are depicted in bold face.

frequently perceived together. Finally, we would like to perform a larger study, involving on the one hand a larger genre repertoire and on the other an audience more diverse in terms of cultural background.

7. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Fund (FWF): P25655 and was made possible by the EU FP7 project no. 601166 (“PHENICX”).

8. REFERENCES

[1] L.L. Balkwill and W.F. Thompson. A cross-cultural investigation of tlhe perception of emotion in music. *Music Perception*, 17(1):43–64, 1999.

[2] Tomas Chamorro-Premuzic and Adrian Furnham. Personality and music: can traits explain how people use music in everyday life? *British Journal of Psychology*, 98:175–85, May 2007.

[3] T. Eerola and J.K. Vuoskoski. A comparison of the discrete and dimensional models of emotion in music. *Psychology of Music*, 39(1):18–49, 2011.

[4] Tuomas Eerola, Olivier Lartillot, and Petri Toiviainen. Prediction of Multidimensional Emotional Ratings in Music from Audio Using Multivariate Regression Models. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, Kobe, Japan, October 2009.

[5] Paul Ekman. *Basic Emotions*, pages 45–60. John Wiley & Sons Ltd., New York, NY, USA, 1999.

[6] Bruce Ferwerda, Markus Schedl, and Marko Tkalcic. Personality & Emotional States : Understanding Users Music Listening Needs. In Alexandra Cristea, Judith Masthoff, Alan Said, and Nava Tintarev, editors, *UMAP 2015 Extended Proceedings*, 2015.

[7] Bruce Ferwerda, Emily Yang, Markus Schedl, and Marko Tkalčič. Personality Traits Predict Music Taxonomy Preferences. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA ’15*, pages 2241–2246, 2015.

[8] T. Fritz, S. Jentschke, N. Gosselin, D. Sammler, I. Peretz, R. Turner, A.D. Friederici, and S. Koelsch. Universal recognition of three basic emotions in music. *Current Biology*, 19(7):573–576, 2009.

[9] Samuel D. Gosling, Peter J. Rentfrow, and William B. Swann Jr. A very brief measure of the Big-Five personality domains. *Journal of Research in Personality*, 37(6):504–528, December 2003.

[10] K. Hevner. Expression in Music: A Discussion of Experimental Studies and Theories. *Psychological Review*, 42, March 1935.

[11] Xiao Hu, J. Stephen Downie, and Andreas F. Ehmann. Lyric Text Mining in Music Mood Classification. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, Kobe, Japan, October 2009.

[12] Xiao Hu and Jin Ha Lee. A Cross-cultural Study of Music Mood Perception Between American and Chinese Listeners. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, Porto, Portugal, October 2012.

[13] A Huq, J.P. Bello, and R. Rowe. Automated Music Emotion Recognition: A Systematic Evaluation. *Journal of New Music Research*, 39(3):227–244, November 2010.

[14] P.N. Juslin and P. Laukka. Expression, perception, and induction of musical emotions: A review and a questionnaire study of everyday listening. *Journal of New Music Research*, 33(2):217–238, 2004.

[15] Katerina Kosta, Yading Song, Gyorgy Fazekas, and Mark B. Sandler. A Study of Cultural Dependence of Perceived Mood in Greek Music. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, November 2013.

[16] Klaus Krippendorff. *Content Analysis – An Introduction to Its Methodology*. SAGE, 3rd edition, 2013.

[17] J.R. Landis and G.G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174, 1977.

[18] C. Laurier, J. Grivolla, and P. Herrera. Multimodal Music Mood Classification Using Audio and Lyrics. In *Proceedings of the 7th International Conference on Machine Learning and Applications (ICMLA)*, pages 688–693, December 2008.

[19] Cyril Laurier. *Automatic Classification of Music Mood by Content-based Analysis*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2011.

[20] Adam J Lonsdale and Adrian C North. Why do we listen to music? A uses and gratifications analysis. *British Journal of Psychology*, 102(1):108–34, February 2011.

[21] Robert R McCrae and Oliver P John. An Introduction to the Five-Factor Model and its Applications. *Journal of Personality*, 60(2):175–215, 1992.

[22] A. Pike. A phenomenological analysis of emotional experience in music. *Journal of Research in Music Education*, 20:262–267, 1972.

[23] Robert Plutchik. The Nature of Emotions. *American Scientist*, 89(4):344–350, 2001.

[24] P. J. Rentfrow and S. D. Gosling. The content and validity of music-genre stereotypes among college students. *Psychology of Music*, 35(2):306–326, February 2007.

[25] Peter J. Rentfrow, Lewis R. Goldberg, and Daniel J. Levitin. The structure of musical preferences: A five-factor model. *Journal of Personality and Social Psychology*, 100(6):1139–1157, 2011.

[26] Peter J. Rentfrow and Samuel D. Gosling. The do re mi's of everyday life: The structure and personality correlates of music preferences. *Journal of Personality and Social Psychology*, 84(6):1236–1256, 2003.

[27] Erik M. Schmidt and Youngmoo E. Kim. Projection of Acoustic Features to Continuous Valence-Arousal Mood Labels via Regression. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, Kobe, Japan, October 2009.

[28] Abhishek Singhi and Daniel G. Brown. On Cultural, Textual and Experiential Aspects of Music Mood. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, Taiwan, October 2014.

[29] Justin Sloboda and Patrick Juslin. *Music and Emotion: Theory and Research*. Oxford University Press, 2001.

[30] Yading Song, Simon Dixon, and Marcus Pearce. Evaluation of Musical Features for Emotion Classification. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, Porto, Portugal, October 2012.

[31] Erika Trent and Emilia Gómez. Correlations Between Musical Descriptors and Emotions Recognized in Beethoven's Eroica. In *Proceedings of the 9th Triennial Conference of the European Society for the Cognitive Sciences of Music (ESCOM)*, Manchester, UK, August 2015.

[32] Yi-Hsuan Yang and Homer H. Chen. Machine recognition of music emotion: A review. *Transactions on Intelligent Systems and Technology*, 3(3), May 2013.

[33] M. Zenter, D. Grandjean, and K.R. Scherer. Emotions evoked by the sound of music: Characterization, classification, and measurement. *Emotion*, 8:494, 2008.

AN ATTACK/DECAY MODEL FOR PIANO TRANSCRIPTION

Tian Cheng, Matthias Mauch, Emmanouil Benetos and Simon Dixon
Centre for Digital Music, Queen Mary University of London
{t.cheng, m.mauch, emmanouil.benetos, s.e.dixon}@qmul.ac.uk

ABSTRACT

We demonstrate that piano transcription performance for a known piano can be improved by explicitly modelling piano acoustical features. The proposed method is based on non-negative matrix factorisation, with the following three refinements: (1) introduction of attack and harmonic decay components; (2) use of a spike-shaped note activation that is shared by these components; (3) modelling the harmonic decay with an exponential function. Transcription is performed in a supervised way, with the training and test datasets produced by the same piano. First we train parameters for the attack and decay components on isolated notes, then update only the note activations for transcription. Experiments show that the proposed model achieves 82% on note-wise and 79% on frame-wise F-measures on the 'ENSTDkCl' subset of the MAPS database, outperforming the current published state of the art.

1. INTRODUCTION

Automatic music transcription (AMT) converts a musical recording into a symbolic representation, i.e. a set of note events, each consisting of pitch, onset time and duration. Non-negative matrix factorisation (NMF) is commonly used in the AMT area for over a decade since [1]. It factorises a spectrogram (or other time-frequency representation, e.g. Constant-Q transform) of a music signal into non-negative spectral bases and corresponding activations. With constraints such as sparsity [2], temporal continuity [3] and harmonicity [4], NMF provides a meaningful mid-level representation (the activation matrix) for transcription. A basic NMF is performed column by column, so NMF-based transcription systems usually provide frame-wise representations with note transcription as a post-processing step [5].

One direction of AMT is to focus on instrument-specific music, in order to make use of more information from instrumental physics and acoustics [5]. For piano sounds, several acoustics-associated features, such as inharmonicity, time-varying timbre and decaying energy, are examined for their utilities in transcription. Rigaud *et al.* show that an explicit inharmonicity model leads to improvement in piano transcription [6], while a note-dependent inharmonicity parameter is needed for initialisation. Modelling time-varying timbre not only provides a better reconstruction of the spectrogram, but also improves note tracking results by imposing constraints between note stages (attack, sustain and decay) [7, 8]. For decaying energy, Chen *et al.*'s preliminary work uses an exponential model for energy evolution of notes [9]. Berg-Kirkpatrick *et al.* represent the energy evolution of a piano note by a trained envelope [10]. Cogliati and Duan use a sum of two decaying exponentials to approximate decays of piano partials [11]. Ewert *et al.* represent both time-varying timbre and temporal evolution of piano notes by time-frequency patches [12]. Temporal evolution modelling allows a note event to be represented by a single amplitude parameter for its whole duration, enabling the development of note-level systems with promising transcription results [9, 10, 12].

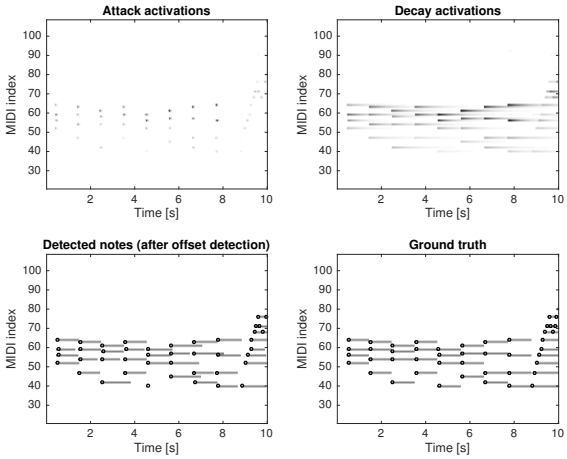


Figure 1: An example of output from the proposed model.

The proposed method is also motivated by piano acoustics. Based on our previous studies on piano decay, we know that exponential decay explains the major energy evolution for each partial in spite of various decay patterns [13]. Here, we further simplify the decay stage using an exponential decay function and a harmonic template per pitch. We separately represent the attack stage for the percussive onset of piano sounds. These two stages are coupled by shared note activations. A supervised NMF framework is used to estimate note activations, and hence activations of the attack and decay stages (see Figure 1). We detect note onsets by peak-picking on attack activations, then offsets for each pitch individually. Experiments

© Tian Cheng, Matthias Mauch, Emmanouil Benetos and Simon Dixon. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Tian Cheng, Matthias Mauch, Emmanouil Benetos and Simon Dixon. "An Attack/Decay Model for Piano Transcription", 17th International Society for Music Information Retrieval Conference, 2016.

show that the proposed method significantly improves supervised piano transcription, and compares favourably to other state-of-the-art techniques.

The proposed method is explained in Section 2. The transcription and comparison experiments are described in Section 3. Conclusions and discussions are drawn in Section 4.

2. METHOD

In this section we first introduce the attack and decay model for piano sounds. Parameters are estimated using a sparse NMF. Then we explain onset and offset detection methods, respectively.

2.1 A model of attack and decay

A piano sound is produced by a hammer hitting the string(s) of a key. It starts with a large energy, then decays till the end of the note. At the attack stage, the strike of the hammer produces a percussive sound. It evolves quickly to an almost harmonic pitched sound, and then immediately enters the decay stage. Considering the different spectral and temporal features, we reconstruct these two phases individually. The attack sound is generated by:

$$V^a_{ft} = \sum_{k=1}^{K} W^a_{fk} H^a_{kt}, \tag{1}$$

where $\mathbf{V^a}$ is the reconstructed spectrogram of the attack phase, as shown in Figure 2(d), and $\mathbf{W^a}$ is the percussive template (Figure 2(e)). $f \in [1, F]$ is the frequency bin, $t \in [1, T]$ indicates the time frame, and $k \in [1, K]$ is the pitch index. Attack activations $\mathbf{H^a}$ (Figure 2(c)) are formulated by the convolution as follows:

$$H^a_{kt} = \sum_{\tau=t-T_t}^{t+T_t} H_{k\tau} P(t-\tau), \tag{2}$$

where $\mathbf{H}$ are spike-shaped note activations, shown in Figure 2(b). $\mathbf{P}$ is the transient pattern, and its typical shape is shown in Figure 5. The range of the transient pattern is determined by the overlap in the spectrogram, with T_t equal to the ratio of the window size and frame hop size.

For the decay part we assume that piano notes decay approximately exponentially [13, 14]. The harmonic decay is generated by

$$V^d_{ft} = \sum_{k=1}^{K} W^d_{fk} H^d_{kt}, \tag{3}$$

where $\mathbf{V^d}$ is the reconstructed spectrogram of the decay phase (Figure 2(g)), and $\mathbf{W^d}$ is the harmonic template (Figure 2(h)). Decay activations $\mathbf{H^d}$ in Figure 2(f) are generated by convolving activations with an exponentially decaying function:

$$H^d_{kt} = \sum_{\tau=1}^{t} H_{k\tau} e^{-(t-\tau)\alpha_k}, \tag{4}$$

where α_k are decay factors, and e^{α_k} indicates the decay rate per frame for pitch k. Offsets are not modelled; instead

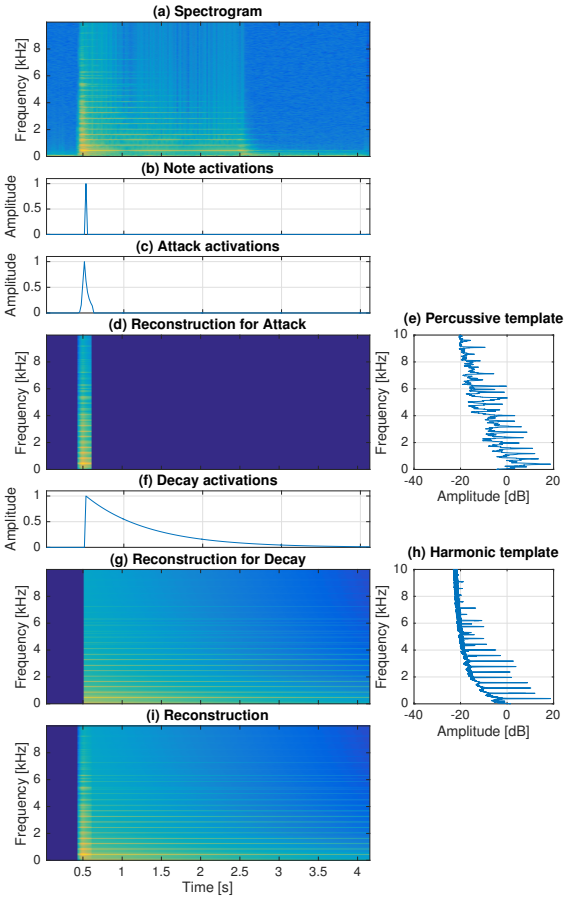


Figure 2: An illustration of the proposed model (note D3 with the MIDI index of 50).

it is assumed that the energy of a note decays forever. Then the complete model is formulated as follows:

$$\begin{aligned} V_{ft} &= V^a_{ft} + V^d_{ft} \\ &= \sum_{k=1}^{K} W^a_{fk} \sum_{\tau=t-T_t}^{t+T_t} H_{k\tau} P(t-\tau) \\ &+ \sum_{k=1}^{K} W^d_{fk} \sum_{\tau=1}^{t} H_{k\tau} e^{-(t-\tau)\alpha_k}, \end{aligned} \tag{5}$$

where $\mathbf{V}$ is the reconstruction of the whole note, as shown in Figure 2(i).

Parameters $\theta \in \{\mathbf{W^a}, \mathbf{W^d}, \mathbf{H}, \mathbf{P}, \alpha\}$ are estimated by minimising the difference between the spectrogram $\mathbf{X}$ and the reconstruction $\mathbf{V}$ by multiplicative update rules [15]. The derivative of the cost function D with respect to θ is written as a difference of two non-negative functions:

$$\nabla_\theta D(\theta) = \nabla^+_\theta D(\theta) - \nabla^-_\theta D(\theta). \tag{6}$$

The multiplicative algorithm is given by

$$\theta \leftarrow \theta . \nabla^-_\theta D(\theta) ./ \nabla^+_\theta D(\theta). \tag{7}$$

We employ the β-divergence as the cost function. The full update equations are provided online. [1]

[1] https://code.soundsoftware.ac.uk/projects/decay-model-for-piano-transcription.

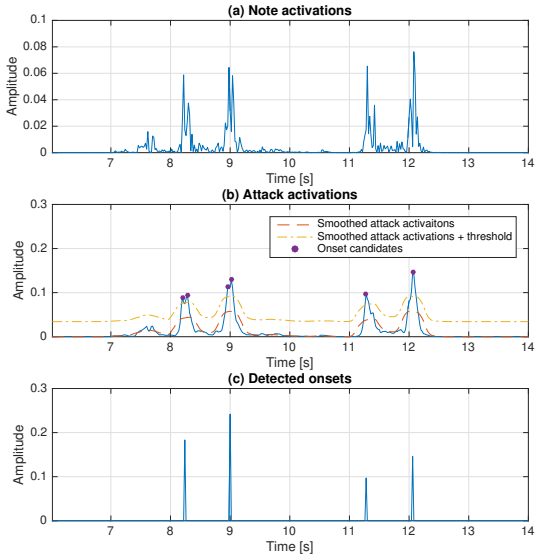


Figure 3: Example of onset detection showing how activations are processed.

2.2 Sparsity

To ensure spike-shaped note activations, we simply impose sparsity on activations $\mathbf{H}$ using element-wise exponentiation after each iteration:

$$\mathbf{H} = \mathbf{H}^{\gamma}, \tag{8}$$

where γ is the sparsity factor, usually larger than 1. The larger the factor is, the sparser the activations are.

A preliminary test confirmed that the number of peaks in activations decreases as the degree of sparsity increases. We also apply an annealing sparsity factor [16], which means a continuously changing factor. In this paper, we set γ to increase from 1 to $\gamma_a \in [1.01, 1.05]$ gradually within the iterations.

2.3 Onset detection

Different playing styles and overlapping between notes may cause a mismatch between the observed attack energy and the trained transient pattern. This results in multiple peaks around onsets in the activations. Figures 3(a) and (b) show note activations and attack activations of pitch G2 in a music excerpt, respectively. Attack activations indicate the actual transient patterns of notes obtained by the proposed model. Therefore, we detect onsets from attack activations by peak-picking. First, we compute smoothed attack activations for each pitch, using a moving average filter with a window of 20 bins. Only peaks which exceed smoothed attack activations by a threshold will be detected as onset candidates, as shown in Figure 3(b). The threshold is adapted to each piece with the parameter δ:

$$Thre = \delta \max_{k,t} H^a_{k,t}. \tag{9}$$

We test various $\delta \in \{-21\text{dB}, -22\text{dB}, \ldots, -40\text{dB}\}$ in this paper.

We find that there are still double peaks around onsets after thresholding. In order to deal with this problem, we simply merge pairs of peaks which are too close to each other. We set the minimal interval between two successive notes of the same pitch to be 0.1 second. If the interval between two peaks is smaller than the minimal interval, we generate a new peak. The index of the new peak is a weighted average of the indices of the two peaks, while its amplitude is the sum of that of the two peaks. Figure 3(c) shows detected onsets after merging double peaks. We apply the above process again to get rid of triple peaks.

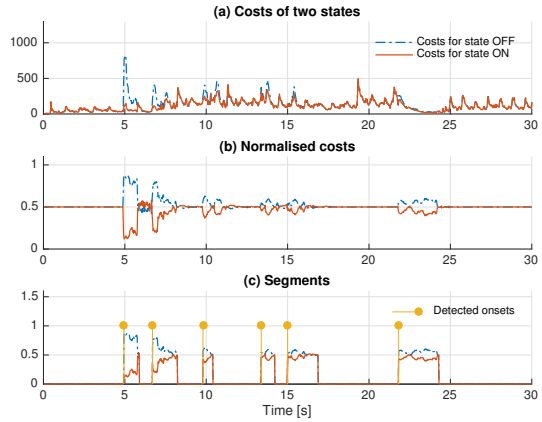


Figure 4: Costs and segments for pitch F3 (MIDI index 53).

2.4 Offset detection

We adapt the method of [12] to detect the offsets by dynamic programming. For each pitch, there are two states $s \in \{0, 1\}$, denoting state 'off' and 'on' respectively. The costs are defined below:

$$C_k(s,t) = \begin{cases} \sum_{f=1}^{F} D_{KL}(X_{ft}, V_{ft} - V^k_{ft}), & s = 0 \\ \sum_{f=1}^{F} D_{KL}(X_{ft}, V_{ft}), & s = 1 \end{cases} \tag{10}$$

where V^k is the reconstruction of pitch k, and $V - V^k$ is the reconstruction excluding pitch k. $D_{KL}(a, b)$ denotes the KL-divergence between a and b. Then we normalise the costs per pitch to sum to 1 in all frames: $\widetilde{C_k}(s,t) = C_k(s,t) / \sum_{\tilde{s}} C_k(\tilde{s}, t)$. Figures 4 (a) and (b) show the costs and normalised costs for pitch F3 in a music piece, respectively.

We can find the optimal state sequence by applying dynamic programming on the normalised costs. To do this, we need an accumulated cost matrix and a step matrix to store the smallest accumulated costs and previous states. The accumulated cost matrix D_k is recursively defined as

$$D_k(s,t) = \begin{cases} \min_{\tilde{s} \in \{0,1\}} (D_k(\tilde{s}, t-1) + \widetilde{C_k}(s,t) w(\tilde{s}, s)), & t > 1 \\ \widetilde{C_k}(s,t), & t = 1 \end{cases} \tag{11}$$

where w is the weight matrix, which favours self-transitions, in order to obtain a smoother sequence. In this paper, the weights are $[0.5, 0.55; 0.55, 0.5]$. The step matrix E is defined as follows:

$$E_k(s,t) = \arg \min_{\tilde{s} \in \{0,1\}} (D_k(\tilde{s}, t-1) + \widetilde{C_k}(s,t) w(\tilde{s}, s)), t > 1 \tag{12}$$

The states are given by

$$S_k(t) = \begin{cases} \arg\min_{\tilde{s}\in\{0,1\}} D_k(\tilde{s}, t), & t = T \\ E_k(S_k(t+1), t+1), & t \in [1, T-1] \end{cases} \quad (13)$$

We find that when the activation of the pitch is 0 or very small, the costs of two states are the same or very close, and no state transition occurs. In these parts, the pitch state is off, while dynamic programming can not jump out from the previous state. In order to deal with this problem we need to exclude these parts before applying dynamic programming. Figure 4(c) shows the segmentation by detected onsets and the costs. Each segment starts at a detected onset and ends when the difference of the smoothed normalised costs is less than a set threshold. We track the states of the pitch for each segment individually.

3. EXPERIMENTS

In the experiments we first analyse the proposed model's performance on music pieces produced by a real piano from the MAPS database [17]. Then we compare to three state-of-the-art transcription methods on this dataset and two other synthetic datasets.

To compute the spectrogram, frames are segmented by a 4096-sample Hamming window with a hop-size of 882. [2] A discrete Fourier Transform is performed on each frame with 2-fold zero-padding. Sample frequency f_s is 44100Hz. To lessen the influence of beats in the decay stage [13], we smooth the spectrogram with a median filter covering 100ms. During parameter estimation, we use the KL-divergence ($\beta = 1$) as the cost function. The proposed model is iterated for 50 times in all experiments to achieve convergence.

Systems are evaluated by precision (P), recall (R) and F-measure (F), defined as:

$$P = \frac{N_{tp}}{N_{tp}+N_{fp}},\ R = \frac{N_{tp}}{N_{tp}+N_{fn}},\ F = 2 \times \frac{P \times R}{P+R},$$

where N_{tp}, N_{fp}, N_{fn} are the numbers of true positives, false positives and false negatives, respectively. In addition, we use the accuracy in [18] to indicate the overall accuracy: $A = \frac{N_{tp}}{N_{tp}+N_{fp}+N_{fn}}$. We employ both frame-wise and note-wise evaluation [19], denoted by subscript 'f' and 'on', respectively.

3.1 Transcription experiment

The main transcription experiment is performed on the 'ENSTDkCL' subset of the MAPS database [17]. The piano sounds of this subset are recorded on a Disklavier piano. We train percussive and harmonic templates, decay rates and the transient pattern on the isolated notes produced by the same piano. The transcription experiment is run on the music pieces using the first 30s of each piece. [3]

[2] A 20ms hop size is used to reduce computation time. For frame-wise evaluation, transcription results are represented with a hop size of 10ms by duplicating every frame.

[3] The proposed model runs at about 3 × real-time using MATLAB on a MacBook Pro laptop (I7, 2.2GHz, 16GB).

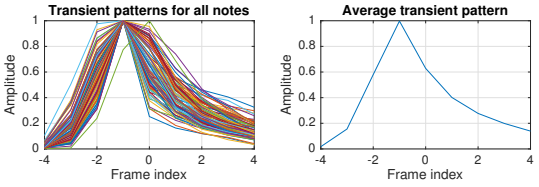


Figure 5: Transient patterns.

Table 1: Note tracking results with different fixed sparsity factors (above) and annealing sparsity factors (below).

γ	P_{on}	R_{on}	F_{on}	A_{on}	δ(dB)
1.00	**88.52**	77.70	**82.24**	**70.54**	-29
1.01	87.70	**78.18**	82.23	70.53	-30
1.02	87.67	77.36	81.80	69.87	-30
1.03	87.22	77.31	81.62	69.66	-31
1.04	86.95	76.84	81.26	69.17	-32
1.05	86.38	75.99	80.51	68.14	-33
$1 \rightarrow 1.01$	87.77	**78.08**	82.18	70.49	-30
$1 \rightarrow 1.02$	**88.49**	77.79	**82.36**	**70.73**	-30
$1 \rightarrow 1.03$	88.22	77.78	82.27	70.60	-31
$1 \rightarrow 1.04$	87.86	77.66	82.09	70.35	-32
$1 \rightarrow 1.05$	86.83	77.83	81.76	69.84	-34

3.1.1 *The training stage*

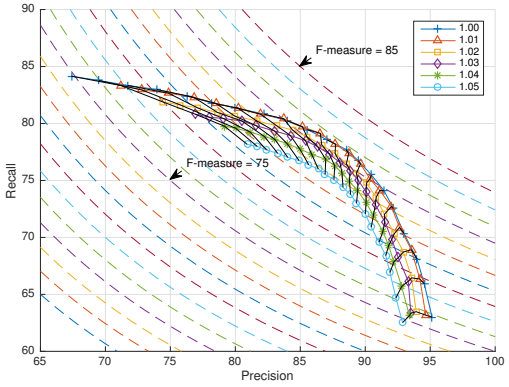
The training stage includes two rounds. In the first round, we first fix note activations ($\mathbf{H}$) for each isolated note according to the ground truth, then update all other parameters ($\mathbf{W^a}, \mathbf{W^d}, \mathbf{P}$ and α). The transient patterns are normalised to maximum of 1 after each iteration. In theory, the transient patterns follow a certain shape and could be shared by all pitches. So we use the average of the trained transient patterns to reduce the number of parameters and to avoid potential overfitting. The trained transient patterns and the average transient pattern are shown in Figure 5. In the second round, we fix the note activations ($\mathbf{H}$) and the transient pattern ($\mathbf{P}$), then update all other parameters ($\mathbf{W^a}, \mathbf{W^d}$ and α).

3.1.2 *Transcription results*

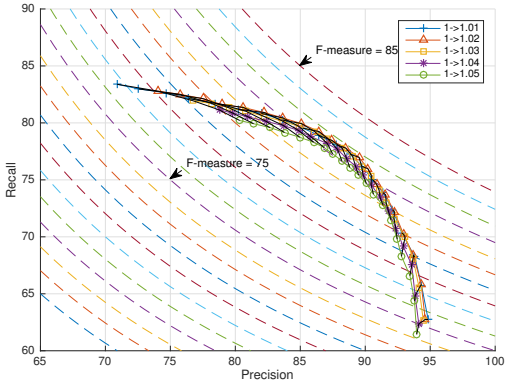
For transcription, we update note activations $\mathbf{H}$, keeping parameters ($\mathbf{W^a}, \mathbf{W^d}, \mathbf{P}$ and α) fixed from the training stage. Table 1 shows note tracking results (presented as percentage) using different sparsity factors. The optimal thresholds are shown in the last column. The top part of Table 1 are results using fixed sparsity factors. The best results are achieved without the sparsity constraint (γ = 1.00), with an F-measure of 82.24%. The performance decreases with increasing sparsity factor. The second part of the experiment gives results for using annealing sparsity. The best F-measure is 82.36% with the setting ($1.00 \rightarrow 1.02$). The difference between the best and the worst F-measure is only 0.6 percentage points. In general, all results with different sparsity constraints are considerably good with optimal thresholds, and the optimal threshold decreases when sparsity gets higher. However, F-measures considering both onsets and offsets are quite low, around 40%.

In the proposed model, the activation of each note decays after its onset, as shown in the decay activations of Figure 1. Given a note is played, we consider two situations. In the first case, there is another note of the same pitch being played later. We know that in this case the first note should be ended then. If the activation of the first note has already decreased to a low level, there is little influence on detecting the second note. However, if these two notes are very close, detection of the second note might be missed because of the remaining activation of the first note. In the second case, there is another note of a different pitch being played. The activation of the first note won't be changed by the attack of the second note in our model, while for standard NMF, there is always some interference with the first note's activation.

We compute the performance using thresholds ranging from -40 to -21dB to study performance variations as a function of the threshold. Figure 6(a) shows the results for different fixed sparsity factors. It is clear that precision decreases with the increase of the threshold, while recall increases. The higher the sparsity factor is, the more robust the results are on threshold changes. This is because small peaks in activations are already discounted when imposing sparsity, as shown in Figure 7. Lowering the threshold does not bring many false positives. Results with higher sparsity are less sensitive to the decrease of the threshold. However, when the threshold becomes larger, the results with low sparsity still outperform those with high sparsity. With a larger threshold, the number of true positives decreases. There are more peaks in activations when using lower sparsity, so more true positives remain. This favours the assumption that the true positives have larger amplitudes. Figure 6(b) shows the robustness of using annealing sparsity factors. The transcription results are close to each other. With annealing sparsity, the results are better and more tolerant to threshold changes.



(a) Results with fixed sparsity factors



(b) Results with annealing sparsity factors

Figure 6: Performance (presented percentage) using different sparsity factors and thresholds. The sparsity factors are indicated by different shapes, as shown in the top-right box. Lines connecting different shapes are results achieved via the same threshold. The threshold of the top set is -40dB, and the bottom set is -21dB. The dashed lines show F-measure contours, with the values dropping from top-right to bottom-left.

3.2 Comparison with state-of-the-art methods

We apply a comparison experiment on three datasets, pieces from a real piano ('ENSTDkCl') and a synthetic piano ('AkPnCGdD') in the MAPS database [17], and another 10 synthetic piano pieces (denoted as 'PianoE') used in [12]. All experiments are performed on the first 30s of each piece. We compare to two top transcription methods. Vincent *et al.*'s method applies adaptive spectral bases generated by linear combinations of narrow-band spectra, so the spectral bases have a harmonic structure and the flexibility to adapt to different sounds [20]. Benetos and Weyde's method employs 3 templates per pitch, and the sequence of templates is constrained by a probabilistic model [21]. In the PianoE dataset, we also compare to another state-of-the art method of Ewert *et al.* [12]. This method identifies frames in NMD patterns with states in a dynamical system. Note events are detected with constant amplitudes but various durations. In the comparison experiment, the proposed system is also trained on isolated notes from the AkPnCGdD and PianoE pianos. Vincent *et al.*'s method is performed in an unsupervised way, to indicate what can be achieved without training datasets. We use the version of Benetos and Weyde's method from the MIREX competition [22]. We have access to the code and train this model on isolated notes of the corresponding pianos. For Ewert's method we only have access to the published data in [12]. These two methods are performed in a supervised way.

Based on previous analysis, we employ the following parameters for the proposed model in comparison experiments. The sparsity factor is $\gamma = 1 \rightarrow 1.04$ by balancing among note tracking results and the robustness to different thresholds. Onsets are detected with threshold $\delta = -30$dB. In the first dataset ('ENSTDkCl'), results of other methods are also reported with optimal thresholds with best note-wise F-measures. Then the same thresholds are used for two synthetic piano datasets.

Results on piano pieces from the 'ENSTDkCl' subset are shown in Table 2(a). The proposed model has a note tracking F-measure of 81.80% and a frame-wise F-measure of 79.01%, outperforming Vincent *et al.*'s unsupervised method by around 10 and 20 percentage points,

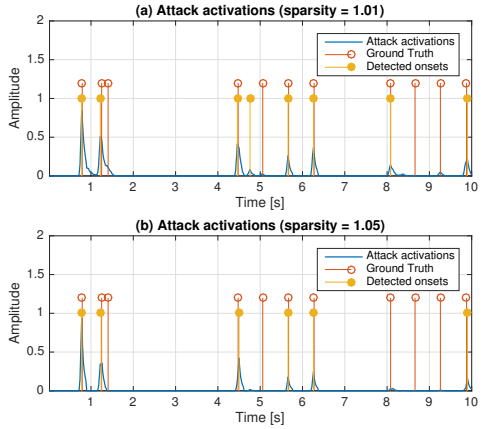


Figure 7: Detected onsets with different sparsity for pitch G4 (MIDI index 67).

respectively. Results of Benetos and Weyde's method are in between.

Results on the synthetic piano 'AkPnCGdD' are shown in Table 2(b). In general, all methods perform better on this dataset than on the 'ENSTDkCl' dataset, especially on note tracking results. The proposed model has the best results (84.63% on note tracking F-measure and 80.81% on frame-wise F-measure), outperforming all other methods by at least 5 percentage points.

Results on the other synthetic dataset 'PianoE' are shown in Table 2(c). Note tracking results of all methods are good but frame-wise results are poor. Ewert *et al.*'s method performs the best on note tracking (88% on F-measure), and Benetos and Weyde's method is the second (83.80% on F-measure). The proposed model only outperforms Vincent *et al.*'s method, with F-measures of 81.28% and 79.41% for these two methods respectively. However, the proposed model remains the best on the frame-wise F-measure (66.77%). Pieces in this dataset are from a piano competition. Many notes have very short durations. The remaining energies of a short note in the proposed model may interfere with later notes, causing false negatives.

A supervised neural network model also works on the MAPS database for piano transcription [23]. Besides an acoustic model, the method employs a music language model to capture the temporal structure of music. Although the method is not directly comparable, it is noticeable that our method exceeds its results by at least 5 percentage points on F-measures. When tested on the real recordings using templates trained in the synthetic piano notes, the proposed method has both F-measures of around 65%, outperforming the method of [23] by 10 percentage points on note-wise F-measure in a similar experiment.

4. DISCUSSION AND CONCLUSION

In this paper we propose a piano-specific transcription system. We model a piano note as a percussive attack stage and a harmonic decay stage, and the decay stage is explicitly modelled as an exponential decay. Parameters are learned in a sparse NMF, and transcription is performed in a supervised way. The proposed model provides promising transcription results, with around 82% and 79% for note tracking and frame-wise F-measures in music pieces from a real piano in the 'ENSTDkCl' dataset. The annealing sparsity factor improves both performance and the robustness of the proposed model. The comparison experiment shows that the proposed model outperforms two state-of-the-art methods by a large margin on real and synthetic pianos in the MAPS database. On a different synthetic dataset, the other methods performs relatively better, especially on note tracking, while the proposed method remains best on frame-wise metrics.

The proposed model can also be understood as a deconvolution method in which a patch is parameterised by two sets of templates and activations. One advantage of the proposed model is that we can build a note-level system by deconvolution, which has provided good transcription results [9, 10, 12]. The other is that using parametric patches reduces the number of parameters. The model also provides us with a way to analyse piano decay rates.

In the future, we would like to represent a note's decay stage by a decay filter instead of a decay rate, which is more in line with studies on piano decay [13]. Secondly, the good performance on piano music transcription is partly due to the availability of the training datasets. We would like to build an adaptive model, which could work in a more general scenario, hence more automatically. Finally, we are keen to find a way to estimate note offsets more accurately in the proposed model.

Table 2: The comparison experiment

(a) Transcription results on 'ENSTDkCl'

Method	F_{on}	A_{on}	F_f	A_f
Decay	**81.80**	**69.94**	**79.01**	**65.89**
Vincent [20]	72.15	57.45	58.84	42.71
Benetos [21]	73.61	59.73	67.79	52.15

(b) Transcription results on 'AkPnCGdD'

Method	F_{on}	A_{on}	F_f	A_f
Decay	**84.63**	**74.03**	**80.81**	**68.39**
Vincent [20]	79.86	67.32	69.76	55.17
Benetos [21]	74.05	59.57	53.94	38.65

(c) Transcription results on 'PianoE'

Method	F_{on}	A_{on}	F_f	A_f
Decay	81.28	69.12	**66.77**	**51.63**
Vincent [20]	79.41	66.39	58.59	42.45
Benetos [21]	83.80	**72.82**	60.69	44.24
Ewert [12]	**88**	-	-	-

5. ACKNOWLEDGEMENT

TC is supported by a China Scholarship Council/Queen Mary Joint PhD Scholarship. EB is supported by a Royal Academy of Engineering Research Fellowship (grant no. RF/128). We would like to thank Dr. Sebastian Ewert for his comments on this paper.

6. REFERENCES

[1] P. Smaragdis and J. C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Proc. IEEE WASPAA*, pages 177–180, 2003.

[2] A. Cont. Realtime multiple pitch observation using sparse non-negative constraints. In *Proc. ISMIR*, pages 206–211, 2006.

[3] T. Virtanen. Monaural sound source separation by non-negative matrix factorization with temporal continuity and sparseness criteria. *IEEE Trans. on Audio, Speech and Language Processing*, 15(3):1066–1074, 2007.

[4] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Trans. on Audio, Speech and Language Processing*, 18(3):538–549, 2010.

[5] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434, 2013.

[6] F. Rigaud, A. Falaize, B. David, and L. Daudet. Does inharmonicity improve an NMF-based piano transcription model? In *Proc. ICASSP*, pages 11–15, 2013.

[7] E. Benetos and S. Dixon. Multiple-instrument polyphonic music transcription using a temporally constrained shift-invariant model. *The Journal of the Acoustical Society of America*, 133(3):1727–1741, 2013.

[8] T. Cheng, S. Dixon, and M. Mauch. Improving piano note tracking by HMM smoothing. In *Proc. EUSIPCO*, pages 2009–2013, 2015.

[9] Z. Chen, G. Grindlay, and D. Ellis. Transcribing multi-instrument polyphonic music with transformed eigeninstrument whole-note templates. In *MIREX*, 2012.

[10] T. Berg-Kirkpatrick, J. Andreas, and D. Klein. Unsupervised transcription of piano music. In *Advances in Neural Information Processing Systems 27*, pages 1538–1546. 2014.

[11] A. Cogliati and Z. Duan. Piano music transcription modeling note temporal evolution. In *Proc. ICASSP*, pages 429–433, 2015.

[12] S. Ewert, M. D. Plumbley, and M. Sandler. A dynamic programming variant of non-negative matrix deconvolution for the transcription of struck string instruments. In *Proc. ICASSP*, pages 569–573, 2015.

[13] T. Cheng, S. Dixon, and M. Mauch. Modelling the decay of piano sounds. In *Proc. ICASSP*, pages 594–598, 2015.

[14] G. Weinreich. Coupled piano strings. *The Journal of the Acoustical Society of America*, 62(6):1474–1484, 1977.

[15] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Proc. Advances in Neural Information Processing Systems 13*, pages 556–562, 2000.

[16] T. Cheng, S. Dixon, and M. Mauch. A deterministic annealing EM algorithm for automatic music transcription. In *Proc. ISMIR*, pages 475–480, 2013.

[17] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Trans. on Audio, Speech, and Language Processing*, 18(6):1643–1654, 2010.

[18] S. Dixon. On the computer recognition of solo piano music. In *Proc. Australasian Computer Music Conference*, pages 31–37, 2000.

[19] M. Bay, A. F. Ehmann, and J. S. Downie. Evaluation of multiple-F0 estimation and tracking systems. In *Proc. ISMIR*, pages 315–320, 2009.

[20] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Trans. on Audio, Speech and Language Processing*, 18(3):528–537, 2010.

[21] E. Benetos and T. Weyde. An efficient temporally-constrained probabilistic model for multiple-instrument music transcription. In *Proc. ISMIR*, pages 701–707, 2015.

[22] E. Benetos and T. Weyde. Multiple-F0 estimation and note tracking for MIREX 2015 using a sound state-based spectrogram factorization model. In *MIREX*, 2015.

[23] S. Sigtia, E. Benetos, and S. Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939, 2016.

AUTOMATIC DRUM TRANSCRIPTION USING BI-DIRECTIONAL RECURRENT NEURAL NETWORKS

Carl Southall, Ryan Stables, Jason Hockman
Digital Media Technology Laboratory (DMT Lab)
Birmingham City University
Birmingham, United Kingdom
{carl.southall, ryan.stables, jason.hockman} @bcu.ac.uk

ABSTRACT

Automatic drum transcription (ADT) systems attempt to generate a symbolic music notation for percussive instruments in audio recordings. Neural networks have already been shown to perform well in fields related to ADT such as source separation and onset detection due to their utilisation of time-series data in classification. We propose the use of neural networks for ADT in order to exploit their ability to capture a complex configuration of features associated with individual or combined drum classes. In this paper we present a bi-directional recurrent neural network for offline detection of percussive onsets from specified drum classes and a recurrent neural network suitable for online operation. In both systems, a separate network is trained to identify onsets for each drum class under observation—that is, kick drum, snare drum, hi-hats, and combinations thereof. We perform four evaluations utilising the IDMT-SMT-Drums and ENST *minus one* datasets, which cover solo percussion and polyphonic audio respectively. The results demonstrate the effectiveness of the presented methods for solo percussion and a capacity for identifying snare drums, which are historically the most difficult drum class to detect.

1. INTRODUCTION

Within the field of music information retrieval, automatic music transcription systems seek to produce a symbolic notation for the instruments in an audio recording. There are a variety of areas in the educational, analytical and creative industries that would benefit from high quality music transcription. To date, the majority of such systems focus on transcription of pitched instruments, with relatively few systems intended for the extraction of drum notation. Automatic drum transcription (ADT) is useful in determining the rhythm and groove inherent in recordings consisting of either drum solos or polyphonic instrument mixtures. While high classification accuracies have been demonstrated for isolated drum hits [9], the task of classification becomes more difficult when multiple different drum instrument hits occur at the same time [10], and is further complicated when other instrumentation is introduced creating a polyphonic mixture.

© Carl Southall, Ryan Stables, Jason Hockman. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Carl Southall, Ryan Stables, Jason Hockman. "Automatic Drum Transcription using Bi-directional Recurrent Neural Networks", 17th International Society for Music Information Retrieval Conference, 2016.

1.1 Background

Using the categorisation presented in [7], the majority of previous ADT systems can be understood as either *segment and classify*, *match and adapt*, or *separate and detect*. Segment and classify methods [2, 6, 17] first divide recordings into regions using either onset detection or a metrical grid derived from beat tracking; second, extract features from the segments; and third perform classification to determine the drum instruments in the segments. Match and adapt methods [21, 22] first associate instruments to predetermined templates then iteratively update the templates to reflect the spectral character of the recording. Separate and detect methods [5, 13, 14, 16] attempt to separate the music signal into the drum sources that make up the mixture prior to identifying the onsets of each source. To date, the most effective separate and detect method for ADT has been non-negative matrix factorisation (NMF), an algorithm that divides a recording into a number of basis functions and corresponding time variant gains. Systems have been proposed for both offline and online applications. Dittmar and Gärtner [3] proposed three types of NMF—fixed, adaptive and semi-adaptive—which can be used in online situations taking each frame as its own NMF instance. For polyphonic audio, Wu and Lerch [20] used harmonic basis functions to separate the drums under observation from the mixtures and improved on standard NMF by introducing new iterative update methods.

In addition to the above methods, ADT systems have been proposed that do not fit in the above categorisation. Paulus incorporated hidden Markov models to identify the probability of drum events based on previous information [15]. Thompson used support vector machines (SVM) with a large dictionary of possible rhythmic configurations to classify automatically detected bars [18].

1.2 Motivation

With the exception of [15, 18] the majority of recent ADT systems rely on single basis functions for each instrument.

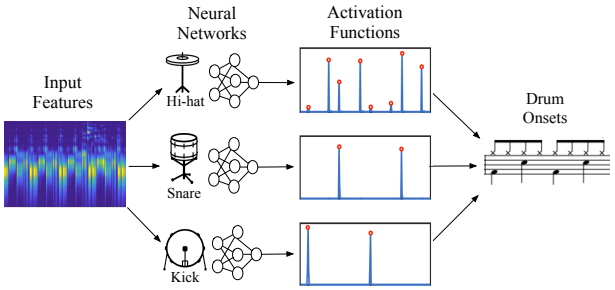


Figure 1: Overview of proposed method. Features are input to individual neural networks for each instrument, resulting in activation functions. Drum onsets are found by peak-picking the activation functions.

This has the potential to overfit to a specific playing technique associated with an individual instrument and fails to recognise more subtle usage. The instrument with the most varied playing techniques in the standard drum kit is the snare drum (e.g., flam, rolls, ghost notes), which not surprisingly is the most difficult to reliably detect. In addition, spectral overlap between basis functions may produce crosstalk between instruments such as snare drums and hi-hats, which can result in noisy time-variant gains, ultimately making peak picking more difficult.

Neural networks are capable of associating complex configurations of features with both individual or combined classes. They have also demonstrated excellent performance in fields related to ADT, such as source separation [8, 11, 12] and onset detection [1, 4]. Other supervised learning techniques such as SVMs have been incorporated in ADT systems [14, 18, 19], however neural networks are capable of capturing the association of class labels with time-series data and producing clean activation functions for the subsequent peak-picking stage. We therefore propose to extend the use of neural networks to ADT in order to exploit their well-known prowess for class separability and their ability to capture the variety of playing techniques associated with each instrument class under observation.

The remainder of this paper is structured as follows: Section 2 outlines our proposed methods for ADT. The evaluation and results are outlined in Section 3 and Section 4 presents a discussion of these results. Conclusions and possible future work are provided in Section 5.

2. METHOD

An overview of our proposed method for ADT is presented in Figure 1. For each percussive instrument under observation, features obtained from the audio recording are input into the pre-trained neural networks iteratively by frame. We then select the peaks from the resulting activation functions to determine the location of onsets for the corresponding instruments.

2.1 Neural Networks

Recurrent neural networks (RNN) incorporate information from previous time steps that allow for temporal information to be understood. Bi-directional neural networks (BDRNN) include information from future time steps by combining two RNNs: the first is a standard backwards directional RNN that incorporates present and previous time information; the second RNN is instead positioned to incorporate information from present and future time positions, achieved by reversing the order of the input time steps. As BDRNNs are unsuitable for online applications, we propose two separate models: an RNN for online usage and a BDRNN for applications that can operate offline. An overview of both neural networks is given in Figure 2.

2.1.1 Recurrent Neural Network

The RNN architecture is represented in Figure 2 by the solid lines. For an RNN with L layers, the equation for each layer l is:

$$a_0^l(t) = f_l(a_0^{l-1}(t)W_0^l + \beta(a_0^l(t-1)U_0^l) + b_0^l), \quad (1)$$

where $\beta = 0$ for $l = L$, and 1 otherwise. With layer l output a, the weight matrices W and U and the bias matrices b. The transfer function is determined by the layer, and is defined as:

$$f_l(x) = \begin{cases} 2/(1+e^{-2x}) - 1, & l \neq L \\ y = e^x/(\sum e^x), & l = L. \end{cases} \quad (2)$$

2.1.2 Bi-directional Recurrent Neural Network

The additional BDRNN connections are represented by dashed lines in Figure 2. For a BDRNN with L layers, the equation for each hidden layer l is:

$$a_n^l(t) = f_l(a_n^{l-1}(t)W_n^l + a_n^l(t-1)U_n^l + a_{(1-n)}^{l-1}Z_{(1-n)}^{l-1} + b_n^l) \quad (3)$$

where the layer is defined as forward directional when $n = 0$ and backwards directional when $n = 1$. Z is an additional weight matrix. The output layer for time t can then be defined as:

$$a_0^L(t) = f_L(a_0^{L-1}(t)W_0^L + a_1^{L-1}(t)W_1^L + b^L) \quad (4)$$

2.2 Input Features

Following the approach in [20], input audio (mono .wav files sampled at 44.1 kHz with 16-bit resolution) is transformed into a 1024 x n spectrogram representation using the short-time Fourier transform (STFT), in which n is the numbers of frames. The STFT is calculated using a Hanning window with a window length of 2048 samples and a hop size of 512 samples.

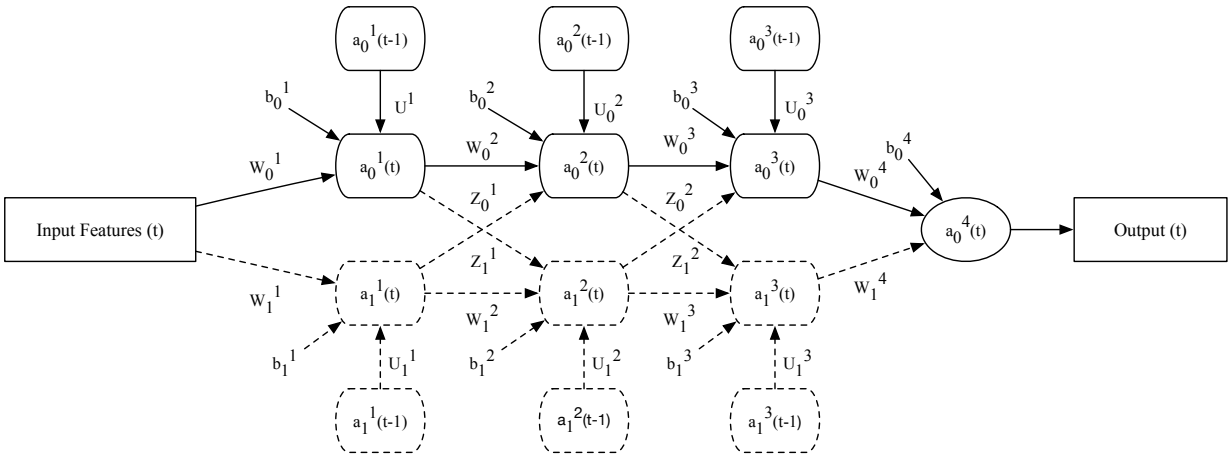


Figure 2: Overview of the proposed bi-directional recurrent neural network (BDRNN) and recurrent neural network (RNN). Solid lines represent the RNN connections and dashed lines are additional BDRNN connections. Tan sigmoid layers are shown as curved rectangles and soft max layers are represented by circles. The weight matrices are denoted as W, U and Z and the biases as b. The output of layer two of the backwards directional recurrent neural network is denoted by a_1^2.

2.3 Architecture

The neural network architectures in each instrument are identical, consisting of three dense hidden layers of 50 neurons. This configuration was chosen as it achieved the highest results in preliminary testing. The neural networks are trained using target activation function representations created from training data annotations. The first row of the activation function frames in which onsets occur are set to one and all other frames are set to zero. The networks are trained with a learning rate of 0.05 using truncated back propagation through time, which updates the weights and biases iteratively using the output errors. A maximum iteration limit is set to 1000, and the weights and biases are initialised to random non-zero values between ± 1 ensuring that training commences correctly. To prevent overfitting, a validation set is created from 10% of the training data. If no improvement is demonstrated on the validation set over 100 iterations, then training is stopped. The performance measure used is cross entropy combined with a softmax output layer, as this proved to be the most effective configuration.

2.4 Onset Detection

Once a drum activation function θ has been generated for each drum class under observation, the onsets must be temporally located from within θ. We adopt the method from [4] for onset detection in the BDRNN. To calculate onset positions, a threshold is first determined using the mean of all frames and a constant λ:

$$T = mean(\theta) * \lambda. \tag{5}$$

If the current frame n is determined to be both a peak and above the threshold T then it is accepted as an onset Γ:

$$\Gamma(n) = \begin{cases} 1, & \theta(n-1) < \theta(n) \geq \theta(n+1) \;\&\; T < \theta(n) \\ 0, & otherwise. \end{cases} \tag{6}$$

For online applications using the RNN, where future information can not be used within the peak picking process, the threshold is determined by taking the mean of the current frame and the previous ρ frames with an onset being accepted if the current frame is greater than the threshold and the previous frame. We selected $\rho = 9$ after initial informal testing. Due to the iterative classification of each frame, onsets may be detected in adjacent frames. We therefore disregard onsets detected within 50 ms of each other to ensure false positives are not obtained for a drum event that has already been detected.

3. EVALUATION

We conduct four evaluations intended to test the presented systems in a variety of different contexts in which an ADT system could be used. The first evaluation, termed *automatic*, aims to demonstrate system performance on drum solo recordings in a general purpose way where no prior information about the test track is known. Following [3], the second evaluation allows information from the test tracks to be used to aid in transcription of drum solo recordings in a *semi-automatic* manner. This scenario could be used for compositional or educational purposes either for identifying an arrangement of a specified drum solo that has been resequenced in another recording, or in a studio situation in which a single drum kit is being used. The third and fourth evaluations aim to evaluate the systems in polyphonic mixtures, where instruments other than the drums under observation are found. Mixtures containing other drums (e.g., floor toms, ride cymbal) are used in the third evaluation and additional harmonic accompaniment (e.g., guitars, keyboards) is found in the fourth. These evalua-

	Precision				Recall				F-measure			
	kick	*snare*	*hi-hat*	*mean*	*kick*	*snare*	*hi-hat*	*mean*	*kick*	*snare*	*hi-hat*	*mean*
BDRNN	0.912	0.834	0.795	**0.847**	0.929	**0.901**	0.729	0.853	0.909	**0.852**	0.738	**0.833**
RNN	0.890	**0.856**	0.741	0.829	0.909	0.851	**0.788**	0.849	0.884	0.833	0.729	0.816
PFNMF	0.934	0.633	**0.939**	0.835	0.931	0.889	0.743	0.854	0.926	0.699	**0.811**	0.812
AM1	0.934	0.633	**0.939**	0.835	0.931	0.889	0.743	0.854	0.926	0.699	**0.811**	0.812
AM2	**0.937**	0.651	0.893	0.827	**0.934**	0.886	0.786	**0.868**	**0.929**	0.713	0.805	0.816

Table 1: Precision, recall and F-measure results for the automatic evaluation using the IDMT-SMT-Drums dataset, including the PFNMF, AM1 and AM2 systems and the proposed BDRNN and RNN systems. The highest accuracy achieved in each of the categories is highlighted bold.

tions are termed *percussive mixtures* and *multi-instrument mixtures* respectively.

3.1 Evaluation Methodology

Standard precision, recall, and F-measure scores are used to measure system performance. Precision and recall are determined from detected drum instrument onsets, with candidate onsets determined as correct if found within 50 ms of annotations. Only kick drum, snare drum and hi-hat onsets are taken into consideration. The mean F-measure is calculated by taking the average of the individual instrument F-measures. We set the λ parameter used in neural network peak picking using grid-search across the dataset.

3.1.1 Automatic Evaluation

To test the generalisation of the proposed system, we undertake the automatic evaluation, using the IDMT-SMT-Drums dataset [3]. This dataset consists of 95 tracks (14 *real drum* tracks, 11 *techno drum* tracks, and 70 *wave drum* tracks) with individual kick drum, snare drum and hi-hat recordings. The average track length is 15 seconds, and in total there are 3471 onsets. Using three-fold cross validation the dataset is split into training and testing data, resulting in approximately 89,000 and 37,000 frames, respectively. Mean precision, recall and F-measure scores are taken across tested folds for each system under evaluation. The proposed neural network systems are evaluated alongside the three methods proposed in [20]: PFNMF, which uses fixed percussive basis functions in conjunction with harmonic basis functions within a NMF framework; AM1, which iteratively updates the percussive basis functions of PFNMF; and AM2, which updates the PFNMF basis functions and activation functions in an alternating fashion. Each of the NMF systems are initialised by taking the mean of each of the basis functions derived from the individual tracks.

3.1.2 Semi-automatic Evaluation

In order to test the systems ability to adapt to a specific situation, we undertake the semi-automatic evaluation. We again utilise the IDMT-SMT-Drums dataset, however in this context we provide the systems exclusively with individual drum hits that are used in the overall track under analysis. For a performance comparison in this evaluation, we also test the worth of training the neural networks using mixed drum hits (e.g., kick drum and hi-hat played together). The proposed methods are evaluated alongside the semi-adaptive online NMF technique CD as presented in [3]. As the evaluation procedures herein are identical to those in [3] the results from this work have been incorporated for comparison.

3.1.3 Percussion and Multi-instrument Evaluations

To test how well the proposed system can identify drums within various types of mixtures, we perform the percussion and multi-instrument evaluations, using the same procedure as in the automatic evaluation. For these evaluations, we use the ENST *minus one* dataset as it contains drum tracks with additional drum instruments (e.g., floor tom, ride cymbal) and techniques (e.g., ghost notes, flams, rolls) as well as accompaniment tracks. The ENST minus one dataset contains 64 recordings performed by three drummers; two drummers performed 21 tracks each and the third drummer performed 22 tracks. The BDRNN and RNN are provided recordings of two of the drummers as training, while testing on the third. The average track length is 55 seconds with a total of 22,410 kick drum, snare drum and hi-hat onsets, resulting in 210,000 and 105,000 frames for training and testing respectively in each fold. We mix the accompaniment and drum recordings in the dataset using the same ratios ($\frac{1}{3}$ and $\frac{2}{3}$, respectively) as in [7, 15, 20]. The evaluation procedures in these two evaluations are identical to those in [7, 15, 20], and as such the results from these studies have been used for comparison herein.

Method	*Mean F-measure*
RNN (individual drums)	0.634
BDRNN (individual drums)	0.700
RNN (mixed drums)	0.955
BDRNN (mixed drums)	**0.961**
CD	0.950

Table 2: Mean F-measure results for the semi-automatic evaluation. BDRNN and RNN systems are trained on individual drum hits (individual drums) or mixtures of drum hits (mixed drums) and are compared with that of the CD method in [3].

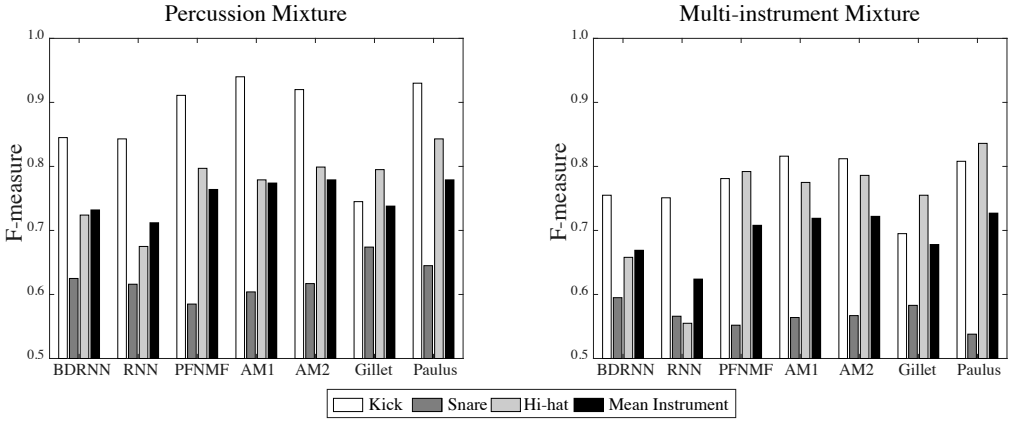


Figure 3: Kick drum, snare drum, hi-hat and mean instrument F-measure results for the BDRNN and RNN. Results for *percussion mixture* (left) and *multi-instrument mixture* (right) evaluation scenarios are compared to those obtained in [7, 15, 20].

3.2 Results

3.2.1 Automatic Results

The proposed BDRNN achieved a higher mean instrument F-measure than existing ADT methods in the first evaluation, which focuses on drum solos. Table 1 demonstrates that the neural network approaches achieved the highest scores in six of the twelve categories, with the largest relative improvement being the snare F-measure and precision. As expected, the RNN achieved lower F-measure scores than the BDRNN for all instruments, however the results matched those of the other evaluated systems. The three methods proposed in [20] achieved similar results as previously obtained on the IDMT-SMT-Drums dataset. Initial tests revealed that the best performance for all three algorithms was achieved with the rank parameter set to 1 and an offset coefficient of 0.2. AM1 showed no improvement on PFNMF in this instance, however AM2 did slightly improve.

3.2.2 Semi-Automatic Results

Table 2 shows the results of the semi-automatic evaluation scenario with both systems compared to the results of the CD system obtained in [3]. Training the neural networks using individual drum hits alone resulted in low accuracies, however when training includes mixed drum instrument signals (e.g., kick drums and hi-hats playing at the same time) both the BDRNN and RNN achieve the highest results of the tested systems.

3.2.3 Percussion and Multi-Instrument Mixture Results

Figure 3 shows the results of the BDRNN and RNN methods as compared to those achieved by the Gillet [7] and Paulus [15] systems, as well as the results of the PFNMF, AM1, and AM2 systems in [20]. The results are shown for both scenarios: percussive mixtures (left figure) and multi-instrument mixtures (right figure). In both evaluations, the neural network approaches achieve high snare F-measures relative to the other systems, and the BDRNN achieves the highest snare F-measure for the multi-instrument mixture evaluation. Figure 4 shows the mean precision and recall scores of the neural network systems in comparison to the other evaluated systems. The highest recall scores are achieved by the BDRNN and RNN for both percussion and multi-instrument mixtures. While the neural networks achieved lower mean F-measure scores, this high recall demonstrates the potential worth of the clean activation functions.

4. DISCUSSION

The results show that the proposed neural network systems achieve higher results for a solo drum dataset in offline and online situations in both automatic and and semi-automatic evaluations. The offline bi-directional recurrent neural network architecture outperformed the online recurrent neural network architecture in all evaluations, demonstrating the worth of additional future information for applications that allow it. The high results for the snare drum class achieved throughout the evaluation indicate the ability of the neural networks to associate multiple different frequency bases to the same class making them well suited to detect a variety

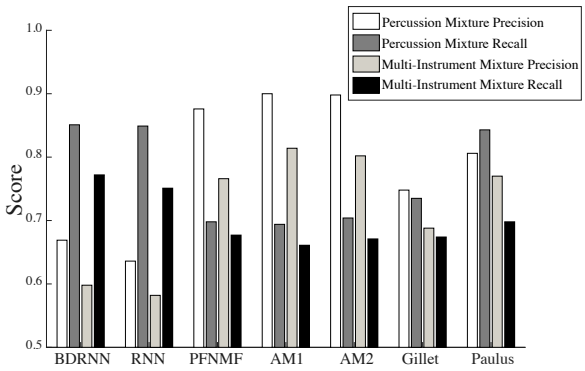


Figure 4: Mean Precision and Recall results from the percussion and multi-instrument mixture evaluations.

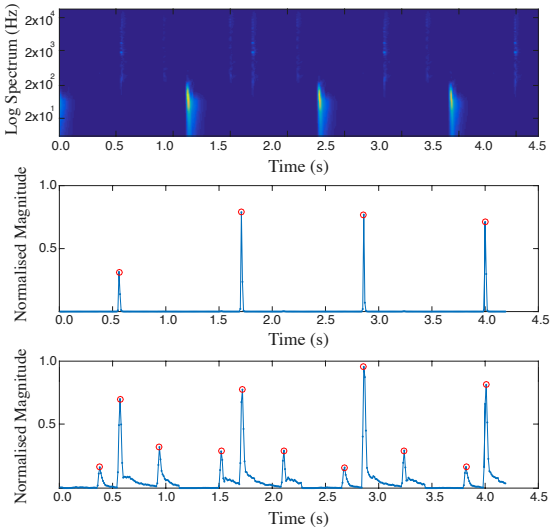


Figure 5: Comparison between neural network activation function and PFNMF time-variant gain: (*top*) Mixed spectrogram containing kick drum, snare drum and hi-hat; (*middle*) BDRNN snare drum activation function and found onsets; (*bottom*) PFNMF snare drum time-variant gain and found onsets.

of playing techniques for a given instrument (e.g., flams, rolls, ghost notes). An example of the instrument-specific focus achievable by neural networks is shown in Figure 5, where spectral overlap exists between the snare drum and hi-hats. While the PFNMF output for this particular example shows the effect of crosstalk, the BDRNN is able to achieve a less noisy activation function.

Although high F-measures for the kick drum and hi-hat were achieved by both the RNN and BDRNN methods, they had lower scores than other techniques for all situations other than the semi-automatic drum transcription test. The precision scores for the kick drum and hi-hat were the main factor in the lower F-measure score. As shown in Table 1 and Figure 4, the BDRNN and the RNN achieved the highest mean recall scores for all tests using the ENST *minus one* dataset, which indicates that the methods benefit from a simplified peak picking process due to clean activation functions. However, F-measures scores for these tests indicate that the BDRNN and RNN were not as successful as other systems—a somewhat expected result as this dataset contains polyphonic mixtures. The addition of a pre-processing stage similar to [20] could remove these sources prior to ADT and potentially improve results for the BDRNN and RNN methods. Another area for possible improvement would be to evaluate the worth of different input features such as MFCCs, which have already been demonstrated to be successful in conjunction with neural networks in the related task of onset detection [1].

5. CONCLUSIONS AND FUTURE WORK

We have presented two neural network based approaches for ADT: the BDRNN for off-line usage and the RNN online applications. Results from the conducted evaluations demonstrate that the proposed methods are capable of outperforming existing ADT systems on drum solo recordings in both automatic and semi-automatic situations. The ability to learn a rich representation of drum classes enables the neural networks to detect multiple playing techniques within the same class. Evaluations were also carried out on polyphonic mixtures in which the neural network achieved high snare F-measures relative to existing approaches. To improve performance of the proposed methods for polyphonic audio, an additional pre-processing source separation stage could be introduced into the system to separate the desired drums from additional instrumentation prior to ADT. Furthermore, additional time step connections to additional previous and future time steps may potentially increase the accuracy of the system. One method for doing this is by using long short-term memory cells within the neural network architecture which have already proven to be effective for onset detection [4]. Further evaluation will be carried out to determine performance when additional drum classes are present, as well as testing other input features.

6. ACKNOWLEDGEMENTS

The authors would like to thank Chih-Wei Wu and Christian Dittmar for their help in providing code for implementations of their methods, as well as the reviewers for their valuable feedback in this paper.

7. REFERENCES

[1] S. Böck, A. Arzt, F. Krebs, and M. Schedl. Online real-time onset detection with recurrent neural networks. In *Proc. of the 15th International Conference on Digital Audio Effects (DAFx-12)*, 2012.

[2] C. Dittmar. Drum detection from polyphonic audio via detailed analysis of the time frequency domain. In *Proc. of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2005.

[3] C. Dittmar and D. Gärtner. Real-time transcription and separation of drum recordings based on nmf decomposition. In *Proc. of the 17th International Conference on Digital Audio Effects (DAFX)*, 2014.

[4] F. Eyben, S. Böck, B. Schuller, and A. Graves. Universal onset detection with bidirectional long short-term memory neural networks. In *Proc. of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 589–594, 2010.

[5] D. Fitzgerald. *Automatic drum transcription and source separation*. PhD thesis, Dublin Institute of Technology, 2004.

[6] O. Gillet and G. Richard. Automatic transcription of drum loops. In *Proc. of the 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 269–272, 2004.

[7] O. Gillet and G. Richard. Transcription and separation of drum signals from polyphonic music. *IEEE Transactions on Audio, Speech and Language Processing*, 16(3):529–540, 2008.

[8] E. M. Grais, M. U. Sen, and H. Erdogan. Deep neural networks for single channel source separation. In *Proc. of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3734–3738, 2014.

[9] P. Herrera, A. Yeterian, and F. Gouyon. Automatic classification of drum sounds: A comparison of feature selection methods and classification techniques. In *Proc. of the International Conference on Music and Artificial Intelligence*, pages 69–80, 2002.

[10] P. Herrera-Boyer, G. Peeters, and S. Dubnov. Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(1):3–21, 2003.

[11] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis. Deep learning for monaural speech separation. In *Proc. of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1562–1566, 2014.

[12] J. Le Roux, J. R. Hershey, and F. Weninger. Deep NMF for speech separation. In *Proc. of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 66–70, 2015.

[13] H. Lindsay-Smith, S. McDonald, and M. Sandler. Drumkit transcription via convolutive NMF. In *Proc. of the 15th International Conference on Digital Audio Effects (DAFx)*, 2012.

[14] A. Moreau and A. Flexer. Drum transcription in polyphonic music using non-negative matrix factorization. pages 353–354, 2007.

[15] J. Paulus and A. Klapuri. Drum sound detection in polyphonic music with hidden Markov models. *EURASIP Journal on Audio, Speech, and Music Processing*, 2009(1):1–9, 2009.

[16] J. Paulus and T. Virtanen. Drum transcription with non-negative spectrogram factorization. In *Proc. of the 13th European Signal Processing Conference (EUSIPCO)*, pages 1–4, 2005.

[17] K. Tanghe, S. Degroeve, and B. De Baets. An algorithm for detecting and labeling drum events in polyphonic music. pages 11–15, 2005.

[18] L. Thompson, M. Mauch, and S. Dixon. Drum transcription via classification of bar-level rhythmic patterns. In *Proc. of the International Conference on Music Information Retrieval (ISMIR)*, pages 187–192, 2014.

[19] D. van Steelant and K. Tanghe. Classification of percussive sounds using support vector machines. In *Proc. of the 2004 Machine Learning Conference of Belgium and The Netherlands*, pages 146–152, 2004.

[20] C.-W. Wu and A. Lerch. Drum transcription using partially fixed non-negative matrix factorization with template adaptation. In *Proc. of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 257–263, 2015.

[21] K. Yoshii, M. Goto, and H. G. Okuno. Automatic drum sound description for eeal-world music using template adaptation and matching methods. In *Proc. of the 5th International Conference on Music Information Retrieval*, pages 184–191, 2004.

[22] K. Yoshii, M. Goto, and H. G. Okuno. Drum sound recognition for polyphonic audio signals by adaptation and matching of spectrogram templates with harmonic structure suppression. *IEEE Transactions on Audio, Speech and Language Processing*, 15(1):333–345, 2007.

AUTOMATIC PRACTICE LOGGING: INTRODUCTION, DATASET & PRELIMINARY STUDY

R. Michael Winters, Siddharth Gururani, Alexander Lerch
Georgia Tech Center for Music Technology (GTCMT)
{mikewinters, siddgururani, alexander.lerch}@gatech.edu

ABSTRACT

Musicians spend countless hours practicing their instruments. To document and organize this time, musicians commonly use practice charts to log their practice. However, manual techniques require time, dedication, and experience to master, are prone to fallacy and omission, and ultimately can not describe the subtle variations in each repetition. This paper presents an alternative: by analyzing and classifying the audio recorded while practicing, logging could occur automatically, with levels of detail, accuracy, and ease that would not be possible otherwise. Towards this goal, we introduce the problem of Automatic Practice Logging (APL), including a discussion of the benefits and unique challenges it raises. We then describe a new dataset of over 600 annotated recordings of solo piano practice, which can be used to design and evaluate APL systems. After framing our approach to the problem, we present an algorithm designed to align short segments of practice audio with reference recordings using pitch chroma and dynamic time warping.

1. INTRODUCTION

Practice is a widespread and indispensable activity that is required of all musicians who wish to improve [5]. While a musical performance progresses through a score in linear-time and with few note-errors, practice is characterized by repetitions, pauses, mistakes, various tempi, and fragmentation. It can also take a variety of forms, including technique, improvisation, repertoire work, and sight-reading. It can occur with any musical instrument (often with many simultaneously), and can take place in a range of acoustic environments.

Within this context, we present the problem of Automatic Practice Logging (APL), which attempts to identify and characterize the content of musical practice from recorded audio during practice. For a given practice session, an APL system would output exactly what was practiced at all points in time, and describe how practice occurred. [1]

By its nature, an APL system must be robust to wrong notes, pauses, repetitions, fragmentation, dynamic tempi, and other typical "errors" of practice. It should be able to operate in challenging acoustic environments, work with any instrument, and even with ensembles. Most importantly, it needs to identify *what* is being practiced and characterize *how* practice is occurring, so that it can describe and transcribe its content for a user.

In the following paper we elaborate on the subject of automatic practice logging (APL), including its benefits and challenges. We present precursors and relevant methods that have been developed in the MIR community, and which frame APL as a viable area of application. We then introduce a publicly available dataset of 34 hours of annotated piano practice including a typology for practice that informed our annotation. We conclude with a description of a preliminary algorithm capable of identifying the piece that is being practiced from short segments using pitch chroma and dynamic time warping.

2. MOTIVATION

At all skill levels, practice is key to learning music, advancing technique, and increasing expression [13]. Keeping track of the time spent practicing, or "practice logging" is an important component of practice, with many uses and benefits. Logging practice is a complex endeavor. For example, a description of practice might include amount of time spent practicing, specific pieces or repertoire that were practiced, specific sections or measure numbers, approaches to practicing, and types of practicing (e.g. technique exercises, sight-reading, improvisation, other instruments, or ensemble work). An even greater level of detail would describe how a particular section was practiced, and even the many nuances involved in each repetition. For performers, an APL system can offer unprecedented levels of detail, ease, and accuracy, not to mention additional advantages of digitization. The output of an APL system could help musicians to structure and organize the time spent practicing, to provide insight into personal improvement, and to engage in good practice habits (e.g., deliberate, goal-oriented practice [13]). For teachers and supporters, practice logs provide a window into a musician's private practice, which may foster a better understanding of improvements (or lack

[1] E.g., "Chopin's Raindrop Prelude, Op. 28, No. 15, mm. 1–26 was practiced 11 times with a metronome gradually increasing tempo from 40–55 BPM. Mm. 19–26 were played slower on average and were characterized by fragmentation and pauses."

© R. Michael Winters, Siddharth Gururani, Alexander Lerch. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** R. Michael Winters, Siddharth Gururani, Alexander Lerch. "Automatic Practice Logging: Introduction, Dataset & Preliminary Study", 17th International Society for Music Information Retrieval Conference, 2016.

thereof), leading to more informed and thoughtful feedback. Researchers can benefit from detailed accounts of practice, gaining insights into performance and rehearsal strategies. For the field of Music Information Retrieval (MIR), APL offers a new and challenging area of application, which may culminate in valuable tools for researchers studying practice as well.

2.1 The Benefits of APL

Primary benefits of Automatic Practice Logging (APL) are increased levels of detail and ease of use. In repertoire practice, it is common for musicians to repeat sections of pieces many times, with the progression of these repetitions resulting in the musical development from error-ridden sight-reading to expressive performance. Marking and tallying these many repetitions manually would be impractical, and describing each repetition in terms of nuances (e.g., tempo changes, wrong/correct notes, expressive timing and intonation) would be even more so. However by using APL, repetitions could be identified and tallied automatically. Simply remembering to turn on the system and occasionally tagging audio could be the extent of user input. Once a section has been identified, a host of other MIR tools could be used to characterize and describe small variations in each repetition.

Another benefit of APL is accuracy. In addition to the relative dearth of detail that was mentioned previously, manual practice logging is plagued by the fallibility of human memory, resulting in omission and fallacy in logged practice [13]. Especially for students that are uncommitted to their instrument, manual logging may be prone to exaggeration and even deceit. By using the audio recorded directly from practice, an APL system could more accurately reflect the content of practice.

A host of other benefits of would arise due to the digitization of the information. Using a digital format could lead to faster sharing of practice with teachers, who might be able to comment on practice remotely and provide support in a more continuous manner. Practice descriptions could be combined with ancillary information such as the day of the week, location of the practice, local weather, mood, and time of day, and lend itself to visualization through graphs and other data displays, assisting in review and decision making. Over time, this information might be combined and used by an intelligent "practice companion" that can encourage effective practice behaviors.

2.2 APL Challenges

Automatic practice logging, however, is not easy and a successful system must overcome a variety of challenges that are unique to audio recorded during practice. While live performances and studio recordings are almost flawless—including few (if any) wrong notes and unfolding linearly with respect to the score—the same can not be said about practice. Instead, practice is error-laden, characterized by fragmentation, wrong notes, pauses, short repetitions, erratic jumps (even to completely different pieces), and slower, variable, and unsteady tempi. In polyphonic practice (e.g., a piano or ensemble), it is not uncommon to practice individual parts or hands separately.

Additional problems for APL arise from the fact that recordings made in a natural practice session will occur in an environment that is far from ideal. For example, metronomes, counting out-loud, humming, tapping, page-turning, and singing are common sound sources that do not arise directly from the instrument. Speech is also common in practice, and needs to be identified and removed from a search, but can also occur while the instrument is playing. Unlike recording studios and performance halls, practice environments are also subject to extraneous sound sources. These sources might include the sounds of other instruments and people, but also HVAC systems and a host of other environmental sounds. The microphone used to record practice might also be subject to bad practices such as poor placement, clipping, and sympathetic vibrations with the surface on which it was placed.

Last but not least, using APL for repertoire practice needs to address issues of audio-to-score alignment. Scores commonly include structural repetitions such as those marked explicit (e.g., repeat signs), and those occurring on a phrase level. At an even smaller time frame, it is not uncommon to have sequences of notes repeated in a row (e.g., ostinato), or short segments repeated at different parts of the piece (e.g., cadences). For a window that has many near-identical candidates in a given score, an APL system will have difficulties determining to which repeat the window belongs. This difficulty is compounded by the fact that practice is highly fragmented in time, so using longer time-frames for location cues may not be feasible.

3. RELATED WORK

Given the importance and prevalence of practice in the lives of musicians, the subject of practice has received considerable attention in the music research community [2, 13]. Important questions include the role of practice in attaining expertise [19], the effects of different types of practice [1,6], and the best strategies for effective practice [8, 11]. However, to the best knowledge of the authors, automatically recognizing and characterizing musical practice has not specifically been addressed in MIR. It draws important parallels with many application spaces, but also offers its own unique challenges (see Sect. 2.2).

Perhaps its closest neighbor is the task of cover song detection [17], which in turn might derive methods from audio-to-audio or audio-to-score alignment and audio similarity [10]. Another possible area of interest is automatic transcription [12], and piano transcription [15] in particular for the presented dataset. In this section, techniques of cover song detection are described and compared with the unique requirements for an APL system. The cover song detection problem may be formulated as the following: Given a set of reference tracks and test tracks, identify tracks in the test set that are cover songs of a reference track. Ellis and Poliner derive a chroma-per-beat matrix representation and cross-correlate the reference and query track's matrices

to search for sharp peaks in the correlation function that translate to a strong local alignment [7]. The chroma-per-beat helps with tempo-invariance and chroma-vectors can be circular shifted to handle transpositions. Ravuri and Ellis make use of similar features to train a Support Vector Machine (SVM) classifier that classifies a reference/test song pair as a reference/cover song pair [16]. Serra et al. propose to extract harmonic pitch-class profile (HPCP) features from the reference and query track [18]. Dynamic Time Warping (DTW) is then used to compute the cost of alignment between the reference HPCP and query HPCP features. The DTW cost is representative of the degree to which a track is a cover of another. A system for large-scale cover-song detection is presented by Bertin-Mahieux and Ellis [4] as a modification of a landmark-based fingerprinting system [20]. The landmarks in this cover-song detection algorithm are pitch chroma-based instead of frequency-based as in the original fingerprinting algorithm. This makes the hashing key-invariant because it is possible to circular-shift the query chroma while searching for a match.

By analogy to cover song detection, repertoire practice consists of fragments of the practiced piece that should be independently identified as belonging to a particular track. Identifying the start and end times of a particular segment computationally is non-trivial, but must be the basis of a subsequence search algorithm (e.g., [9]). The subsequence search algorithm must furthermore be robust against practice artifacts such as pauses, various tempi, missed notes, short repetitions, and sporadic jumps. The cover-song detection methods described above take care of tempo invariance and algorithms for APL may leverage this for robustness against varying tempi.

Commercial products exist that focus on music practice and education, such as: SmartMusic, [2] Rocksmith [3] and Yousician. [4] SmartMusic is a music education software that enables teachers to enter lessons, track their students' progress and give feedback. Students also have access to pieces in the SmartMusic library. Rocksmith is an educational video game for guitar and bass that interfaces with a real instrument and helps users learn to play by choosing songs and exercises of a skill level that increases as a user progresses through the game. Yousician is a mobile application that teaches users how to play guitar, bass, ukulele and piano. It also employs tutorials to help users progress. In APL, the exercises are not predefined and an APL system should be able to detect and log a user's practice session without knowing what exercise or repertoire was practiced beforehand, making it less intrusive and more flexible.

4. THE APL DATASET

4.1 Considerations

Apart from the issues related to the recorded audio discussed in Sect. 2.2, APL needs to accommodate the many forms that practice might take. Although repertoire practice using scores is common in the western art-music tradition, practice might also incorporate technique exercises, sight-reading, improvisation, and ensemble practice.

Bearing this framework in mind, the annotations for the dataset were informed by a typology of musical practice that frames the problem of APL in terms of two fundamental questions:

1. What type of practice occurred?
2. What was practiced?

The first question refers to the many types of practice that can occur, while the second question pertains to the actual content of practice. For a given type of practice (e.g., repertoire practice), question two can be addressed using two descriptors: *what* piece was practiced and *where* in the piece practice occurred.

To answer the first question, we organize the types of practice based upon the following basic categories: *technique*, *repertoire practice*, *sight-reading*, *improvisation*, and *ensemble work*.

'Technique' refers to the numerous fundamental repetitive patterns (e.g., scales and arpeggios) a performer would undertake. These have a pedagogical purpose, and typically involve involve basic musical elements, but would also include advanced technical and mental exercises like transposition and polymeters. 'Repertoire practice' refers to the repetitive practice of specific pieces of music for long-term musical goals such as public concerts and recordings. These repertoire pieces should be distinguishable from musical pieces that were practiced for a comparatively short amount of time (e.g., once or twice before moving on), which were labeled as 'sight-reading.' Although improvisation might be used as a type of technique or mental exercise, we choose to list it as a separate category given its importance in entire genres of music that is based only loosely upon a score if at all. The last category, 'ensemble work,' is meant to reflect the fact that the experience of practicing music is often shared by other performers, with their own unique instruments. However, it should be mentioned that the other items in this typology could be repeated in the ensemble work category.

4.2 Description

To begin working towards an APL system, we created a dataset of 34 hours of recorded piano practice including detailed annotations of the type of practice that was occurring, and the piece that was being played. These 34 hours of practice were chosen from a larger set of 250 hours of recordings made by one performer over the course of a year. They were targeted because they included repertoire practice that occurred in preparation for a studio-recording of a particular multi-movement piano piece: *Prokofiev's Piano Sonata No. 4 in C-minor, Op. 29*.

Recordings were made using a H4N Zoom recorder on a variety of Baby-Grand pianos in partially sound-isolated practice rooms. On each day of the recording, the microphone was placed upon the music rack of the piano, facing

[2] http://www.smartmusic.com, Date accessed: May 24, 2016.
[3] http://rocksmith.ubi.com/rocksmith/, Date accessed: May 23, 2016.
[4] https://get.yousician.com/, Date accessed: May 23, 2016.

the harp of the piano. The microphone input gain was adjusted to a level that was maximized to prevent clipping and adjusted only marginally if and only if clipping was discovered. To automatically remove silence from the recordings, an automatic recording process was used that triggered the start of a recording with signal level above a threshold SPL value. Similarly, recordings were automatically stopped when the SPL fell below a threshold, and stayed below the threshold for four seconds. This process created some tracks which were "empty" due to a false trigger. These were removed from the dataset. All recordings were made using the built-in stereo microphones. Recordings were made at 44.1kHz sampling rate and used the H4Ns built-in 96kbps MP3 encoder.

4.3 Annotation

Using this method of automatic recording, between 10 and 60 sound-files were recorded each day depending upon the length of practice, which ranged from approximately 30 minutes to 3 hours. The pieces were annotated by the performer, who by nature was the most familiar with the work and could identify and annotate their practice with the greatest speed and accuracy. The performer annotated them using Sennheiser CX 300 II earbuds at a comfortable listening volume in one to two hour-long chunks. Using VLC's "short forward/back jump hot-key", the performer made annotation of the piece being practiced in 10-second intervals. For each segment, the performer listened to enough audio to identify the piece being played and then skipped to the next section. In this way, if there were any changes in piece during a track, they could be identified efficiently.

Annotations were made on an online spreadsheet and exported to CSV and TSV format. The columns of the spreadsheet were titled as follows:

1. Track Name
2. Type of Practice
3. Descriptor #1 (e.g., Composer)
4. Descriptor #2 (e.g., Piece)
5. Start & End Time (if applicable)
6. Other (e.g., metronome, humming, distortion)

The track names were the auto-generated track names generated by the recorder, which include the date of the recording and the recording number. The type of practice was labeled as either repertoire, sight-reading, technique, or improvisation. The third category was used to list the composer for repertoire and sight-reading, or, for technique, was used to provide a general type (e.g., arpeggios, scales). For improvisation, this category and the next were not used. For repertoire and sight-reading, the next category was used to label the piece being played (e.g., Op. 29, Mvt. 1). For sight-reading, labeling this column was challenging as some pieces that had been played only once could not be identified by ear anymore.

Table 1. Number of files and length for major items in the APL dataset.

	# of Tracks	# of Minutes
Op. 29, Mvt. 1	74	115
Op. 29, Mvt. 2	50	61
Op. 29, Mvt. 3	215	250
Other Repertoire	94	80
Sight-Reading	45	73
Technique	106	177
Improvisation	14	17

The start and end times were used for cases when the track needed to be broken up due to the presence of other practice. In repertoire practice, this might occur when the performer suddenly switched pieces or movements without the necessary amount of silence to trigger a new recording. For these cases, a new annotation was created using the same track name as the original, but with different labels for composer and piece, and different start and end times. If the piece was kept constant throughout the track, the start and end times were not used. Last, the 'Other' category was used to provide annotations of atypical sounds that occurred such as humming, tapping, metronome use and practice of individual parts in an otherwise polyphonic texture. It was also used to denote tracks of special interest, such as when a score was played through without fragmentation as in a performance.

Table 1 presents the number of files and amount of time for major components of the dataset. The dataset, including the annotations and recordings have been made publicly available on Archiv.org. [5] In the future, efforts will be directed towards extending the annotation scheme to accommodate more exact score-locations (e.g., measure numbers), adding a third question to the previous two: *How did practice occur?* Updated annotations will be kept with a version controlled repository. The database will also be expanded to include more instruments, and types of practice. Limiting factors to the growth are the creation of annotations, which require time and attention to annotate in detail. Those wishing to contribute to the database may contact the first author.

5. PRELIMINARY STUDY

5.1 Problem Formulation

As discussed in Sect. 4, we separate the APL task for repertoire practice into two primary components: 1) recognition of *which* repertoire piece is being practiced, and 2) recognition of *where* in the piece the practice is occurring. The former gives a general insight into the content of practice while the latter provides a more detailed view on the evolution of practice within the piece itself. Currently, we focus on the first component and present an algorithm that determines a matching reference track for each frame of the query track.

[5] https://archive.org/details/Automatic_Practice_Logging, Date accessed, May 23, 2016.

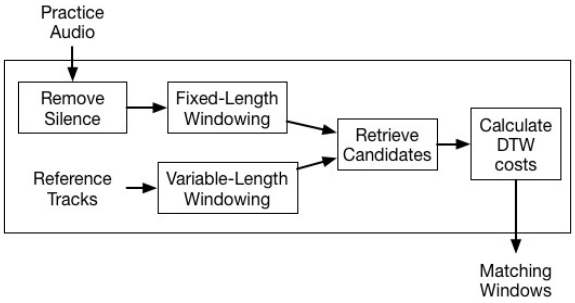


Figure 1. Block diagram of the presented system.

5.2 Overview

Although the task of automatically identifying practice audio is difficult, we present a simple approach that handles some of the major challenges of APL: pauses, fragmentation, and variable, unsteady tempi in the recorded audio.

A block diagram of the algorithm is provided in Fig. 1. We begin with a library of "reference tracks" that are full-length recordings of the repertoire being practiced. These reference tracks can, for example, be a commercial CD recording or a full recording of the student or teacher's performance. After blocking these tracks, we compute a 12-dimensional pitch chroma vector per block. The pitch chroma captures the octave-independent pitch content of the block mapped across the 12 pitch classes [3]. We aggregate multiple pitch chromas by averaging them over larger texture windows with pre-defined lengths. Windows containing silence are dropped. The results of this computation are then one chroma vector per window, resulting in multiple chroma matrices for each of the reference tracks and window lengths.

Incoming query tracks are processed similarly. For each query texture window, a distance to all reference windows is calculated in order to select the candidates with the least distance. Subsequently, we compute the DTW cost between the selected reference texture window and the query texture window using the original (not aggregated) pitch chroma blocks. The DTW cost is the overall cost of warping the subsequence pitch chroma matrix from the query texture window to the reference pitch chroma matrix [14]. The reference track with the least DTW cost is chosen as the match for the query window.

5.3 Feature Extraction

The pitch chroma is extracted in blocks of length 4096 samples (app. 93 ms) with 50% overlap. The pitch chromas are then averaged into texture windows of 16 times the block length, with 7/8 overlap between neighboring windows for the query audio. As a preprocessing step, silences are ignored. Windows containing more than 50% samples with magnitude less than a threshold are dropped and labeled as zero windows. The remaining windows are labeled non-zero windows and are used for search. The feature extraction for the reference tracks is identical, however, multiple texture window lengths are used in order to account for different possible tempi. More specifically, lengths ranging from $N = 8, 10, 12, 14, 16, 18$ times the block size are used. Note that the length distribution is biased towards shorter windows as the query audio is more likely to be played slower than the reference. At the end of this step, we have an aggregated pitch chroma vector for the query audio and a set of aggregated pitch-chroma matrices for the reference tracks.

5.4 Candidate Track Selection

A match between query and reference is likely if the aggregated query pitch chroma matches one of the aggregated reference pitch chromas. We select a group of 15 likely track candidates for each reference track by computing the Euclidean distance between the query vector and all reference track vectors. At the end of this step, we have a pool of 15 candidates across all window lengths across for each of the reference tracks, making 45 matches total.

5.5 Track Identification

For the last step, we step back to the original short-time pitch chroma sequence. This means that our query track and reference tracks are now represented as a matrix of dimension $12 \times (2N-1)$, where $N = 16$ for the query track and $N = \{8, 10, 12, 14, 16, 18\}$ for the reference tracks. The DTW cost is then computed for all 45 pairs of query matrix and reference matrices. For all pairs, the reference track with the texture window that has the lowest DTW cost relative to its path length and reference window size is chosen as the repertoire piece being practiced in that particular texture window of the query audio. Additional information such as the matching texture window length and matching frame are available, but not analyzed presently.

Using this sequence of steps, texture windows in the reference library will be chosen for each query texture window. These windows correspond to particular locations in the reference tracks, while the window sizes correspond to the best matching tempo. Figure 2 presents the results of running this algorithm on all of the non-zero windows one track of practiced audio, plotting the detected windows over the practiced windows. The correct track is plotted as asterisks.

6. RESULTS

To test our approach on a large body of practice audio, we ran our algorithm on 50,000 windows of practice from the APL dataset. As our approach is targeted towards repertoire practice, we chose recordings from a piece the performer was working towards at that time, namely Prokofiev's *Piano Soanta No. 4 in C-Minor, Op. 29*. The piece is a three-movement work including sections of various tempi, note-densities, tonal strengths and key centers, and at various levels of completion and familiarity.

To create a roughly even distribution of query windows across the three reference tracks, particular days in the APL dataset were chosen for analysis. The APL dataset includes

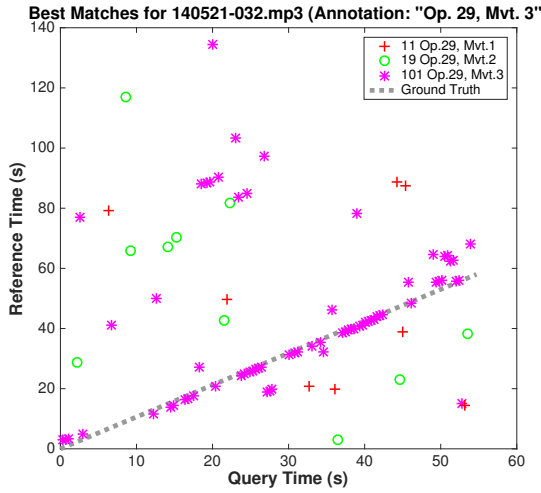


Figure 2. Detected piece across three reference tracks (see legend) and detected time in piece for all non-zero windows of a 60s query track of repertoire practice.

Table 2. Confusion matrix for the 50,000 windows belonging to either Mvt. 1, 2, or 3.

	Mvt. 1	**Mvt. 2**	**Mvt. 3**
Mvt. 1	0.55	0.29	0.15
Mvt. 2	0.17	0.66	0.17
Mvt. 3	0.13	0.21	0.66

a disproportionate amount of work on the third movement, so days were selected that included relatively more work on the first and second movements. These were May 5th, 7th, 11th, 14th, 15th, 21st and 22nd, 2014. Tracks annotated as 'technique,' 'sight-reading,' or 'improvisation' were not included. Furthermore, tracks that included annotations in the 'Other' category were not included as this category was used to indicate tracks with audio sources not from the instrument (e.g., metronome, humming, singing, counting, but also distortion). Last, tracks that included more than one piece being practiced, or more than one kind of practice were not included. A confusion matrix displaying the results of this test are displayed in Table 2.

7. DISCUSSION

The results demonstrate that an APL system based upon the pitch chroma of short windows of practice audio can be used to identify the piece being practiced. The results have targeted a broad level of description, specifically the correct identification of the piece being practiced. However, further levels of detail are provided by this approach: namely a specific location in the reference track, the window size corresponding to the match, and the amount of dissimilarity (cost) for that combination.

Although the present results are far from perfect, it is important to remember that APL by nature identifies audio that is error-laden. Pauses, short-repetitions, wrong-notes and general fragmentation make correct identification of every window a hard challenge. Instead, it is more practical for APL to use some form of monotonicity constraint. In the example of the present algorithm, a single window that is identified as Op. 29, Mvt. 2 that is surrounded by windows that are classified as belonging to a particular section in Op. 29, Mvt. 1, likely belongs to Mvt. 1. One could also favor windows that are in a sequence in the reference tracks, or have the same window length (same relative tempo). It is interesting to note that for the present results, a simple majority vote for non-zero windows across each query track could be used to remove chosen candidates from minority identifications and replace them with candidates from the majority identification. Even this course interpolation would lead to dramatic improvements in the confusion matrix of Table 2.

It is also necessary to acknowledge the importance of reference tracks in APL. In the present case, we make use of "full versions" of the repertoire pieces played by the same performer in a similar recording environment as the practiced audio. However, in general, complete versions of repertoire pieces are not available until the performer has already practiced them significantly. Although one could choose to use studio recordings as reference, recording and production artifacts like microphone placement, SNR, spectral and temporal effects and reverberation may leave traces in the feature vector that can make correct identification more difficult. Furthermore, each performer and performance is subject to subtle timing deviations, which may create a systematic deviation when trying to match with those of the user. An alternative might be to use audio from a reference MIDI score, which would provide the highest amount of control and the additional benefit of measure numbers for matches. Generating "reference" material from the performer themselves however remains an interesting prospect for APL, which might have the most use when a score is not available (e.g., improvisation, new music).

8. CONCLUSION

This paper has presented current efforts towards Automatic Practice Logging (APL) including an annotated dataset, and a preliminary approach to identification. Practice is a ubiquitous component of music, and despite challenges, there are many benefits to logging its content automatically. Practice occurs in many forms, and for the purpose of annotating it, we presented a typology and annotation framework that can be generalized to many instruments, musicians and types of practice. We presented a preliminary approach that searches a reference library using pitch-chroma computed on very short segments, and uses dynamic-time warping as an additional step to find the best match from a collection of candidates. Incorporating additional local assumptions such as score-continuity and constant tempo might lead to increased performance in the future, but one should be mindful that practice is globally fragmented and variable in tempo. We hope that this work will encourage others to explore APL as an interesting and valuable topic for MIR.

9. REFERENCES

[1] N. Barry. The effects of different practice techniques upon technical accuracy and musicality in student instrumental music performance. *Research Perspectives in Music Education*, 44(1):4–8, 1990.

[2] N. Barry and S. Hallam. Practice. In R. Parncutt and G. E. McPherson, editors, *The Science and Psychology of Music Performance: Creative Strategies for Teaching and Learning*, pages 151–66. Oxford University Press, New York, NY, 2001.

[3] M. A. Bartsch and G. H. Wakefield. To catch a chorus: Using chroma-based representations for audio thumbnailing. In *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, pages 15–18, New Paltz, NY, October 2001.

[4] T. Bertin-Mahieux and D. P. W. Ellis. Large-scale cover song recognition using hashed chroma landmarks. In *Proceedings of the 2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 117–20, New Paltz, NY, October 2011.

[5] R. Chaffin and A. F. Lemieux. General perspectives on achieving musical excellence. In A. Williamson, editor, *Musical Excellence: Strategies and Techniques to Enhance Performance*, pages 19–40. Oxford University Press, New York, NY, 2004.

[6] J. E. Driskell, C. Copper, and A. Moran. Does mental practice enhance performance? *Journal of Applied Psychology*, 79(4):481–92, 1994.

[7] D. P. W. Ellis and G. E. Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. In *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1429–32, Honolulu, HI, 2007.

[8] K. A. Ericsson, R. T. Krampe, and C. Tesch-Rmer. The role of deliberate practice in the acquisition of expert performance. *Psychological Review*, 100(3):363–406, 1993.

[9] A. Guo and H. Siegelmann. Time-warped longest common subsequence algorithm for music retrieval. In *Proceedings of the 5th International Conference on Music Information Retrieval*, pages 258–61, Barcelona, Spain, 2004.

[10] N. Hu, R. B. Dannenberg, and George Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proceedings of the 2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 185–8, New Paltz, NY, 2003.

[11] H. Jørgensen. Strategies for individual practice. In Aaron Williamon, editor, *Musical Excellence: Strategies and Techniques to Enhance Performance*, pages 85–104. Oxford University Press, New York, NY, 2004.

[12] A. P. Klapuri. Automatic music transcription as we know it today. *Journal of New Music Research*, 33(3):269–82, 2004.

[13] A. C. Lehmann, J. A. Sloboda, and R. H. Woody. *Psychology for Musicians: Understanding and Acquiring the Skills*, chapter 4, pages 61–81. Oxford University Press, New York, NY, 2007.

[14] M. Müller. *Information Retrieval for Music and Motion*, chapter 4, pages 69–84. Springer, Berlin, Germany, 2007.

[15] C. Raphael. Automatic transcription of piano. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, pages 15–19, Paris, France, October 2002.

[16] S. Ravuri and D. P. W. Ellis. Cover song detection: from high scores to general classification. In *Proceedings of the 2010 IEEE International Conference on Acoustics Speech and Signal Processing*, pages 65–8, Dallas, TX, 2010.

[17] J. Serrà, E. Gómez, and P. Herrera. Audio cover song identification and similarity: Background, approaches, evaluation, and beyond. In Z. W. Raś and A. A. Wieczorkowska, editors, *Advances in Music Information Retrieval*, pages 307–32. Springer, Berlin, Germany, 2010.

[18] J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(6):1138–51, 2008.

[19] J. A. Sloboda, J. W. Davidson, M. J. A. Howe, and D. G. Moore. The role of practice in the development of performing musicians. *British Journal of Psychology*, 87(2):287–309, 1996.

[20] A. L.-C. Wang. An industrial strength audio search algorithm. In *Proceedings of the 4th International Conference on Music Information Retrieval*, pages 7–13, Baltimore, MD, 2003.

DATA-DRIVEN EXPLORATION OF MELODIC STRUCTURES IN HINDUSTANI MUSIC

Kaustuv Kanti Ganguli[1] **Sankalp Gulati**[2] **Xavier Serra**[2] **Preeti Rao**[1]
[1] Department of Electrical Engineering, Indian Institute of Technology Bombay, Mumbai, India
[2] Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
kaustuvkanti@ee.iitb.ac.in

ABSTRACT

Indian art music is quintessentially an improvisatory music form in which the line between 'fixed' and 'free' is extremely subtle. In a rāga performance, the melody is loosely constrained by the chosen composition but otherwise improvised in accordance with the rāga grammar. One of the melodic aspects that is governed by this grammar is the manner in which a melody evolves in time in the course of a performance. In this work, we aim to discover such implicit patterns or regularities present in the temporal evolution of vocal melodies of Hindustani music. We start by applying existing tools and techniques used in music information retrieval to a collection of concerts recordings of ālāp performances by renowned khayal vocal artists. We use svara-based and svara duration-based melodic features to study and quantify the manifestation of concepts such as vādi, samvādi, nyās and graha svara in the vocal performances. We show that the discovered patterns corroborate the musicological findings that describe the "unfolding" of a rāga in vocal performances of Hindustani music. The patterns discovered from the vocal melodies might help music students to learn improvisation and can complement the oral music pedagogy followed in this music tradition.

1. INTRODUCTION

Hindustani music is one of the two art music traditions of Indian art music [6], the other being Carnatic music [28]. Melodies in both these performance oriented music traditions are based on the framework of rāga [3]. Performance of a rāga in Indian art music (IAM) is primarily improvisatory in nature [26]. While some of these improvisations are based on a composed musical piece, Bandish, others are completely impromptu expositions of a rāga, Ālāp. Rāga acts as a grammar both in composition and in improvisation of melodies.

The rules of the rāga grammar are manifested at different time scales, at different levels of abstraction and demand a different degree of conformity. While some of the elements of the rāga grammar are explicit, others are implicit and may require years of musical training to grasp. A number of textbooks and musicological studies exist that describe different improvisatory aspects of melodies in IAM [1, 3, 4, 6, 10, 18, 26]. These works also attempt to uncover some of the implicit aspects of rāga grammar.

A majority of the studies mentioned above are musicological in nature. These typically involve either a thorough qualitative analysis of a handful of chosen musical excerpts or a compilation of expert domain knowledge. Though these studies often present interesting musical insights, there are several potential caveats in such works. Some of these caveats are summarized below:

- Small repertoire used in the studies challenge the generalizability of the proposed musical models
- Bias introduced due to the subjectivity in the analysis of musical excerpts
- Absence of concrete quantitative evidences supporting the arguments
- The kind of analysis that can be done (manually) is limited by human capabilities, limited memory (both short- and long-term)
- Difficulty in reproducibility of the results

In addition to the musicological works mentioned above, there are several studies that perform computational modeling of different melodic aspects in IAM. These studies address computational research tasks such as rāga recognition [5, 15], melodic similarity [11, 17, 20], discovery and search of melodic patterns [12, 16], segmentation of a musical piece [27] and identification of specific landmarks in melodies [14]. These approaches typically employ signal processing and machine learning methodologies to computationally model the relevant melodic aspect. These studies can provide a ground for developing tools and technologies needed to navigate and organize sizable audio collections of music, perform rāga-based search and retrieval from large audio archives and in several other pedagogical applications.

Several qualitative musicological works bring out new musical insights but are prone to criticism of not having supported their findings using a sizable corpus. Contrary to that, quantitative computational studies manage to scale to

© Kaustuv Kanti Ganguli, Sankalp Gulati, Xavier Serra, Preeti Rao. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Kaustuv Kanti Ganguli, Sankalp Gulati, Xavier Serra, Preeti Rao. "Data-driven exploration of melodic structures in Hindustani music", 17th International Society for Music Information Retrieval Conference, 2016.

sizable data sets, but fall short of discovering novel musical insights. In the majority of cases, computational studies attempt to automate a task that is well known and is fairly easy for a musician to perform. There have been some studies that try to combine these two types of methodologies of working and corroborate several concepts in musical theories using computational approaches. In Chinese opera music, [22] has performed a comparison of the singing styles of two Jingju schools where the author exploit the potential of MIR techniques for supporting and enhancing musicological descriptions. Autrim [1] (Automated Transcription for Indian Music) has used MIR tools for visualization of Hindustani vocal concerts that created a great impact on music appreciation and pedagogy in IAM. We find that such literation pertaining to melodic structures in Indian art music is scarce.

In this paper, we perform a data-driven exploration of several melodic aspects of Hindustani music. The main objective of this paper is to use existing tools, techniques and methodologies in the domain of music information retrieval to support and enhance qualitative and descriptive musicological analysis of Hindustani music. For this we select five melodic aspects which are well described in musicological texts and are implicitly understood by musicians. Using computational approaches on a music collection that comprises representative recordings of Hindustani music we aim to study these implicit structures and quantify different melodic aspects related with them. In addition to corroborating existing musicological works, our findings are useful in several pedagogical applications. Furthermore, the proposed methodology can be used for analyzing artist or gharana-specific [2] melodic characteristics.

2. HINDUSTANI MUSIC CONCEPTS

Bor [3] remarks that rāga, although referred to as a concept, really escapes such categories as concept, type, model, pattern etc. Meer [26] comments that technically a rāga is a musical entity in which the intonation of svaras, as well as their relative duration and order, is defined. A rāga is characterized by a set of melodic features that include a set of notes ('svara'), the ascending and descending melodic progression ('ārōhana-avrōhana'), and a set of characteristic phrases ('pakad'). There are three broad categories of segments in the melody: stable svara regions, transitory regions and pauses. While svaras comprise certain hierarchical sub-categories like nyās, graha, amsa, vādi and samvādi, the transitions can also be grouped into a set of melodic ornamentation ('alankar') like meend, andolan, kan, khatka etc. [26]. The third important melodic event is the pause. Pauses carry much information about the phrases; in fact, a musician's skill lies in the judicious use of the pause [10]. We shall next go over these three broad aspects.

Many authors [3, 26] refer to the importance of certain svaras in a rāga. From the phrase outline we may filter certain svaras which can be used as rest, sonant or predominant; yet the individual function and importance of the svaras should not be stressed [26]. Nyās svara is defined as the resting svara, also referred to as 'pleasant pause' [7] or 'landmark' [14] in a melody. Vādi and samvādi are best understood in relation with melodic elaboration or vistār ('barhat'). Over the course of a barhat, artists make a particular svara 'shine' or 'sound'. There is often a corresponding svara which sustains the main svara and has a perfect fifth relation with it. The subtle inner quality of a rāga certainly lies in the duration of each svara in the context of the phraseology of the rāga. A vādi, therefore, is a tone that comes to shine, i.e., it becomes attractive, conspicuous and bright [26]. Another tone in the same rāga may become outstanding that provides an answer to the former tone. This second tone is the samvādi and should have a fifth relationship with the first tone. This relationship is of great importance.

A rāga is brought out through certain phrases that are linked to each other and in which the svaras have their proper relative duration. This does not mean that the duration, the recurrence and the order of svaras are fixed in a rāga; they are fixed only within a context [26]. The svaras form a scale, which may be different in ascending (ārōhana) and descending (avrōhana) phrases, while every svara has a limited possibility of duration depending on the phrase in which it occurs. Furthermore, the local order in which the svaras are used is rather fixed. The totality of these musical characteristics can best be laid down in a set of phrases ('calana') which is a gestalt that is immediately recognizable to the expert. In a rāga some phrases are obligatory while others are optional [26]. The former constitute the core of the rāga whereas the latter are elaborations or improvised versions. Specific ornamentation can add to the distinctive quality of the rāga [10].

There is a prescribed way in which a 'khayal' performance develops. The least mixed variety of khayal is that where an ālāp is sung, followed by a full sthāyi (first stanza of the composition) in madhya (medium) or drut (fast) laya (tempo), then layakari (rhythmic improvisation) and finally tān (fast vocal improvisation). When the barhat reaches the higher (octave) Sa (root svara of Hindustani music), the antara (second stanza of the composition) is sung. If the composition is not preceded by an ālāp, the full development of the rāga is done through barhat. The composition is based on the general lines of the rāga, the development of the rāga is again based on the model of the composition [26].

There are four main sources of a pause in a melody, these include: (i) gaps due to unvoiced consonants in the lyric, (ii) short breath pauses taken by the musician when out of breath, (iii) medium pauses where the musician shifts to a different phrase, and (iv) long pauses where the accompanying instruments improvise. Musically meaningful or musician-intended melodic chunks are delimited only by (iii) and (iv) [9].

Though Hindustani music is often regarded as improvisatory, the improvisation is structured. On a broader level there is a well defined structure within the space of

[1] https://autrimncpa.wordpress.com/

[2] Refers to a lineage or school of thought.

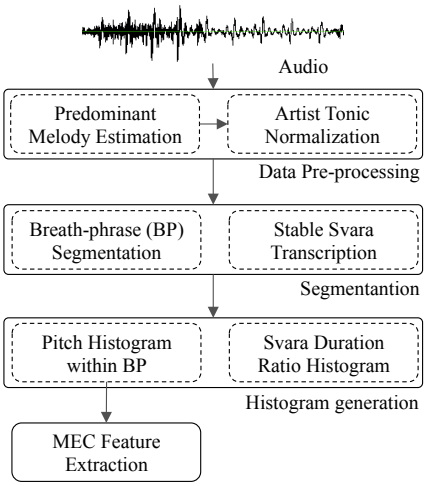


Figure 1. Block diagram for data processing

which an artist improvises following the rāga grammar. The overall skeleton of the melodies in Hindustani music can have defined structure at different levels. Some of these can be for the entire music tradition (for all rāgas and artists), some for specific rāgas, some of these can be gharana-specific, while some other artist specific. In this study, we aim to obtain overall structure in a melody and some rāga-specific patterns by the processing of audio concert recordings.

3. DATA PROCESSING

The block diagram for data processing is shown in Figure 1. It contains four main processing modules: preprocessing, segmentation, histogram generation and feature extraction. We describe these modules in detail in the subsequent sections.

3.1 Pre-processing

3.1.1 Predominant melody estimation

We start by estimating the pitch of the predominant melodic source in the audio signal and regard that as the representation of the melody. For predominant pitch estimation, we use the method proposed by Salamon and Gómez [23]. This method is reported to performed favorably in MIREX 2011 [3] on a variety of music genres, including IAM, and has been used in several other studies [8, 16]. We use the implementation of this algorithm as available in Essentia [2]. Essentia [4] is an open-source C++ library for audio analysis and content-based MIR. We use the default values of the parameters, except for the frame and hop sizes, which are set to 46 and 4.44 ms, respectively.

3.1.2 Tonic normalization

The base frequency chosen for a melody in a performance of IAM is the tonic pitch of the lead artist [13]. All other accompanying instruments are tuned with respect to this tonic pitch. Therefore, for analyzing features that are derived from the predominant melody across artists, the predominant melody should be normalized with respect to the tonic pitch. For this normalization we consider the tonic pitch ω as the reference frequency during the Hertz-to-cent-scale conversion, which is automatically identified using the multi-pitch approach proposed by Gulati et al. [13]. This approach is reported to obtain state-of-the-art results and has been successfully used elsewhere [12, 15].

3.2 Melodic segmentation

3.2.1 Breath-phrase segmentation

As described in Section 2 there are different types of unvoiced segments in the predominant melody. While some of these segments are musically a part of a melodic phrase (short-pauses), some others delineate consecutive melodic phrases. A distribution of the duration of all the unvoiced segments for the entire music collection revealed that their type can be identified based on the duration of the pause. For identifying intended breath pauses that separate melodic phrases we empirically set the duration threshold to be 500 ms. The duration of the intra-phrase pauses is considerably smaller than this threshold. All the intra-phrase breath pauses (i.e., with duration smaller than 500 ms) are interpolated using a cubic spline curve. We shall refer to a breath-phrase as BP hereafter.

3.2.2 Stable svara transcription

In Indian art music, written notation has a purely prescriptive role. Transcribing the melody of a musical performance into a written notation is a challenging task and requires an in-depth knowledge of the rāga [21, 29]. In this study we consider a simple melodic transcription that retains only the stable svara regions of a pitch contour and discards the transitory pitch regions. We first detect all the valid svaras and their precise frequencies used in a melody by computing a pitch histogram. Subsequently, we segment the stable svara regions by identifying the fragments of pitch contour that are within 35 cents [20] of the svara frequencies. Next, we filter out the svara fragments that are smaller than 250 ms in duration, as they are too short to be considered as perceptually meaningful held svaras [19]. This leaves a string of fragments each labeled by a svara. Fragments with the same svara value that are separated by gaps less than 100 ms are merged [12]. The resulting symbol sequence thus comprises a tuple of svara name and duration.

3.3 Histogram generation

3.3.1 Pitch histogram of breath-phrases

We compute the histogram of the transcribed svaras corresponding to each BP. Figure 2 shows the pitch histogram for each BP of the concert of rāga Todi sung by Ajoy Chakrabarty. . The 12th bin along the y-axis corresponds to the tonic svara Sa (0 cents), 24th for its octave (1200 cents).

[3] http://www.music-ir.org/mirex/wiki/2011
[4] https://github.com/MTG/essentia

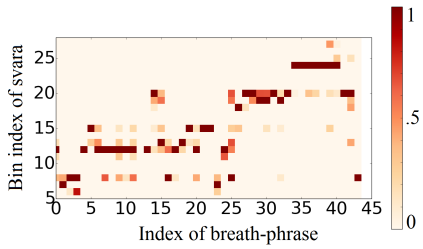


Figure 2. Pitch histogram of svaras for each breath-phrase.

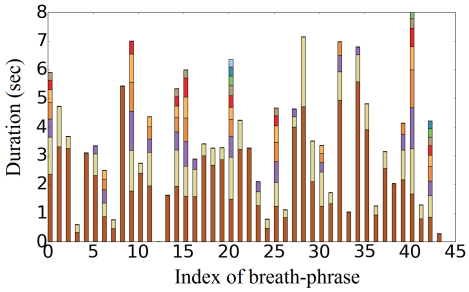


Figure 3. Bar graph of svara duration stacked in sorted manner for each breath-phrase. We observe that breath-phrases often comprise one long nyas svara and several other svaras of less duration.

3.3.2 Svara duration distribution

We consider the distribution of the svara duration for each BP. Figure 3 shows a stacked bar graph of the sorted durations of the svaras for each BP for the same concert. We observe that the cumulative duration of the transcribed svaras range from approximately 1 to 8 seconds. An important point to note here is that there is a difference between the absolute duration of a BP and the height of the stacked bar (in Figure 3). This is caused by the transient pitch segments that we ignored in our representation. Readers must note that the stable svara transcription, therefore, has an implication for the further analyses.

3.4 Post-processing

To capture the changes in the svara pattern at a broader time-scale, we time-average the pitch histogram across ten BPs with a hop of one BP. This is followed by tracking of the most salient bin across the smoothened histogram . Finally, the obtained contour is further median filtered with one preceding and succeeding BP. We refer to this contour as the *evolution contour* (hereafter EC). Figure 4 shows the time-averaged histogram superimposed with the EC for the same concert.

3.5 MEC feature extraction

We would like to compare the ECs of different concerts to explore whether there is any temporal trend that holds across the collection. To normalize the time-scale and pitch range of the EC, we normalize each EC within a unit

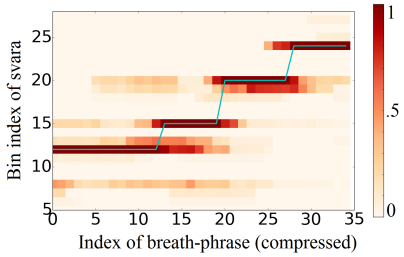


Figure 4. Time-averaged pitch histogram superimposed with the evolution contour.

range in both temporal and pitch dimensions. Thus a *modified evolution contour* (hereafter MEC) is obtained as:

$$MEC = \frac{EC - \min(EC)}{\max(EC) - \min(EC)} \quad (1)$$

with 100 samples between [0,1] .

We extract a collection of heuristic features (slope-based, duration-based, jump-based and level-based) from the MEC. A few important features are: slope between the MEC value of 0^{th} frame and the first frame where $MEC = 1$ (referred to as Slp) , proportion of duration spent on each svara (referred to as Pro), centroid (considering salience of the bins as the weights in the centroid computation) of each svara (referred to as Cen) , starting and ending svaras, (second) longest svara and proportion of its duration, magnitude of lowest/highest jumps between consecutive levels etc.

4. MUSIC COLLECTION AND ANNOTATIONS

The music collection used in this study is taken from the Hindustani music dataset (HMD) compiled as a part of the CompMusic project [24, 25] (130 hours of commercially available audio recordings stored as 160 kbps mp3 stereo audio files). All the editorial metadata for each audio recording is publicly available on an open-source metadata repository called MusicBrainz [5] . The selected music material in our collection is diverse in terms of the number of artists (40), recordings (mostly live concerts of both male and female renowned musicians from the last 6 decades) and the number of unique compositions (67). In these terms, it can therefore be regarded as a representative subset of real-world collections. Our collection includes a total of 75 concerts from 10 widely used rāgas (8 pieces per rāga on an average) that are diverse both in terms of the number of svaras and their pitch-classes (svarasthānās). All the concerts belong to either madhya or drut laya (and non-metered ālāp). The pitch range of the recordings spans approximately two octaves (middle octave and half of the lower/upper octave). All of the concerts comprise elaborations based on a bandish.

The scope of the study is limited to only the ālāp and vistār (barhat) [3, 4, 6, 18, 26] sections of the concert. Almost all of the concerts continue to subsequent fast improvisatory section (tān) after rendering the vistār. The

[5] https://musicbrainz.org/

melodic cue where the antara ends and the rhythmic cue where there is a slight increase in tempo just after, is quite universal and musically unambiguous. For reference, please observe Section 3.1 in [30]. We employ a performing Hindustani musician (trained over 20 years) to annotate the time-stamps where the vistār ends. As the said annotation can be considered an obvious one (there is a less chance of getting subjective biases), we limit the manual annotation to one subject only. After cutting the concerts to the musician-annotated time-stamp, the average duration per concert is 16 minutes making a total of 20 hours of data.

5. ANALYSIS AND DISCUSSION

We choose certain music concepts which are widely discussed among musicians and musicologists, for which there has not yet been an objective way of interpreting them from the audio. We cite the concepts (or knowledge-hypotheses, referred to as 'K') and discuss how a data-driven approach can help us validate them.

5.1 K1: Evolution of melody in time

As discussed in Section 2, the barhat of a rāga performance refers to the gradual "unfolding" of a rāga by building on the svaras with a progression in a broad time-scale. But it is not very clearly illustrated in the musicology literature what the precise duration is spent on each svara in course of this progression. Figure 5 shows the MECs of 37 (50% randomly chosen from our collection) concerts. We observe that the MECs, in general, start from a lower svara and gradually reach the highest svara in a step-like manner. The slope Slp of MEC, is quite consistent (mean = 1.3, standard deviation = 0.34) over the whole corpus. This gives an important insight that irrespective of the rāga and concert-duration, artists take the same time to explore the melody and hit the highest svara. This also reinforces the nature of the time-scaling of a performance: for either a short 20 minute- or a long 50 minute-concert, the melodic organization bases more on relative and not absolute time. We also observe a sharp fall of the MEC at the end of the many concerts, this reflects how artists come down to a lower svara to mark an end to the vistār (this coincides with the musician's annotation). This phenomenon has a high resemblance with the time evolution of melody in course of the vistār, as shown in Figure 11 in [30].

5.2 K2: Transitional characteristics of nyās svaras

Rāga guidelines mention about allowed svara sequences within a melodic phrase but it would be interesting to see if artists maintain any specific order in choosing the nyās svara across BPs or take liberty to emphasize any other svara. This is to be captured from the granularity of BPs and not in the time-averaged MEC. We generate a svara-transition matrix and populate it with a uniform weight for all transitions of the salient bins across BPs. Figure 6 shows the salient svara-transition matrix where the diagonal elements refer to self transitions. As indicative from

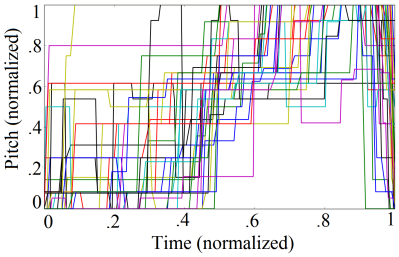


Figure 5. Modified evolution contours for 37 concerts in our music collection.

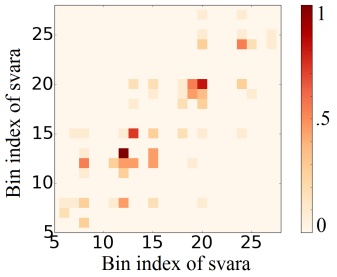


Figure 6. Svara-transition matrix of salient svaras of each breath-phrase. Intensity of each bin is proportional to the number of transitions taken from the svara of bin index on x-axis to the svara of bin index on y-axis.

wide steps of the MECs, there are quite a few self transitions but to our interest the salient transitions across BPs also follow a pattern alike the allowed svara-transitions within a melodic phrase. This is not a trivial event. We compute a feature to measure the steadiness quotient Stq of the transition matrix, defined as the ratio of the trace of the svara-transition matrix to the sum of all bins. We observe a very low standard deviation (0.23) across our music collection which conforms to the fact that artists 'focus' on a nyās svara for consecutive BPs to establish that svara.

5.3 K3: Relationship between functional roles of svaras and their duration in melody

We discussed about functional roles of vādi/samvādi svaras, but it is not explicitly known how their 'prominence' is defined. Earlier work [5] use histogram and show that they are one of the most used tonal pitches. But it is not evident from a pitch histogram whether the peak heights are contributed by a large number of 'short' svara segments or a fewer 'long' svara segments. Figure 7 shows the mean (left) and standard deviation (right) of all svaras (octave folded) for each svara along x-axis being the salient svara in a BP. We observe that the role of each svara is defined in terms of their duration in context of a nyās svara. This conforms with the concepts discussed in Section 2. This also reconfirms the well-defined structure of the melodic improvisation that any svara cannot be stretched arbitrarily long, the nyās svara of the BP and the phrase itself decides how much variance all other svaras can incorporate. This also brings out a question whether there is any special func-

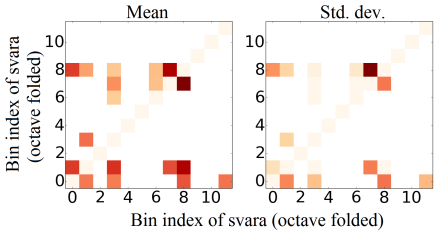


Figure 7. Mean (left) and standard deviation (right) of all svaras (octave folded) for each svara along x-axis being the salient svara in a breath-phrase.

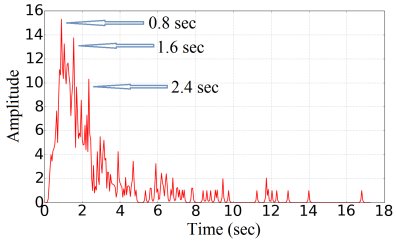


Figure 8. Ratio of inter-onset-interval of salient svaras across breath-phrases. We see a tatum pulse (peak) at 0.8 seconds and its harmonics.

tional role of vādi/samvādi svara in terms of their duration in a specific BP. One observation is that the vādi/samvādi svara, while a salient svara in the respective BPs, constrain the duration of all other svaras, making its relative prominence higher.

5.4 K4: Duration and position of svaras in melody

In music theory, vādi and samvādi svaras are among the concepts which are often discussed. But we do not have an objective measure to observe how these svaras are different from other svaras in a rāga. It is also interesting to know whether the vādi or samvādi svara always takes a focal role irrespective of their location in a BP and the overall position in a performance.

Position in melody: An important feature of the MEC is the Cen. We observe that the Cen is a rāga dependent feature. E.g., an uttaranga vādi rāga would have its vādi centroid in the latter half of a concert. This is supportive of the fact that the vādi is explored in due course of melodic improvisation adhering to the trend observed as in Section 5.1. The musicological hypothesis that these are the focal svaras of a rāga does not necessarily imply that these svaras are explored from the very beginning of the concert. Rather the performance starts from a lower svara (graha svara) and reaches the vādi in course of the gradual development of the melody.

Duration in melody: We compute the average duration of all salient svaras per BP in two groups of svaras: (i) group 1: vādi/samvādi, and (ii) group 2: rest. It is observed on the whole corpus that Pro of group 1 is higher than the group 2 for all rāgas. This reinforces the fact the term 'focus' or 'shine' (that qualifies vādi) is manifested in the temporal dimension. This also brings out a question whether we can predict the vādi/samvādi of a rāga from the melody by data-driven features. From the overall pitch histogram it is difficult to infer, but from our designed features, we observe an accuracy of 83% while predicting the vādi/samvādi of a given raga.

5.5 K5: Presence of possible pulsation in melody

There has been a discussion among musicians and musicologists whether there exists a pulsation in the melody of an ālāp in Hindustani music. Musicians agree there is an implicit pulsation present, but quantification is left to subjects. At the same time, the subjective bias only results in an octave difference, i.e., there is a harmonic relation among the pace in which the subjects tap to the melody. We propose a measure, through our data-driven approach, to estimate a possible metric for the pulsation. We assume that the pulse obtained from the melody would correlate to the percussive pulsation. We compute the ratio of inter-onset-interval of the most salient svaras across BPs. Figure 8 shows a pulsation at 0.8 seconds and its harmonics which correspond to 75 beats per minute (bpm) and the percussive tempo of the concert is approximately 40 bpm. The noise in the estimate may also follow from a few short BPs (e.g., BP index 3, 7 etc.) as observed in Figure 3. This measure, therefore, needs further investigation before we generalize over the corpus.

6. CONCLUSION AND FUTURE WORK

In this paper we performed a data driven exploration of implicit structures in melodies of Hindustani music. We outlined the motivation and relevance of computational approaches for quantitatively studying the underlying musical concepts. We computed musically relevant and easy-to-interpret acoustic features such as svara frequency and svara duration histogram. For computing these features we primarily used existing tools and techniques often used in information retrieval of Indian art music. We performed a quantitative analysis of 75 music concerts in Hindustani music in 10 different rāgas. With that we showed how the musical concepts are manifested in real-world performances and experimented with several ways to quantify them. With this we also corroborate some of the interesting music concepts and discover implicit relationships between svaras and duration in the temporal evolution of a rāga performance. In the future, one possible research direction would be to use these findings for characterizing artist-specific aspects and highlighting different nuances across gharanas in Hindustani music.

7. ACKNOWLEDGEMENT

This work received partial funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement 267583 (CompMusic). Part of the work was supported by Bharti Centre for Communication in IIT Bombay.

8. REFERENCES

[1] S. Bagchee. *Nād: Understanding Rāga Music*. Business Publications Inc, 1998.

[2] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X. Serra. Essentia: An audio analysis library for Music Information Retrieval. In *Proc. of Int. Soc. for Music Information Retrieval (ISMIR)*, pages 493–498, 2013.

[3] J. Bor, S. Rao, W. van der Meer, and J. Harvey. *The Raga Guide: A survey of 74 Hindustani ragas*. Nimbus Records with Rotterdam Conservatory of Music, 1999.

[4] A. Chakrabarty. *Shrutinandan: Towards Universal Music*. Macmillan, 2002.

[5] P. Chordia and S. Şentürk. Joint recognition of raag and tonic in north Indian music. *Computer Music Journal*, 37(3):82–98, 2013.

[6] A. Danielou. *The ragas of Northern Indian music*. Munshiram Manoharlal Publishers, New Delhi, 2010.

[7] A. K. Dey. *Nyasa in raga: The pleasant pause in Hindustani music*. Kanishka Publishers, 2008.

[8] S. Dutta and H. A. Murthy. Discovering typical motifs of a raga from one-liners of songs in Carnatic music. In *Int. Soc. for Music Information Retrieval (ISMIR)*, pages 397–402, Taipei, Taiwan, 2014.

[9] K. K. Ganguli. Guruji Padmashree Pt. Ajoy Chakrabarty: As I have seen Him. *Samakalika Sangeetham*, 3(2):93–100, October 2012.

[10] K. K. Ganguli. How do we 'See' & 'Say' a raga: A Perspective Canvas. *Samakalika Sangeetham*, 4(2):112–119, October 2013.

[11] K. K. Ganguli and P. Rao. Discrimination of melodic patterns in Indian classical music. In *Proc. of National Conference on Communications (NCC)*, February 2015.

[12] K. K. Ganguli, A. Rastogi, V. Pandit, P. Kantan, and P. Rao. Efficient melodic query based audio search for Hindustani vocal compositions. In *Proc. of Int. Soc. for Music Information Retrieval (ISMIR)*, pages 591–597, October 2015. Malaga, Spain.

[13] S. Gulati, A. Bellur, J. Salamon, H. G. Ranjani, V. Ishwar, H. A. Murthy, and X. Serra. Automatic tonic identification in Indian art music: approaches and evaluation. *Journal of New Music Research (JNMR)*, 43(1):55–73, 2014.

[14] S. Gulati, J. Serrà, K. K. Ganguli, and X. Serra. Landmark detection in Hindustani music melodies. In *International Computer Music Conference, Sound and Music Computing Conference*, pages 1062–1068, Athens, Greece, 2014.

[15] S. Gulati, J. Serra, V. Ishwar, S. Senturk, and X. Serra. Phrase-based rāga recognition using vector space modeling. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 66–70, Shanghai, China, 2016.

[16] S. Gulati, J. Serrà, V. Ishwar, and X. Serra. Mining melodic patterns in large audio collections of Indian art music. In *Int. Conf. on Signal Image Technology & Internet Based Systems (SITIS-MIRA)*, pages 264–271, Morocco, 2014.

[17] S. Gulati, J. Serrà, and X. Serra. Improving melodic similarity in Indian art music using culture-specific melodic characteristics. In *Proc. of Int. Soc. for Music Information Retrieval (ISMIR)*, pages 680–686, Malaga, Spain, 2015.

[18] N. A. Jairazbhoy. *The Rags of North Indian Music: Their Structure & Evolution*. Popular Prakashan, second edition, 2011.

[19] P. Rao, J. C. Ross, and K. K. Ganguli. Distinguishing raga-specific intonation of phrases with audio analysis. *Ninaad*, 26-27(1):59–68, December 2013.

[20] P. Rao, J. C. Ross, K. K. Ganguli, V. Pandit, V. Ishwar, A. Bellur, and H. A. Murthy. Classification of melodic motifs in raga music with time-series matching. *Journal of New Music Research (JNMR)*, 43(1):115–131, April 2014.

[21] S. Rao and P. Rao. An overview of Hindustani music in the context of Computational Musicology. *Journal of New Music Research (JNMR)*, 43(1):24–33, April 2014.

[22] R. C. Repetto, R. Gong, N. Kroher, and X. Serra. Comparison of the singing style of two jingju schools. In *International Society for Music Information Retrieval (ISMIR)*, pages 507–513, Malaga, Spain, October 2015.

[23] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[24] X. Serra. A multicultural approach to music information research. In *Proc. of Int. Conf. on Music Information Retrieval (ISMIR)*, pages 151–156, 2011.

[25] X. Serra. Creating research corpora for the computational study of music: the case of the Compmusic project. In *Proc. of the 53rd AES Int. Conf. on Semantic Audio*, London, 2014.

[26] W. van der Meer. *Hindustani music in the 20th century*. Martin US Nijhoff Publishers, 1980.

[27] P. Verma, T. P. Vinutha, P. Pandit, and P. Rao. Structural segmentation of hindustani concert audio with posterior features. In *Int.Conf. on Acoustics Speech and Signal Processing (ICASSP)*, pages 136–140. IEEE, 2015.

[28] T. Viswanathan and M. H. Allen. *Music in South India*. Oxford University Press, 2004.

[29] R. Widdess. Involving the performers in transcription and analysis: a collaborative approach to Dhrupad. *Ethnomusicology*, 38(1):59–79, 1994.

[30] R. Widdess. Dynamics of melodic discourse in indian music: Budhaditya mukherjee's ālāp in rāg Pūriyā-Kalyān. pages 187–224, 2011.

DEEP CONVOLUTIONAL NETWORKS ON THE PITCH SPIRAL FOR MUSIC INSTRUMENT RECOGNITION

Vincent Lostanlen and Carmine-Emanuele Cella
École normale supérieure, PSL Research University, CNRS, Paris, France

ABSTRACT

Musical performance combines a wide range of pitches, nuances, and expressive techniques. Audio-based classification of musical instruments thus requires to build signal representations that are invariant to such transformations. This article investigates the construction of learned convolutional architectures for instrument recognition, given a limited amount of annotated training data. In this context, we benchmark three different weight sharing strategies for deep convolutional networks in the time-frequency domain: temporal kernels; time-frequency kernels; and a linear combination of time-frequency kernels which are one octave apart, akin to a Shepard pitch spiral. We provide an acoustical interpretation of these strategies within the source-filter framework of quasi-harmonic sounds with a fixed spectral envelope, which are archetypal of musical notes. The best classification accuracy is obtained by hybridizing all three convolutional layers into a single deep learning architecture.

1. INTRODUCTION

Among the cognitive attributes of musical tones, pitch is distinguished by a combination of three properties. First, it is relative: ordering pitches from low to high gives rise to intervals and melodic patterns. Secondly, it is intensive: multiple pitches heard simultaneously produce a chord, not a single unified tone – contrary to loudness, which adds up with the number of sources. Thirdly, it does not depend on instrumentation: this makes possible the transcription of polyphonic music under a single symbolic system [5].

Tuning auditory filters to a perceptual scale of pitches provides a time-frequency representation of music signals that satisfies the first two of these properties. It is thus a starting point for a wide range of MIR applications, which can be separated in two categories: pitch-*relative* (e.g. chord estimation [13]) and pitch-*invariant* (e.g. instrument recognition [9]). Both aim at disentangling pitch from timbral content as independent factors of variability, a goal that is made possible by the third aforementioned property. This is pursued by extracting mid-level features on top of the spectrogram, be them engineered or learned from training data. Both approaches have their limitations: a "bag-of-features" lacks flexibility to represent fine-grain class boundaries, whereas a purely learned pipeline often leads to uninterpretable overfitting, especially in MIR where the quantity of thoroughly annotated data is relatively small.

In this article, we strive to integrate domain-specific knowledge about musical pitch into a deep learning framework, in an effort towards bridging the gap between feature engineering and feature learning.

Section 2 reviews the related work on feature learning for signal-based music classification. Section 3 demonstrates that pitch is the major factor of variability among musical notes of a given instrument, if described by their mel-frequency cepstra. Section 4 presents a typical deep learning architecture for spectrogram-based classification, consisting of two convolutional layers in the time-frequency domain and one densely connected layer. Section 5 introduces alternative convolutional architectures for learning mid-level features, along time and along a Shepard pitch spiral, as well as aggregation of multiple models in the deepest layers. Sections 6 discusses the effectiveness of the presented systems on a challenging dataset for music instrument recognition.

2. RELATED WORK

Spurred by the growth of annotated datasets and the democratization of high-performance computing, feature learning has enjoyed a renewed interest in recent years within the MIR community, both in supervised and unsupervised settings. Whereas unsupervised learning (e.g. k-means [25], Gaussian mixtures [14]) is employed to fit the distribution of the data with few parameters of relatively low abstraction and high dimensionality, state-of-the-art supervised learning consists of a deep composition of multiple non-linear transformations, jointly optimized to predict class labels, and whose behaviour tend to gain in abstraction as depth increases [27].

As compared to other deep learning techniques for audio processing, convolutional networks happen to strike the balance between learning capacity and robustness. The convolutional structure of learned transformations is derived from the assumption that the input signal, be it a one-

This work is supported by the ERC InvariantClass grant 320959. The source code to reproduce figures and experiments is freely available at www.github.com/lostanlen/ismir2016.

© Vincent Lostanlen and Carmine-Emanuele Cella. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Vincent Lostanlen and Carmine-Emanuele Cella. "Deep convolutional networks on the pitch spiral for music instrument recognition", 17th International Society for Music Information Retrieval Conference, 2016.

dimensional waveform or a two-dimensional spectrogram, is stationary — which means that content is independent from location. Moreover, the most informative dependencies between signal coefficients are assumed to be concentrated to temporal or spectrotemporal neighborhoods. Under such hypotheses, linear transformations can be learned efficiently by limiting their support to a small kernel which is convolved over the whole input. This method, known as weight sharing, decreases the number of parameters of each feature map while increasing the amount of data on which kernels are trained.

By design, convolutional networks seem well adapted to instrument recognition, as this task does not require a precise timing of the activation function, and is thus essentially a challenge of temporal integration [9, 14]. Furthermore, it benefits from an unequivocal ground truth, and may be simplified to a single-label classification problem by extracting individual stems from a multitrack dataset [2]. As such, it is often used a test bed for the development of new algorithms [17, 18], as well as in computational studies in music cognition [20, 21].

Some other applications of deep convolutional networks include onset detection [23], transcription [24], chord recognition [13], genre classification [3], downbeat tracking [8], boundary detection [26], and recommendation [27].

Interestingly, many research teams in MIR have converged to employ the same architecture, consisting of two convolutional layers and two densely connected layers [7,13,15,17,18,23,26], and this article makes no exception. However, there is no clear consensus regarding the weight sharing strategies that should be applied to musical audio streams: convolutions in time or in time-frequency coexist in the recent literature. A promising paradigm [6, 8], at the interaction between feature engineering and feature learning, is to extract temporal or spectrotemporal descriptors of various low-level modalities, train specific convolutional layers on each modality to learn mid-level features, and hybridize information at the top level. Recognizing that this idea has been successfully applied to large-scale artist recognition [6] as well as downbeat tracking [8], we aim to proceed in a comparable way for instrument recognition.

3. HOW INVARIANT IS THE MEL-FREQUENCY CEPSTRUM ?

The mel scale is a quasi-logarithmic function of acoustic frequency designed such that perceptually similar pitch intervals appear equal in width over the full hearing range. This section shows that engineering transposition-invariant features from the mel scale does not suffice to build pitch invariants for complex sounds, thus motivating further inquiry.

The time-frequency domain produced by a constant-Q filter bank tuned to the mel scale is covariant with respect to pitch transposition of pure tones. As a result, a chromatic scale played at constant speed would draw parallel, diagonal lines, each of them corresponding to a different

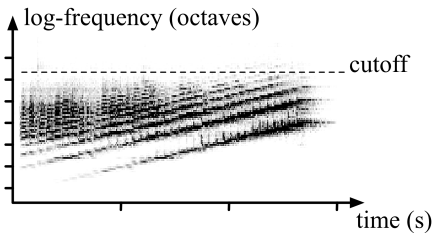


Figure 1: Constant-Q spectrogram of a chromatic scale played by a tuba. Although the harmonic partials shift progressively, the spectral envelope remains unchanged, as revealed by the presence of a fixed cutoff frequency. See text for details.

partial wave. However, the physics of musical instruments constrain these partial waves to bear a negligible energy if their frequencies are beyond the range of acoustic resonance.

As shown on Figure 1, the constant-Q spectrogram of a tuba chromatic scale exhibits a fixed, cutoff frequency at about 2.5 kHz, which delineates the support of its spectral envelope. This elementary observation implies that realistic pitch changes cannot be modeled by translating a rigid spectral template along the log-frequency axis. The same property is verified for a wide class of instruments, especially brass and woodwinds. As a consequence, the construction of powerful invariants to musical pitch is not amenable to delocalized operations on the mel-frequency spectrum, such as a discrete cosine transform (DCT) which leads to the mel-frequency cepstral coefficients (MFCC), often used in audio classification [9, 14].

To validate the above claim, we have extracted the MFCC of 1116 individual notes from the RWC dataset [10], as played by 6 instruments, with 32 pitches, 3 nuances, and 2 interprets and manufacturers. When more than 32 pitches were available (e.g. piano), we selected a contiguous subset of 32 pitches in the middle register. Following a well-established rule [9, 14], the MFCC were defined the 12 lowest nonzero "quefrencies" among the DCT coefficients extracted from a filter bank of 40 mel-frequency bands. We then have computed the distribution of squared Euclidean distances between musical notes in the 12-dimensional space of MFCC features.

Figure 2 summarizes our results. We found that restricting the cluster to one nuance, one interpret, or one manufacturer hardly reduces intra-class distances. This suggests that MFCC are fairly successful in building invariant representations to such factors of variability. In contrast, the cluster corresponding to each instrument is shrinked if decomposed into a mixture of same-pitch clusters, sometimes by an order of magnitude. In other words, most of the variance in an instrument cluster of mel-frequency cepstra is due to pitch transposition.

Keeping less than 12 coefficients certainly improves invariance, yet at the cost of inter-class discriminability, and vice versa. This experiment shows that the mel-frequency cepstrum is perfectible in terms of invariance-

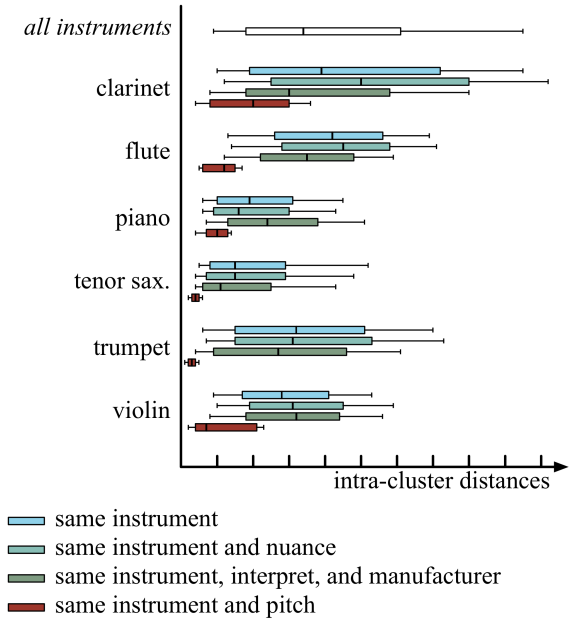


Figure 2: Distributions of squared Euclidean distances among various MFCC clusters in the RWC dataset. Whisker ends denote lower and upper deciles. See text for details.

discriminability tradeoff, and that there remains a lot to be gained by feature learning in this area.

4. DEEP CONVOLUTIONAL NETWORKS

A deep learning system for classification is built by stacking multiple layers of weakly nonlinear transformations, whose parameters are optimized such that the top-level layer fits a training set of labeled examples. This section introduces a typical deep learning architecture for audio classification and describes the functioning of each layer.

Each layer in a convolutional network typically consists in the composition of three operations: two-dimensional convolutions, application of a pointwise nonlinearity, and local pooling. The deep feed-forward network made of two convolutional layers and two densely connected layers, on which our experiment are conducted, has become a *de facto* standard in the MIR community [7, 13, 15, 17, 18, 23, 26]. This ubiquity in the literature suggests that a four-layer network with two convolutional layers is well adapted to supervised audio classification problems of moderate size.

The input of our system is a constant-Q spectrogram, which is very comparable to a mel-frequency spectrogram. We used the implementation from the librosa package [19] with $Q = 12$ filters per octave, center frequencies ranging from A_1 (55 Hz) to A_9 (14 kHz), and a hop size of 23 ms. Furthermore, we applied nonlinear perceptual weighting of loudness in order to reduce the dynamic range between the fundamental partial and its upper harmonics. A 3-second sound excerpt $\boldsymbol{x}[t]$ is represented by a time-frequency matrix $\boldsymbol{x_1}[t, k_1]$ of width $T = 128$ samples and height $K_1 = 96$ frequency bands.

A convolutional operator is defined as a family $\boldsymbol{W_2}[\tau, \kappa_1, k_2]$ of K_2 two-dimensional filters, whose impulse repsonses are all constrained to have width Δt and height Δk_1. Element-wise biases $\boldsymbol{b_2}[k_2]$ are added to the convolutions, resulting in the three-way tensor

$$\begin{aligned}&\boldsymbol{y_2}[t, k_1, k_2] \\ &= \boldsymbol{b_2}[k_2] + \boldsymbol{W_2}[t, k_1, k_2] \overset{t,k_1}{*} \boldsymbol{x_1}[t, k_1] \\ &= \boldsymbol{b_2}[k_2] + \sum_{\substack{0 \leq \tau < \Delta t \\ 0 \leq \kappa_1 < \Delta k_1}} \boldsymbol{W_2}[\tau, \kappa_1, k_2] \boldsymbol{x_1}[t - \tau, k_1 - \kappa_1]. \quad (1)\end{aligned}$$

The pointwise nonlinearity we have chosen is the rectified linear unit (ReLU), with a rectifying slope of $\alpha = 0.3$ for negative inputs.

$$\boldsymbol{y_2^+}[t, k_1, k_2] = \begin{cases} \alpha \boldsymbol{y_2}[t, k_1, k_2] & \text{if } \boldsymbol{y_2}[t, k_1, k_2] < 0 \\ \boldsymbol{y_2}[t, k_1, k_2] & \text{if } \boldsymbol{y_2}[t, k_1, k_2] \geq 0 \end{cases} \quad (2)$$

The pooling step consists in retaining the maximal activation among neighboring units in the time-frequency domain (t, k_1) over non-overlapping rectangles of width Δt and height Δk_1.

$$\boldsymbol{x_2}[t, k_1, k_2] = \max_{\substack{0 \leq \tau < \Delta t \\ 0 \leq \kappa_1 < \Delta k_1}} \left\{ \boldsymbol{y_2^+}[t - \tau, k_1 - \kappa_1, k_2] \right\} \quad (3)$$

The hidden units in $\boldsymbol{x_2}$ are in turn fed to a second layer of convolutions, ReLU, and pooling. Observe that the corresponding convolutional operator $\boldsymbol{W_3}[\tau, \kappa_1, k_2, k_3]$ performs a linear combination of time-frequency feature maps in $\boldsymbol{x_2}$ along the variable k_2.

$$\begin{aligned}&\boldsymbol{y_3}[t, k_1, k_3] \\ &= \sum_{k_2} \boldsymbol{b_3}[k_2, k_3] + \boldsymbol{W_3}[t, k_1, k_2, k_3] \overset{t,k_1}{*} \boldsymbol{x_2}[t, k_1, k_2]. \quad (4)\end{aligned}$$

Tensors $\boldsymbol{y_3^+}$ and $\boldsymbol{x_3}$ are derived from $\boldsymbol{y_3}$ by ReLU and pooling, with formulae similar to Eqs. (2) and (3). The third layer consists of the linear projection of $\boldsymbol{x_3}$, viewed as a vector of the flattened index (t, k_1, k_3), over K_4 units:

$$\boldsymbol{y_4}[k_4] = \boldsymbol{b_4}[k_4] + \sum_{t, k_1, k_3} \boldsymbol{W_4}[t, k_1, k_3, k_4] \boldsymbol{x_3}[t, k_1, k_3] \quad (5)$$

We apply a ReLU to $\boldsymbol{y_4}$, yielding $\boldsymbol{x_4}[k_4] = \boldsymbol{y_4^+}[k_4]$. Finally, we project $\boldsymbol{x_4}$, onto a layer of output units $\boldsymbol{y_5}$ that should represent instrument activations:

$$\boldsymbol{y_5}[k_5] = \sum_{k_4} \boldsymbol{W_5}[k_4, k_5] \boldsymbol{x_4}[k_4]. \quad (6)$$

The final transformation is a softmax nonlinearity, which ensures that output coefficients are non-negative and sum to one, hence can be fit to a probability distribution:

$$\boldsymbol{x_5}[k_5] = \frac{\exp \boldsymbol{y_5}[k_5]}{\sum_{\kappa_5} \exp \boldsymbol{y_5}[\kappa_5]}. \quad (7)$$

Given a training set of spectrogram-instrument pairs $(\boldsymbol{x_1}, k)$, all weigths in the network are iteratively updated

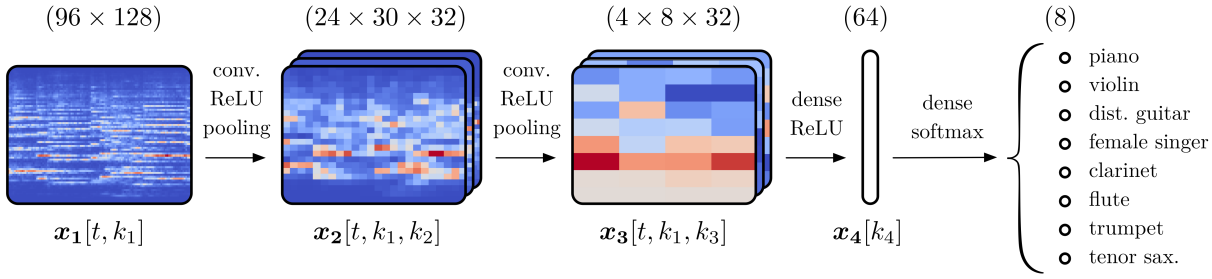


Figure 3: A two-dimensional deep convolutional network trained on constant-Q spectrograms. See text for details.

to minimize the stochastic cross-entropy loss $\mathscr{L}(\boldsymbol{x_5}, k) = -\log \boldsymbol{x_5}[k]$ over shuffled mini-batches of size 32 with uniform class distribution. The pairs $(\boldsymbol{x_1}, k)$ are extracted on the fly by selecting non-silent regions at random within a dataset of single-instrument audio recordings. Each 3-second spectrogram $\boldsymbol{x_1}[t, k_1]$ within a batch is globally normalized such that the whole batch had zero mean and unit variance. At training time, a random dropout of 50% is applied to the activations of $\boldsymbol{x_3}$ and $\boldsymbol{x_4}$. The learning rate policy for each scalar weight in the network is Adam [16], a state-of-the-art online optimizer for gradient-based learning. Mini-batch training is stopped after the average training loss stopped decreasing over one full epoch of size 8192. The architecture is built using the Keras library [4] and trained on a graphics processing unit within minutes.

5. IMPROVED WEIGHT SHARING STRATEGIES

Although a dataset of music signals is unquestionably stationary over the time dimension – at least at the scale of a few seconds – it cannot be taken for granted that all frequency bands of a constant-Q spectrogram would have the same local statistics [12]. In this section, we introduce two alternative architectures to address the nonstationarity of music on the log-frequency axis, while still leveraging the efficiency of convolutional representations.

Many are the objections to the stationarity assumption among local neighborhoods in mel frequency. Notably enough, one of the most compelling is derived from the classical source-filter model of sound production. The filter, which carries the overall spectral envelope, is affected by intensity and playing style, but not by pitch. Conversely, the source, which consists of a pseudo-periodic wave, is transposed in frequency under the action of pitch. In order to extract the discriminative information present in both terms, it is first necessary to disentangle the contributions of source and filter in the constant-Q spectrogram. Yet, this can only be achieved by exploiting long-range correlations in frequency, such as harmonic and formantic structures. Besides, the harmonic comb created by the Fourier series of the source makes an irregular pattern on the log-frequency axis which is hard to characterize by local statistics.

5.1 One-dimensional convolutions at high frequencies

Facing nonstationary constant-Q spectra, the most conservative workaround is to increase the height $\Delta\kappa_1$ of each convolutional kernel up to the total number of bins K_1 in the spectrogram. As a result, $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ are no longer transposed over adjacent frequency bands, since convolutions are merely performed over the time variable. The definition of $\boldsymbol{y_2}[t, k_1, k_2]$ rewrites as

$$\begin{aligned}&\boldsymbol{y_2}[t, k_1, k_2]\\ &= \boldsymbol{b_2}[k_2] + \boldsymbol{W_2}[t, k_1, k_2] \overset{t}{*} \boldsymbol{x_1}[t, k_1]\\ &= \boldsymbol{b_2}[k_2] + \sum_{0 \leq \tau < \Delta t} \boldsymbol{W_2}[\tau, k_1, k_2] \boldsymbol{x_1}[t - \tau, k_1],\end{aligned} \tag{8}$$

and similarly for $\boldsymbol{y_3}[t, k_1, k_3]$. While this approach is theoretically capable of encoding pitch invariants, it is prone to early specialization of low-level features, thus not fully taking advantage of the network depth.

However, the situation is improved if the feature maps are restricted to the highest frequencies in the constant-Q spectrum. It should be observed that, around the n^{th} partial of a quasi-harmonic sound, the distance in log-frequency between neighboring partials decays like $1/n$, and the unevenness between those distances decays like $1/n^2$. Consequently, at the topmost octaves of the constant-Q spectrum, where n is equal or greater than Q, the partials appear close to each other and almost evenly spaced. Furthermore, due to the logarithmic compression of loudness, the polynomial decay of the spectral envelope is linearized: thus, at high frequencies, transposed pitches have similar spectra up to some additive bias. The combination of these two phenomena implies that the correlation between constant-Q spectra of different pitches is greater towards high frequencies, and that the learning of polyvalent feature maps becomes tractable.

In our experiments, the one-dimensional convolutions over the time variable range from A_6 (1.76 kHz) to A_9 (14 kHz).

5.2 Convolutions on the pitch spiral at low frequencies

The weight sharing strategy presented above exploits the facts that, at high frequencies, quasi-harmonic partials are

numerous, and that the amount of energy within a frequency band is independent of pitch. At low frequencies, we make the exact opposite assumptions: we claim that the harmonic comb is sparse and covariant with respect to pitch shift. Observe that, for any two distinct partials taken at random between 1 and n, the probability that they are in octave relation is slightly above $1/n$. Thus, for n relatively low, the structure of harmonic sounds is well described by merely measuring correlations between partials one octave apart. This idea consists in rolling up the log-frequency axis into a Shepard pitch spiral, such that octave intervals correspond to full turns, hence aligning all coefficients of the form $\boldsymbol{x_1}[t, k_1 + Q \times j_1]$ for $j_1 \in \mathbb{Z}$ onto the same radius of the spiral. Therefore, correlations between power-of-two harmonics are revealed by the octave variable j_1.

To implement a convolutional network on the pitch spiral, we crop the constant-Q spectrogram in log-frequency into $J_1 = 3$ half-overlapping bands whose height equals $2Q$, that is two octaves. Each feature map in the first layer, indexed by k_2, results from the sum of convolutions between a time-frequency kernel and a band, thus emulating a linear combination in the pitch spiral with a 3-d tensor $\boldsymbol{W_2}[\tau, \kappa_1, j_1, k_2]$ at fixed k_2. The definition of $\boldsymbol{y_2}[t, k_1, k_2]$ rewrites as

$$\boldsymbol{y_2}[t, k_1, k_2] = \boldsymbol{b_2}[k_2] + \sum_{\tau, \kappa_1, j_1} \boldsymbol{W_2}[\tau, \kappa_1, j_1, k_2] \times \boldsymbol{x_1}[t - \tau, k_1 - \kappa_1 - Qj_1]. \quad (9)$$

The above is different from training two-dimensional kernel on a time-chroma-octave tensor, since it does not suffer from artifacts at octave boundaries.

The linear combinations of frequency bands that are one octave apart, as proposed here, bears a resemblance with engineered features for music instrument recognition [22], such as tristimulus, empirical inharmonicity, harmonic spectral deviation, odd-to-even harmonic energy ratio, as well as octave band signal intensities (OBSI) [14].

Guaranteeing the partial index n to remain low is achieved by restricting the pitch spiral to its lowest frequencies. This operation also partially circumvents the problem of fixed spectral envelope in musical sounds, thus improving the validness of the stationarity assumption. In our experiments, the pitch spiral ranges from A_2 (110 Hz) to A_6 (1.76 kHz).

In summary, the classical two-dimensional convolutions make a stationarity assumption among frequency neighborhoods. This approach gives a coarse approximation of the spectral envelope. Resorting to one-dimensional convolutions allows to disregard nonstationarity, but does not yield a pitch-invariant representation per se: thus, we only apply them at the topmost frequencies, i.e. where the invariance-to-stationarity ratio in the data is already favorable. Conversely, two-dimensional convolutions on the pitch spiral addresses the invariant representation of sparse, transposition-covariant spectra: as such, they are best suited to the lowest frequencies, i.e. where partials are further apart and pitch changes can be approximated by log-frequency translations. The next section reports experiments on instrument recognition that capitalize on these considerations.

	minutes	tracks	minutes	tracks
piano	58	28	44	15
violin	51	14	49	22
dist. guitar	15	14	17	11
female singer	10	11	19	12
clarinet	10	7	13	18
flute	7	5	53	29
trumpet	4	6	7	27
tenor sax.	3	3	6	5
total	158	88	208	139

Table 1: Quantity of data in the training set (left) and test set (right). The training set is derived from MedleyDB. The test set is derived from MedleyDB for distorted electric guitar and female singer, and from [14] for other instruments.

6. APPLICATIONS

The proposed algorithms are trained on a subset of MedleyDB v1.1. [2], a dataset of 122 multitracks annotated with instrument activations. We extracted the monophonic stems corresponding to a selection of eight pitched instruments (see Table 1). Stems with leaking instruments in the background were discarded.

The evaluation set consists of 126 recordings of solo music collected by Joder et al. [14], supplemented with 23 stems of electric guitar and female voice from MedleyDB. In doing so, guitarists and vocalists were thoroughly put either in the training set or the test set, to prevent any artist bias. We discarded recordings with extended instrumental techniques, since they are extremely rare in MedleyDB. Constant-Q spectrograms from the evaluation set were split into half-overlapping, 3-second excerpts.

For the two-dimensional convolutional network, each of the two layers consists of 32 kernels of width 5 and height 5, followed by a max-pooling of width 5 and height 3. Expressed in physical units, the supports of the kernels are respectively equal to 116 ms and 580 ms in time, 5 and 10 semitones in frequency. For the one-dimensional convolutional network, each of two layers consists of 32 kernels of width 3, followed by a max-pooling of width 5. Observe that the temporal supports match those of the two-dimensional convolutional network. For the convolutional network on the pitch spiral, the first layer consists of 32 kernels of width 5, height 3 semitones, and a radial length of 3 octaves in the spiral. The max-pooling operator and the second layer are the same as in the two-dimensional convolutional network.

In addition to the three architectures above, we build hybrid networks implementing more than one of the weight sharing strategy presented above. In all architectures, the densely connected layers have $K_4 = 64$ hidden units and

	piano	violin	dist. guitar	female singer	clarinet	flute	trumpet	tenor sax.	average
bag-of-features and random forest	**99.7 (± 0.1)**	**76.2 (± 3.1)**	**92.7 (± 0.4)**	81.6 (± 1.5)	49.9 (± 0.8)	22.5 (± 0.8)	63.7 (± 2.1)	4.4 (± 1.1)	61.4 (± 0.5)
spiral (36k parameters)	86.9 (± 5.8)	37.0 (± 5.6)	72.3 (± 6.2)	84.4 (± 6.1)	61.1 (± 8.7)	30.0 (± 4.0)	54.9 (± 6.6)	52.7 (± 16.4)	59.9 (± 2.4)
1-d (20k parameters)	73.3 (± 11.0)	43.9 (± 6.3)	91.8 (± 1.1)	82.9 (± 1.9)	28.8 (± 5.0)	**51.3 (± 13.4)**	63.3 (± 5.0)	**59.0 (± 6.8)**	61.8 (± 0.9)
2-d, 32 kernels (93k parameters)	96.8 (± 1.4)	68.5 (± 9.3)	86.0 (± 2.7)	80.6 (± 1.7)	81.3 (± 4.1)	44.4 (± 4.4)	68.0 (± 6.2)	48.4 (± 5.3)	69.1 (± 2.0)
spiral & 1-d (55k parameters)	96.5 (± 2.3)	47.6 (± 6.1)	90.2 (± 2.3)	84.5 (± 2.8)	79.6 (± 2.1)	41.8 (± 4.1)	59.8 (± 1.9)	53.0 (± 16.5)	69.1 (± 2.0)
spiral & 2-d (128k parameters)	97.6 (± 0.8)	73.3 (± 4.4)	86.5 (± 4.5)	**86.9 (± 3.6)**	**82.3 (± 3.2)**	45.8 (± 2.9)	66.9 (± 5.8)	51.2 (± 10.6)	71.7 (± 2.0)
1-d & 2-d (111k parameters)	96.5 (± 0.9)	72.4 (± 5.9)	86.3 (± 5.2)	91.0 (± 5.5)	73.3 (± 6.4)	49.5 (± 6.9)	67.7 (± 2.5)	55.0 (± 11.5)	73.8 (± 2.3)
2-d & 1-d & spiral (147k parameters)	97.8 (± 0.6)	70.9 (± 6.1)	88.0 (± 3.7)	85.9 (± 3.8)	75.0 (± 4.3)	48.3 (± 6.6)	67.3 (± 4.4)	**59.0 (± 7.3)**	**74.0 (± 0.6)**
2-d, 48 kernels (158k parameters)	96.5 (± 1.4)	69.3 (± 7.2)	84.5 (± 2.5)	84.2 (± 5.7)	77.4 (± 6.0)	45.5 (± 7.3)	**68.8** (± 1.8)	52.6 (± 10.1)	71.7 (± 2.0)

Table 2: Test set accuracies for all presented architectures. All convolutional layers have 32 kernels unless stated otherwise.

$K_5 = 8$ output units.

In order to compare the results against shallow classifiers, we also extracted a typical "bag-of-features" over half-overlapping, 3-second excerpts in the training set. These features consist of the means and standard deviations of spectral shape descriptors, i.e. centroid, bandwidth, skewness, and rolloff; the mean and standard deviation of the zero-crossing rate in the time domain; and the means of MFCC as well as their first and second derivative. We trained a random forest of 100 decision trees on the resulting feature vector of dimension 70, with balanced class probability.

Results are summarized in Table 2. First of all, the bag-of-features approach presents large accuracy variations between classes, due to the unbalance of available training data. In contrast, most convolutional models, especially hybrid ones, show less correlation between the amount of training data in the class and the accuracy. This suggests that convolutional networks are able to learn polyvalent mid-level features that can be re-used a test time to discriminate rarer classes.

Furthermore, 2-d convolutions outperform other non-hybrid weight sharing strategies. However, a class with broadband temporal modulations, namely the distorted electric guitar, is best classified with 1-d convolutions.

Hybridizing 2-d with either 1-d or spiral convolutions provide consistent, albeit small improvements with respect to 2-d alone. The best overall accuracy is reached by the full hybridization of all three weight sharing strategies, because of a performance boost for the rarest classes.

The accuracy gain by combining multiple models could simply be the result of a greater number of parameters. To refute this hypothesis, we train a 2-d convolutional network with 48 kernels instead of 32, so as to match the budget of the full hybrid model, i.e. about 150k parameters. The performance is certainly increased, but not up to the hybrid models involving 2-d convolutions, which have less parameters. Increasing the number of kernels even more cause the accuracy to level out and the variance between trials to increase.

Running the same experiments with broader frequency ranges of 1-d and spiral convolutions often led to a degraded performance, and are thus not reported.

7. CONCLUSIONS

Understanding the influence of pitch in audio streams is paramount to the design of an efficient system for automated classification, tagging, and similarity retrieval in music. We have presented deep learning methods to address pitch invariance while preserving good timbral discriminability. It consists in training a feed-forward convolutional network over the constant-Q spectrogram, with three different weight sharing strategies according to the type of input: along time at high frequencies (above 2 kHz), on a Shepard pitch spiral at low frequencies (below 2 kHz), and in time-frequency over both high and low frequencies.

A possible improvement of the presented architecture would be to place a third convolutional layer in the time domain before performing long-term max-pooling, hence modelling the joint dynamics of the three mid-level feature maps. Future work will investigate the association of the presented weight sharing strategies with recent advances in deep learning for music informatics, such as data augmentation [18], multiscale representations [1,11], and adversarial training [15].

8. REFERENCES

[1] Joakim Andén, Vincent Lostanlen, and Stéphane Mallat. Joint time-frequency scattering for audio classification. In *Proc. MLSP*, 2015.

[2] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Bello. MedleyDB: a multitrack dataset for annotation-intensive MIR research. In *Proc. ISMIR*, 2014.

[3] Keunwoo Choi, George Fazekas, Mark Sandler, Jeonghee Kim, and Naver Labs. Auralisation of deep convolutional neural networks: listening to learned features. In *Proc. ISMIR*, 2015.

[4] François Chollet. Keras: a deep learning library for Theano and TensorFlow, 2015.

[5] Alain de Cheveigné. Pitch perception. In *Oxford Handbook of Auditory Science: Hearing*, chapter 4, pages 71–104. Oxford University Press, 2005.

[6] Sander Dieleman, Philémon Brakel, and Benjamin Schrauwen. Audio-based music classification with a pretrained convolutional network. In *Proc. ISMIR*, 2011.

[7] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *Proc. ICASSP*, 2014.

[8] Simon Durand, Juan P. Bello, Bertrand David, and Gaël Richard. Feature-adapted convolutional neural networks for downbeat tracking. In *Proc. ICASSP*, 2016.

[9] Antti Eronen and Anssi Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. In *Proc. ICASSP*, 2000.

[10] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC music database: music genre database and musical instrument sound database. In *Proc. ISMIR*, 2003.

[11] Philippe Hamel, Yoshua Bengio, and Douglas Eck. Building musically-relevant audio features through multiple timescale representations. In *Proc. ISMIR*, 2012.

[12] Eric J. Humphrey, Juan P. Bello, and Yann Le Cun. Feature learning and deep architectures: New directions for music informatics. *JIIS*, 41(3):461–481, 2013.

[13] Eric J. Humphrey, Taemin Cho, and Juan P. Bello. Learning a robust tonnetz-space transform for automatic chord recognition. In *Proc. ICASSP*, 2012.

[14] Cyril Joder, Slim Essid, and Gaël Richard. Temporal integration for audio classification with application to musical instrument classification. *IEEE TASLP*, 17(1):174–186, 2009.

[15] Corey Kereliuk, Bob L. Sturm, and Jan Larsen. Deep Learning and Music Adversaries. *IEEE Trans. Multimedia*, 17(11):2059–2071, 2015.

[16] Diederik P. Kingma and Jimmy Lei Ba. Adam: a method for stochastic optimization. In *Proc. ICML*, 2015.

[17] Peter Li, Jiyuan Qian, and Tian Wang. Automatic instrument recognition in polyphonic music using convolutional neural networks. *arXiv preprint*, 1511.05520, 2015.

[18] Brian McFee, Eric J. Humphrey, and Juan P. Bello. A software framework for musical data augmentation. In *Proc. ISMIR*, 2015.

[19] Brian McFee, Matt McVicar, Colin Raffel, Dawen Liang, Oriol Nieto, Eric Battenberg, Josh Moore, Dan Ellis, Ryuichi Yamamoto, Rachel Bittner, Douglas Repetto, Petr Viktorin, João Felipe Santos, and Adrian Holovaty. librosa: 0.4.1. zenodo. 10.5281/zenodo.18369, October 2015.

[20] Michael J. Newton and Leslie S. Smith. A neurally inspired musical instrument classification system based upon the sound onset. *JASA*, 131(6):4785, 2012.

[21] Kailash Patil, Daniel Pressnitzer, Shihab Shamma, and Mounya Elhilali. Music in our ears: the biological bases of musical timbre perception. *PLoS Comput. Biol.*, 8(11):e1002759, 2012.

[22] Geoffroy Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Technical report, Ircam, 2004.

[23] Jan Schlüter and Sebastian Böck. Improved musical onset detection with convolutional neural networks. In *Proc. ICASSP*, 2014.

[24] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic music transcription. *arXiv preprint*, 1508.01774, 2015.

[25] Dan Stowell and Mark D. Plumbley. Automatic large-scale classification of bird sounds is strongly improved by unsupervised feature learning. *PeerJ*, 2:e488, 2014.

[26] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary detection in music structure analysis using convolutional neural networks. In *Proc. ISMIR*, 2014.

[27] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Proc. NIPS*, 2013.

DTV-BASED MELODY CUTTING FOR DTW-BASED MELODY SEARCH AND INDEXING IN QBH SYSTEMS

Bartłomiej Stasiak
Institute of Information Technology, Lodz University of Technology,
bartlomiej.stasiak@p.lodz.pl

ABSTRACT

Melody analysis is an important processing step in several areas of Music Information Retrieval (MIR). Processing the pitch values extracted from raw input audio signal may be computationally complex as it requires substantial effort to reduce the uncertainty resulting i.a. from tempo variability and transpositions. A typical example is the melody matching problem in Query-by-Humming (QbH) systems, where Dynamic Time Warping (DTW) and note-based approaches are typically applied.

In this work we present a new, simple and efficient method of investigating the melody content which may be used for approximate, preliminary matching of melodies irrespective of their tempo and length. The proposed solution is based on Discrete Total Variation (DTV) of the melody pitch vector, which may be computed in linear time. We demonstrate its practical application for finding the appropriate melody cutting points in the R^*-tree-based DTW indexing framework. The experimental validation is based on a dataset of 4431 queries and over 4000 template melodies, constructed specially for testing Query-by-Humming systems.

1. INTRODUCTION

Content-based search and retrieval is becoming a popular and attractive way to locate relevant resources in the ever-growing multimedia collections and databases. For Music Information Retrieval (MIR) several important application areas have been defined over the last decades, with *Audio Fingerprinting*, and *Query by Humming* (QbH) being perhaps the most prominent examples. The latter one is specific as it is exclusively based on the user-generated sound signal and it depends mostly on a single parameter of this signal – the pitch of the consecutive notes, forming the melody sung by the user.

In a typical QbH system the *query* in the form of raw audio data is subjected to a pitch-tracking procedure, which yields a sequence of pitch values in consecutive time frames, often referred to as a *pitch vector*. The music resources in the database are represented by *templates*, having the similar form, so that the search is essentially based on simply comparing the pitch vectors in order to find the template melody best matching the query melody.

An additional step of note segmentation may be used to obtain symbolic representation, explicitly defining the pitch and length of separate notes. In this case, several efficient methods based on e.g. transportation distance or string matching algorithms may be used. This approach enables fast searching, although it is difficult to obtain high precision due to unavoidable ambiguities of the note segmentation step. Comparing the pitch vectors directly usually yields higher-quality results but on the cost of the increased computational complexity, as the local tempo variations require to use tools for automatic alignment between the compared melodies.

Dynamic Time Warping (DTW) is a standard method applied for comparing data sequences, generated by processes that may exhibit substantial, yet semantically irrelevant local decelerations and accelerations. The examples include e.g. handwritten signature recognition or gait recognition for biometric applications, sign language analysis, spoken word recognition and many other problems involving temporal patterns analysis. It is also a method of choice for direct comparison of pitch vectors in the Query by Humming task.

2. PREVIOUS WORK

Early works on the QbH systems date back to the 1990's, with some background concepts and techniques being introduced much earlier [21, 24]. Initially, the symbolic, note-based approach was used [6, 20, 30], often in the simplified form comprising only melody direction change (U/D/S - Up/Down/the Same) [6, 21]. In the following decade the direct pitch sequence matching with Hidden Markov Models (HMM) [26] and Dynamic Time Warping [11, 19] was proposed and extensively used, in parallel to note-based approaches employing transportation distances, such as Earth Mover's Distance (EMD) [28, 29]. In many practical QbH systems, such as those presented in the annual MIREX competition [1, 27], a multi-stage approach is applied involving the application of the EMD to filter out most of the non-matching candidate templates, leaving only the most promising ones for the accurate, but more computationally expensive DTW-based search [31, 32, 34].

© Bartłomiej Stasiak. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Bartłomiej Stasiak. "DTV-based melody cutting for DTW-based melody search and indexing in QbH systems", 17th International Society for Music Information Retrieval Conference, 2016.

Another possibility of search time optimization is to accelerate the computation of DTW itself. For this purpose several methods have been proposed, including iterative deepening [2], Windowed Time Warping [18], FastDTW [25] and SparseDTW [3].

Yet another approach, which is of special interest to us, is to apply efficient DTW indexing techniques, based on lower bounding functions [13]. These methods reduce computational complexity by limiting the number of times the DTW algorithm must be run, but – unlike the aforementioned EMD-based multi-stage systems – they are not domain specific. Introduced by Keogh [13, 16] as a general-purpose method for time-series matching, they have also been successfully applied for the Query by Humming task [17, 35].

2.1 Indexing the dynamic time warping

DTW indexing is based on a more general approach to indexing time series, known as GEMINI framework (GEneric Multimedia INdexIng) [5, 14]. In this approach the sequences are indexed with R-trees, R*-trees or other Spatial Access Methods (SAM) [7] after being subjected to dimensionality reduction transformation. The typical SAMs require that the data are represented in a not more than 12-16–dimensional index space I [14, 15]. Searching in the index space is guaranteed to yield a superset of all the relevant templates (i.e. it will produce no false rejections), provided that a proper distance measure ρ_I is defined in I. Let N denote the length of the original time series and let $M \ll N$ be the number of dimensions of the index space. It may be shown [5] that when the distance ρ_X between elements T_N, Q_N of the input space X is properly bounded from below by the distance between their low-dimensional representations T_M, Q_M in the index space, i.e. when:

$$\rho_I(T_M, Q_M) \leq \rho_X(T_N, Q_N) \ , \tag{1}$$

then it is possible to construct an indexing mechanism which guarantees no false dismissals. The efficient, SAM-optimized query in the index space may only return some false positives which are then eliminated by direct matching of the time series in the original, input space X. Depending on the tightness of the lower bound (Eq. 1), the number of times the matching must be done in X may be reduced even by orders of magnitude. The detailed description of the appropriate algorithms for k-nearest neighbor search and range queries may be found in [5, 14].

The generality of the GEMINI framework enables its application with many dimensionality reducing transforms, based on e.g. discrete Fourier transform (DFT) [5], Haar Wavelet Transform [12] or piecewise aggregate approximation (PAA) [14, 33]. However, comparing sequences under dynamic time warping differs quite significantly from the case of Euclidean spaces, i.a. because DTW is not – strictly speaking – a *metric* (it does not satisfy the triangle inequality). It has been however shown that it is possible to define a valid lower-bounding distance measure [13] when proper global constraints, such as Sakoe and Chiba band [24] or Itakura parallelogram [9] are used. The dimensionality reduction may be obtained by the simple PAA algorithm, as demonstrated by Keogh in [13]. A more general approach, based on properties of container-invariant time series envelopes, was introduced by Zhu and Shasha, who extended the lower-bounding criterion to the whole class of linear transforms [35].

The aforementioned techniques of DTW indexing may be successfully applied to accelerate the melody matching in Query by Humming systems, as demonstrated in [35] on an example of a small music database of 50 Beatles songs. However, in real-life, large-scale systems some practical problems are likely to occur, especially when heterogeneous audio material is used as input for querying the database.

One of these problems is that the actual length and tempo of the query, with respect to the potentially matching template, are not known in advance. As we will demonstrate in the following section, this uncertainty makes the use of the global constraints of the DTW difficult, which in turn put in doubt the practical applicability of the DTW indexing schemes in the Query by Humming task.

As a remedy, we propose a novel solution, based on computation of Discrete Total Variation (DTV) of the pitch vector, which enables to assess the optimal cutting point of the query with respect to the template (Sect. 4). In this way, the DTW indexing becomes feasible even for diversified input data, containing the queries of varied length and tempo. Moreover, the analysis of the DTV may appear beneficial also for fixed-length queries. This conclusion will be supported by the experimental results presented in Section 5.

3. PROBLEM SETTING

The implicit assumption underlying the efficient DTW indexing methods introduced in [13] is that the beginning and the end of both compared sequences coincide. Unfortunately, in the query-by-humming task, this condition is rarely met, especially with respect to the ending point. The beginning is less problematic because most users typically sing from the beginning of a phrase [8]. The *length* of the query, on the other hand, is often unknown in advance, both in terms of absolute time units and with relation to the template. It is possible to control the absolute length of the query in the acquisition module, e.g. by stopping the recording session after x seconds. Even then, however, the assessment of the exact number of notes or phrases sung is impossible, mainly due to tempo differences between users. The query may therefore end anywhere within the course of the template, as illustrated in Fig. 1.

Fortunately, the DTW may deal with this case quite easily, as the endpoint of the warping path may effectively be searched for along the last row of the DTW matrix (or along the last column, if we also expect queries longer than templates). The real problem is, however, that it is now impossible to effectively use the global constraints, such as Sakoe and Chiba band (Fig. 1b), which are the *sine qua non* condition in the DTW indexing techniques [13, 35].

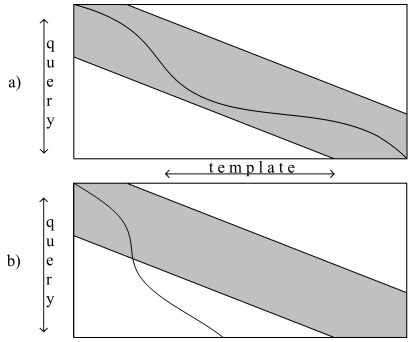


Figure 1. Optimal warping path in the DTW matrix (with Sakoe and Chiba band shown) for fixed-length query, where the query is much shorter than the template: a) The "fast singer" case – both the beginning and the ending points coincide; b) The "normal singer" case – the query ends in the middle of the template melody.

Relaxing the constraints (e.g. increasing the radius of the band) may help to incorporate more queries deviating from the diagonal, but on the cost of making the indexing less efficient. Hence, although in theory the lower bounding techniques guarantee no false dismissals, we arrive again at the trade-off between time-efficiency and retrieval rate. In fact, setting the proper constraints is always a matter of a compromise, but here the problem is much more severe, as even moderate tempo mismatch may lead to significant accumulation of deviations at the end of a query.

A real-life example – a template and the matching queries from Jang's dataset [10] – is presented in Fig. 2 where the ending point of each query, with respect to the template, has been determined on the basis of the optimal warping path in the DTW matrix. Fig. 2 reveals, that although within the fixed time of 8s most users managed to sing between two and three two-bar motifs (out of all four), there were also some "lazy singers" that did not manage to complete two motifs and some "fast singers" who completed the whole or almost the whole melody. With indexing, these queries may be easily lost, unless some extremely loose constraints were applied.

Let us note that the problem occurs even for optimal template length (e.g. from Fig. 2 we may conclude that this particular template is actually too long). In fact, in many datasets the templates tend to be much longer than the queries. For example, in Jang's [10] collection the template length varies from ca. 12 seconds up to over 5 minutes. Hence, determining the reasonable cutting point for the templates, prior to indexing, becomes a necessary preliminary step of processing.

It is important to note that this step cannot be done reliably without some form of melodic content analysis. Naturally we might try – for fixed query length of x seconds – to cut the templates to the same x seconds, assuming that the tempo of the template roughly corresponds to the mean tempo of the queries. However, we have no guarantee that this assumption is correct, which may obviously lead to suboptimal indexing results.

In the following section we present an automatic method for determining the cutting point of the template melody. The same method is also applied to each query to cut it at the point corresponding to the cutting point of the template.

Although it may seem not obvious, we should note that we also touch the problem of query transposition here. A user may sing the melody in any key, so it must be transposed to a reference key before matching, which is typically done by mean subtraction. However, the mean pitch of a melody obviously depends on the location of its ending point, which gives an additional motivation for trying to agree on a common cutting point among all the potentially matching melodies. The proposed method is based on a simple content-based filter, which enables the preliminary match of the lengths of the compared melodies and – in consequence – the practical use of the efficient indexing algorithms.

4. THE DISCRETE TOTAL VARIATION

An intuitive solution to our problem might be formulated as follows: given a perfect note segmentation of the audio files we could cut every melody after a fixed number of notes (the same for all melodies – templates and queries). This would guarantee the endpoint match for efficient indexing and the consecutive DTW would successfully deal with potential rhythm and tempo discrepancies. Unfortunately, while it is straightforward for MIDI-based templates, it is not so for the sung queries. The singer's imprecision on one hand and the specificity of a given pitch tracking algorithm on the other hand may lead to note segmentation errors that will make this approach unusable.

In our approach, instead of a crisp note segmentation we prefer to construct a soft measure of pitch value variability in time. In continuous case, for $p(\tau)$ representing the pitch value at time τ, we would define the following functional:

$$TV(t) = \int_0^t \left| \frac{d}{d\tau} p(\tau) \right| d\tau \ . \tag{2}$$

We may note that this definition, corresponding to the L_1 norm of the pitch signal derivative, may be seen as a one-dimensional version of Total Variation (TV) as introduced by Rudin [22, 23] in the image analysis and noise removal context. The one-dimensional Discrete Total Variation DTV may be defined as:

$$DTV(n) = \sum_{k=1}^{n} |p(k) - p(k-1)| \ , \tag{3}$$

where $p(k)$ denotes the k-th time frame of the pitch vector.

The fundamental property of DTV is that it accumulates pitch changes in the course of the melody, irrespective of the actual direction of the changes (Fig. 3). We may therefore set a threshold T_{DTV} for the accumulated pitch changes and cut all the compared melodies when they

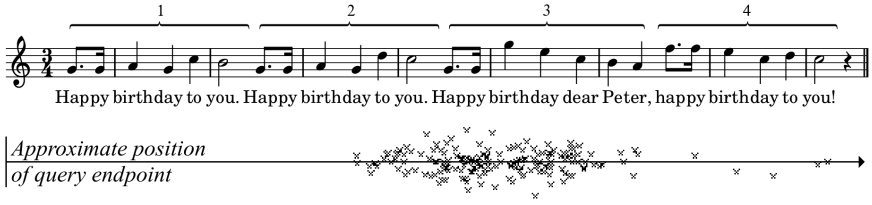


Figure 2. Example template from Jang's [10] collection (top) and the ending points of all the 170 matching queries (bottom).

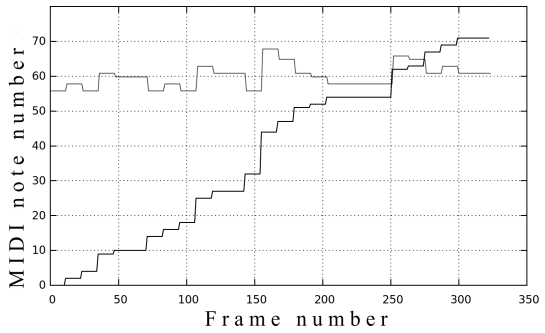


Figure 3. Pitch vector representation of the melody from Fig. 2 (top, light-grey) and the corresponding DTV sequence (bottom, dark-grey).

reach T_{DTV}, as follows:

$$p_c = [p(0), p(1), ..., p(n_c)] \,, \tag{4}$$

where p_c denotes the pitch vector reduced to the first n_c+1 values and where:

$$n_c = \min \{n \in N; DTV(n) \geq T_{DTV}\} \,. \tag{5}$$

The proposed DTV-based cutting scheme has some further properties that are relevant to the considered problem:

1. The DTV sequence is monotonically nondecreasing and it stays constant only within segments of fixed pitch. The latter fact means that the note lengths are basically ignored – only the number of notes and the span of the consecutive musical intervals (pitch difference) influence the increase of the value of DTV.

2. Ignoring the direction of the pitch changes means that the DTV is not unique. For example, ascending chromatic scale will yield the same DTV sequence as the descending one, assuming the same tempo and the same number of notes (diatonic scale would produce different results due to different ordering of whole steps and half steps).

3. DTV-based cutting leads to obtaining the melody representation robust to *glissandi*, occurring frequently in sung queries, where the pitch changes are "spread" over several consecutive time frames.

4. DTV-based cutting leads to obtaining the melody representation which is not robust to jitter and vibrato, which may be present within single notes, i.e. in segments of – otherwise constant – pitch.

Property 1 implies a fundamental fact that two versions of a melody, consisting of identical pitch sequences but with different rhythm and tempo will yield the same DTV sequence for corresponding notes. In particular, their representations obtained with Eq. 4 may have different number of frames, but they will basically represent the same *melodic content*.

Property 2 means that the DTV may be interpreted as a hash function which may occasionally return equal values for dissimilar input data. In fact, what we need is the opposite implication: the results for similar input must be also similar and – fortunately – this condition is generally fulfilled.

Property 3 is connected to an important advantage of the proposed method to ignore the slope of the pitch changes. When singing a musical interval, the second note is often reached after several frames of intermediate pitch values, as opposed to MIDI-based signals where the changes are instantaneous. This difference is well visible in Fig. 4 (top plots). It can be observed that despite the fuzzy note transitions in frames 30–33 and 51–59 of the query (plot **a)**), the obtained DTV sequences (Fig. 4, the bottom plots) are indeed similar. Therefore setting a given threshold value T_{DTV} in Eq. 5 would allow to obtain the similar melody sections both for the query and for the template, irrespective of the significant tempo discrepancy between the two. For example, for $T_{DTV} = 5$ both sequences would be cut at the onset of the 4th note.

Property 4 indicates a potential weakness of the proposed solution as jitter and vibrato are ubiquitous in pitch vectors of sung melodies. However, a popular and simple median filter, which is often used to pre-process the pitch vectors prior to further melody analysis, may be effectively applied here to suppress the minor pitch fluctuations. The example in Fig. 4 had been already filtered with median filter of 9^{th} order which had appropriately smoothed almost the whole signal, except of the small artefact in frames 86–88. As a comparison, Fig. 5 **a)** presents the original query. The dramatic distortion of the obtained DTV sequence may be assessed even better on the right plot (**b)**), where we see over twofold increase of the accumulated pitch changes for the unfiltered query with respect to the filtered one.

5. EXPERIMENTAL VALIDATION

In order to evaluate the usefulness of the proposed method in melody indexing, we performed tests on the well-

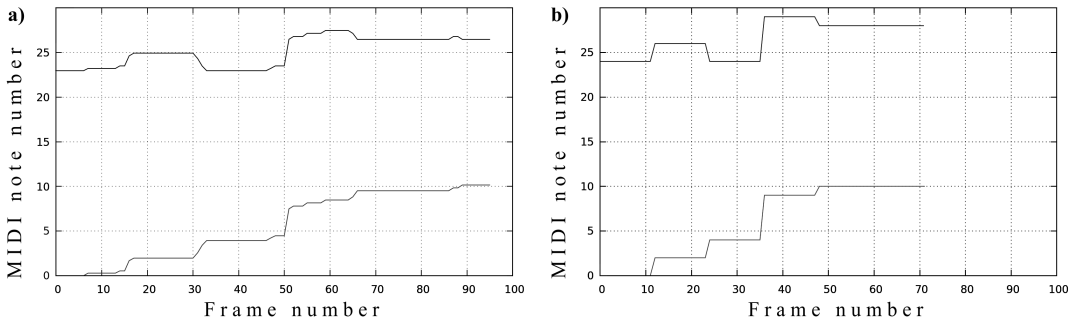


Figure 4. The first motif of the melody from Fig. 2: **a)** query; **b)** template. Top plots present the original pitch vectors (after transposition by 3 octaves down, for visualization purposes); the bottom plots show the DTV sequences.

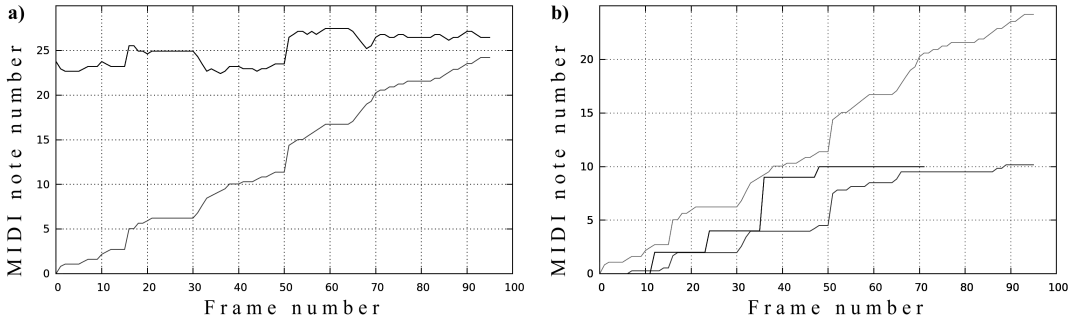


Figure 5. **a)** the same melody as in Fig. 4, but without the median filter; **b)** comparison of the obtained DTV sequences: query without median filter (light grey, top); template (black, middle); query after median filter (dark grey, bottom).

known, publicly available dataset, consisting of a collection of 48 popular songs (in the form of ground-truth MIDI files) to be matched against 4431 queries sung by about 200 users [10]. In order to increase the difficulty of the problem, the set of templates was artificially expanded, so that it contained – apart from the 48 ground-truth files – 4225 additional noise midi files from Essen collection [4].

Piecewise aggregate approximation was applied as a dimensionality reduction technique. Each template melody was represented as a point in 16-dimensional index space where R^*-tree was used as the spatial index. For each query melody the minimal bounding rectangle (MBR) was computed and used for k-NN search, according to [15]. Two quantities were measured during the tests: the CPU time of computation and the number of correctly recognized queries. The baseline results obtained by a non-indexing system, computing the DTW match between every query and every template, were: 4211 out of 4431 queries recognized (95.03%) in 48h 55m 28s.

For the indexing tests several melody cutting strategies were applied and the corresponding results have been presented in Fig. 6. All the queries in the test database are of the same length of 8s. Our first attempt was therefore to apply the straightforward approach based on cutting the templates to the same 8s (250 frames with a step of 32 ms) prior to indexing. This in fact yields the optimal template length also in terms of the melody content because, as we have found, there is no bias towards faster or slower queries in the test database, i.e. the mean tempo of each query is basically the same as the tempo of the corresponding template.

However, it appears that even in this optimal setting, our DTV-based cutting scheme may increase the recognition rate with respect to the fixed template cutting point (for the same number of nearest neighbors). For example, the recognition rate obtained for the fixed-length templates with 1500 nearest neighbors could be obtained for the templates cut on the basis of their DTV with 1100 nearest neighbors, which means ca. 25% gain in the computation time (Fig. 6).

The key point in the effective application of the DTV is setting the proper threshold T_{DTV}. Too low value leads (Fig. 6, $T_{DTV} = 20$) to extracting and indexing very short melody fragments, which means that they contain few notes and hence many templates may even become indistinguishable from each other. Moreover, for short queries the lower bounding measure often happens to be zero which prevents establishing the right order of the results.

On the other hand, too high T_{DTV} threshold means that many queries will not reach it before their "hard" cut point (8s in our case). This problem will not occur in the case of the templates, because they are usually much longer. As a result, after the cutting procedure the templates will be generally longer than the queries and they will also contain much more melodic material, which will make the indexing ineffective.

As a partial remedy, we propose to use a simple condition, limiting the absolute length of the templates to the

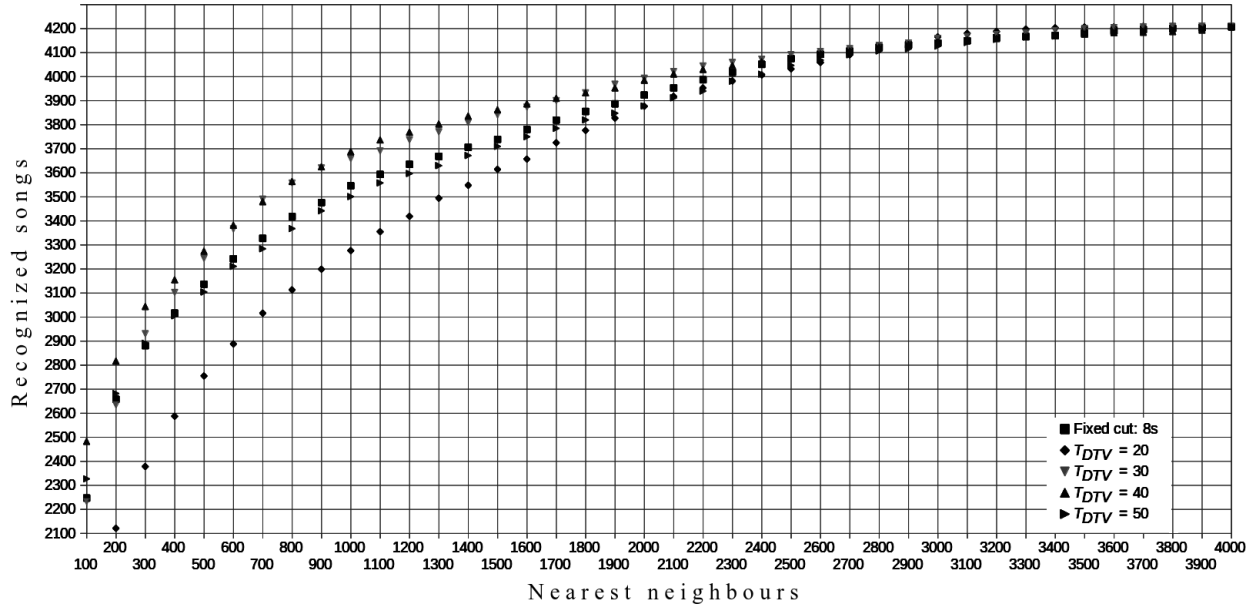


Figure 6. Top-ten results for various cutting point setting

absolute length of the queries:

$$\hat{n}_c = \min\{n_c, |q|\} \tag{6}$$

where n_c is given by Eq. 5 and $|q|$ is the fixed length of the queries. In this way the template lengths are limited similarly as the queries. We should note that the only problem which may occur here is when some monotonous, not-much-varied melodies are sung by a "fast singer", because these queries will contain more melodic material than – then prematurely cut – template. This problem generally cannot be avoided without *a priori* knowledge on the correct classification of the query, but it appears not to have much impact on the recognition rate. The results in Fig. 6 have been obtained with template length limiting (Eq. 6) and as we may see they are stable in a quite broad range of the threshold values ($T_{DTV} = 30$, $T_{DTV} = 40$). Going beyond this optimal range ($T_{DTV} = 50$) deteriorates the recognition but still the obtained results are only slightly worse that the fixed cutting point approach.

6. CONCLUSION

In this work we have introduced a new method of determining the optimal cutting point for melody comparisons, based on Discrete Total Variation of the melody pitch vector. Our solution is fast to compute and it yields useful information about the melody, which enables to effectively apply DTW indexing strategies, introduced in [13]. We demonstrated the usefulness of the proposed solution for the indexing task on a known database, designed for testing Query-by-Humming systems. It should be noted, however, that the method has potentially much broader application area. In particular, much more significant gain may be expected in less constrained, on-line QbH systems, especially when more relaxed limits of the query length are used, and/or when greater singing tempo discrepancies may be expected. The ability to find the appropriate melody length in a fast way, without detailed note-based analysis and without computationally expensive DTW is an advantage which may simplify and accelerate many content-based music information retrieval tasks.

7. REFERENCES

[1] http://www.music-ir.org/mirex.

[2] N. Adams, D. Marquez, and G. Wakefield. Iterative Deepening for Melody Alignment and Retrieval. In *6th Int. Conf. on Music Information Retrieval, ISMIR 2005*, pages 199–206, 2005.

[3] G. Al-Naymat, S. Chawla, and J. Taheri. SparseDTW: A Novel Approach to Speed Up Dynamic Time Warping. In *Proc. of the 8th Australasian Data Mining Conf.*, pages 117–127, 2009.

[4] http://www.esac-data.org, 2009.

[5] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *SIGMOD1994*, pages 419–429. ACM, 1994.

[6] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith. Query By Humming – Musical Information Retrieval in an Audio Database. In *Proc. of the 3rd ACM Int. Conf. on Multimedia*, pages 231–236, 1995.

[7] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD 1984*, pages 47–57. ACM Press, 1984.

[8] S. Huang, L. Wang, S. Hu, H. Jiang, and B. Xu. Query by humming via multiscale transportation distance in

random query occurrence context. In *IEEE Int. Conf. on Multimedia and Expo*, pages 1225–1228, 2008.

[9] F. Itakura. Minimum Prediction Residual Principle Applied to Speech Recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 23(1):67–72, 1975.

[10] http://mirlab.org/dataSet/public/MIR-QBSH-corpus.rar, 2009.

[11] J.-S. R. Jang and H.-R. Lee. Hierarchical filtering method for content-based music retrieval via acoustic input. In *Proc. of the 9th ACM Int. Conf. on Multimedia*, pages 401–410, 2001.

[12] F. K.-P. Chan, A. W.-C. Fu, and C. Yu. Haar Wavelets for Efficient Similarity Search of Time-Series: With and Without Time Warping. *IEEE Trans. on Knowl. and Data Eng.*, 15(3):686–705, 2003.

[13] E. Keogh. Exact indexing of dynamic time warping. In *Proc. of the 28th Int. Conf. on Very Large Data Bases*, VLDB '02, pages 406–417, 2002.

[14] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3(3):263–286, 2001.

[15] E. Keogh and C. A. Ratanamahatana. Exact Indexing of Dynamic Time Warping. *Knowl. Inf. Syst.*, 7(3):358–386, 2005.

[16] E. J. Keogh. Efficiently Finding Arbitrarily Scaled Patterns in Massive Time Series Databases. In *PKDD*, volume 2838 of *Lecture Notes in Computer Science*, pages 253–265. Springer, 2003.

[17] E. Lau, A. Ding, and J. Calvin. MusicDB: A Query by Humming System. Final Project Report, Massachusetts Institute of Technology, USA, 2005.

[18] R. Macrae and S. Dixon. Accurate Real-time Windowed Time Warping. In J. S. Downie and R. C. Veltkamp, editors, *Proc. of the 11th Int. Society for Music Information Retrieval Conf., ISMIR 2010*, pages 423–428, 2010.

[19] D. Mazzoni and R. B. Dannenberg. Melody Matching Directly From Audio. In *2nd Annual Int. Symposium on Music Information Retrieval, ISMIR 2001*, pages 73–82, 2001.

[20] R. J. McNab, L. A. Smith, I. H. Witten, C. L. Henderson, and S. J. Cunningham. Towards the digital music library: tune retrieval from acoustic input. In *Proc. of the 1st ACM Int. Conf. on Digital Libraries*, DL '96, pages 11–18, 1996.

[21] D. Parsons. *The Directory of Tunes and Musical Themes*. S. Brown, 1975.

[22] L. A. Rudin. *Images, Numerical Analysis of Singularities and Shock Filters*. PhD thesis, 1987.

[23] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear Total Variation Based Noise Removal Algorithms. *Phys. D*, 60(1-4):259–268, November 1992.

[24] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

[25] S. Salvador and P. Chan. FastDTW: Toward accurate dynamic time warping in linear time and space. In *3rd Workshop on Mining Temporal and Sequential Data*, 2004.

[26] J. Shifrin, B. Pardo, and W. Birmingham. HMM-Based Musical Query Retrieval. In *Joint Conf. on Digital Libraries. 2002. Portland, Oregon*, pages 295–330, 2002.

[27] B. Stasiak. Follow That Tune - Adaptive Approach to DTW-based Query-by-Humming System. *Archives of Acoustics*, 39(4):467 – 476, 2014.

[28] R. Typke, P. Giannopoulos, R. C. Veltkamp, F. Wiering, and R. van Oostrum. Using transportation distances for measuring melodic similarity. In *ISMIR 2003*, pages 107–114, 2003.

[29] R. Typke, F. Wiering, and R. C. Veltkamp. Transportation distances and human perception of melodic similarity. *Musicae Scientiae*, pages 153–181, 2007.

[30] A. L. Uitdenbogerd and J. Zobel. Manipulation of Music for Melody Matching. In *Proc. of the 6th ACM Int. Conf. on Multimedia*, pages 235–240. ACM, 1998.

[31] L. Wang, S. Huang, S. Hu, J. Laing, and B. Xu. An effective and efficient method for query by humming system based on multi-similarity measurement fusion. In *Int. Conf. on Audio, Language and Image Processing*, pages 471–475, 2008.

[32] L. Wang, Sh. Huang, Sh. Hu, J. Liang, and B. Xu. Improving searching speed and accuracy of query by humming system based on three methods: feature fusion, candidates set reduction and multiple similarity measurement rescoring. In *INTERSPEECH*, pages 2024–2027. ISCA, 2008.

[33] B.-K. Yi and C. Faloutsos. Fast Time Sequence Indexing for Arbitrary Lp Norms. In *Proc. of the 26th Int. Conf. on Very Large Data Bases*, VLDB '00, pages 385–394, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[34] B. Zhu and H. Liu. MIREX 2015 QBSH task: Tencent Bestimage's solution. pages 1–2, 2015.

[35] Y. Zhu and D. Shasha. Warping indexes with envelope transforms for query by humming. In *Proc. of the 2003 ACM SIGMOD Int. Conf. on Management of Data*, pages 181–192, 2003.

ELUCIDATING USER BEHAVIOR IN MUSIC SERVICES THROUGH PERSONA AND GENDER

John Fuller
University of Washington
fuller14@uw.edu

Lauren Hubener
University of Washington
lhubener@uw.edu

Yea-Seul Kim
University of Washington
yeaseul1@uw.edu

Jin Ha Lee
University of Washington
jinhalee@uw.edu

ABSTRACT

Prior user studies in the music information retrieval field have identified different personas representing the needs, goals, and characteristics of specific user groups for a user-centered design of music services. However, these personas were derived from a qualitative study involving a small number of participants and their generalizability has not been tested. The objectives of this study are to explore the applicability of seven user personas, developed in prior research, with a larger group of users and to identify the correlation between personas and the use of different types of music services. In total, 962 individuals were surveyed in order to understand their behaviors and preferences when interacting with music streaming services. Using a stratified sampling framework, key characteristics of each persona were extracted to classify users into specific persona groups. Responses were also analyzed in relation to gender, which yielded significant differences. Our findings support the development of more targeted approaches in music services rather than a universal service model.

1. INTRODUCTION

Commercial music streaming services represent the fastest growing sector of the music recording industry. User studies from the field of Music Information Retrieval (MIR) that can be used to guide the strategic development of these streaming services as well as new methods for the navigation of large music collections have been increasing. Several studies have pointed out that the large majority of research in MIR systems has been focused on the evaluation of system accuracy and performance, creating a gap in user-centric research [19, 27, 29]. Although MIR studies are increasing in number, many tend to employ a qualitative approach and derive findings from the investigation of a limited number of users [29]. Understanding users' experiences with MIR systems, and particularly in relation to commercial music streaming services, can benefit from the adoption of more quantitative methods of analysis in addition to these qualitative assessment techniques.

This study aims to triangulate findings from prior qualitative MIR user studies adopting a quantitative approach with a larger sample to verify the findings and provide complementary insights. In particular, we conducted a follow-up study to further explore the findings of Lee and Price [18]. They presented seven different personas surrounding the use of commercial music services, derived from interviewing and observing 40 users who regularly use at least one commercial music service. Our study aims to test the applicability of the previously defined personas with a larger user population, as the results of the original study involved a relatively small sample. In this study, 962 users were surveyed in order to capture characteristics of the individuals' music listening habits based on their preferred commercial music services, and the data were analyzed in connection to the results derived from the principal study. In particular, this study seeks to elaborate on the previous findings and address the following questions:

RQ1: How similar or distinct are the seven personas when generalized to a larger stratified user sample?
RQ2: How similar or different are the persona distributions for different commercial streaming services?
RQ3: Is there a significant difference between genders with regard to their persona distribution?

2. RELEVANT WORK

2.1 Users of Music Streaming Services

The emergence of innovative music streaming services has raised the awareness and desire, both in academia and industry, for a ubiquitous system that seamlessly allows for the search, retrieval, and recommendation of music. However, a deeper understanding of the user is crucial for the development of more personalized and context-aware systems that will meet the users' needs in a wide variety of situations [27].

There have been a few studies focusing on understanding the reasons for the popularity of music steaming services such as Spotify or YouTube (e.g., [13, 21]) based on survey data or specifically measuring the quality of the music recommendations provided by commercial services such as Apple iTunes Genius (e.g., [3]). From the studies that have focused on evaluating users' experiences with music streaming services, a number of trends have emerged. Based on a large-scale survey, Lee and Waterman [20] determined an increased consumption of music on mobile devices, an increased desire for the serendipitous discovery of music and the option of customization in their listening experiences. Lee and Price [18] found that music streaming services are often perceived as "good enough" for users' purposes, and many individuals use a variety of services to accommodate needs across various contexts, illustrating Bates' "berrypicking" search behavior. A qualitative study conducted by Laplante and Downie [16] exploring the information-seeking behavior of young

© John Fuller, Lauren Hubener, Yea-Seul Kim, Jin Ha Lee. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** John Fuller, Lauren Hubener, Yea-Seul Kim, Jin Ha Lee. "Elucidating User Behavior in Music Services Through Persona and Gender", 17th International Society for Music Information Retrieval Conference, 2016.

adults in the discovery of new music found a preference for informal channels (e.g. friends and family) as well as a distrust of experts. It was also revealed that the act of music seeking is strongly motivated by curiosity as opposed to actual information needs, which helps illustrate why browsing is an often-employed technique. Ferwerda et al. [10] explore how individuals' personality traits relate to their explicit choice of music taxonomies, and propose the creation of an interface that can adapt to users' preferred methods of music browsing. Ethnographic and observational research into music collection by Cunningham et al. [9] found that users create an implicit organization to their own music collections (digital or physical) without being fully conscious of it, and are reluctant to remove or delete songs even if they have not listened to them in months or years.

In addition, a growing number of researchers have come forward to suggest more comprehensive user models in MIR systems. Cunningham et al. [8] investigated how the factors of human movement, emotional status, and the physical environment relate to musical preferences, and based on the findings, present a model for playlist creation. In addition, assuming a mobile-based consumption of music, systems seeking to match the pace of someone in movement have also been proposed (e.g., [22, 25]). These typically aim to correlate the music played to the user's heartbeat, though most of the systems proposed would require additional context-logging hardware. A study by Baltrunas et al. [2] outlines a context-aware recommendation system for music consumption while driving by taking into account eight contextual facts including mood, road type, driving style, and traffic conditions. There have also been a few studies in which users were categorized into different types of users or personas: Celma [6] identified four different types of music listeners (i.e., savants, enthusiasts, casuals, and indifferents) representing different degrees of interest in music, and Lee and Price [18] derived seven personas, hypothetical archetypes of users representing specific goals and behavior [7], from empirical user data to help understand the different needs they have regarding the use of music services, which our study is building upon.

There is notably less research dedicated to exploring how various users search for, interact with, and listen to music digitally since the widespread growth of commercial music streaming services. This paper will elaborate on how users listen to music and interact with streaming services through the application of user personas developed in the principal study [18] in order to identify behavioral differences, preferences, and varying MIR goals.

2.2 Gender and Musical Interactions

Several researchers have studied the impact of demographic characteristics such as age [5, 15] and nationality [11, 14] on musical interactions. These studies have found that both age and cultural background prove to be significant factors in individuals' music perceptions and preferences. To a lesser extent, the influence of gender in music studies has been explored, often in the field of ethnomusicology [23, 24] and music education [1, 26]. O'Neill, a leading researcher in music and education, has found striking differences in boys' and girls' preferences for music and musical activities [26]. Her study of 153 children aged 9 to 11, explored the extent to which boys' and girls' preferences are a product of gender-stereotyped associations. She found that girls showed a significantly stronger proclivity for the piano, violin, and flute, whereas boys expressed preference for the guitar, drums, and trumpet. Also each gender had similar ideas regarding which instruments should not be played by boys or girls. The study supports the notion that gendered perspectives regarding music are developed early and could have a lasting impact on the way different genders go about seeking musical information. A few studies have focused on the general listening preferences and music processing of the two genders. Koelsch et al. [12] studied the differences between genders in the processing of music information. They discovered that early electric brain activity occurs bilaterally in females, and with right hemispheric predominance in males, supporting the claim that differences between genders in the processing of auditory information go beyond the linguistic domain. LeBlanc et al. [17] found the variables of age, country, and gender to all be significant factors in individuals' music listening preferences, but determined each of these variables to be involved in complex interactions with other variables. Though the effectiveness of the age and gender variables was confirmed, they did not perform the same way in each country, and therefore should be explored in relation to cultural context. In the context of MIR, few studies explored gender as a variable while examining the needs and behavior of music listeners. In our study, we specifically wanted to investigate whether there is a significant difference between genders with regard to their use of music streaming services, to complement what we already know about gender differences in other musical interactions.

3. STUDY DESIGN AND METHOD

3.1 Background

The principal study [18] was conducted in 2015 in order to gain insights into how users' personalities and characteristics affect their interactions with and preferences for particular MIR systems. In the study, 40 subjects participated in semi-structured interviews regarding how they evaluate music services and think-aloud sessions during which subjects described and narrated their actions as they used their preferred music service. A card sorting activity recording subjects' comments, actions, and behaviors throughout the process derived seven personas: Active Curator (AC), Addict (AD), Guided Listener (GL), Music Epicurean (ME), Music Recluse (MR), Non-Believer (NB), and Wanderer (WA). The typical behaviors and tendencies for each of these personas regarding their use of commercial music services are described in detail in [18].

3.2 Method

The online survey was developed using SurveyMonkey, an online survey tool, and responses were collected during April and May 2015, before the demise of streaming services Grooveshark and Rdio and before the release of Apple Music. The survey contained a total of 26 questions

pertaining to individuals' music listening habits and behaviors, their interactions with MIR systems, and preferred music streaming services. The questions were developed in connection to the results of the principal study [18] in order to test the applicability of the seven personas with a larger user population. A majority of the survey questions and response options targeted one or more of the previously defined personas with the intention to classify a user's response to a particular persona (further discussed in Section 4.2). A limited amount of demographic information was also collected, including age and gender.

Distribution methods to recruit participants included an open call for individuals age 14 or older via University of Washington departmental listservs and posts to online message boards such as Reddit. A total of 1,028 responses were collected, of which 962 complete responses were used in the analysis. The survey responses were used to generate user profiles consisting of a list of behaviors and preferences exhibited when interacting with music streaming services. The 26 survey questions were translated into 32 variables. The data were then analyzed in RStudio and Excel environments.

4. FINDINGS AND DISCUSSION

4.1 Demographic Information

The average age of respondents was 23.4 (SD = 7.8). Sixty-one respondents did not report an age, and four respondents under the survey participation age of 14 were excluded in the analysis. The breakdown of gender distribution were as follows: 57.3% were male, 36.4% were female, 1.7% selected "other," and 4.7% did not specify their gender. 78.3% of respondents described themselves as White or Caucasian, 7.0% described themselves as Asian, 3.7% described themselves as Multiracial, 2.4% described themselves as Hispanic, and 2.2% described themselves as Black, respectively. 5.5% of respondents did not indicate or specify an ethnic affiliation. Responses were collected from 54 countries. The five countries with the most responses were: the United States (63.52%), Canada (7.1%), the United Kingdom (3.8%), Australia (2.6%), and Germany (2.1%). 5.7% of respondents did not indicate or specify a country of residence.

4.2 Filtering Method

Survey respondents were asked a series of questions regarding their music listening habits, behaviors, and actions. Questions pertaining to individuals' behavior and actions when searching for and listening to music were pivotal in determining and classifying respondents to specific personas. To classify respondents into personas, a filtering method using a combination of question-response(s) was applied to the entire sample. For each persona, if the participant selected the primary question and response pair and one of two secondary question and response pairs, he or she was classified as a respondent exhibiting that particular persona. These primary and secondary question-response pairs are summarized in Table 1.

Active Curator	Primary: Regularly curates and listens to playlists
	Secondary: Makes playlists more than once a month OR Cares about organizing collection and willing to spend time for it
Addict	Primary: Spends more time searching rather than browsing for music
	Secondary: Likes a few songs and listens to them over and over OR Most likely to listen to a song repeatedly when they hear a new song they enjoy
Guided Listener	Primary: Most likely to use a new song they like to generate recommendations of similar songs
	Secondary: Finds recommendations generated by a streaming service most appealing OR Wants to engage with a music streaming service minimally
Music Epicurean	Primary: Likely to seek out the whole album or search online to learn more about a song they like
	Secondary: Very or somewhat interested in an artist's relationship to other artists, genres, or music scenes OR Purchases music approximately once a month or more
Music Recluse	Primary: Not at all or not very likely to recommend a song to a friend
	Secondary: Listens to music they consider a "guilty pleasure" often to all the time OR Not willing or reluctant to share personal information/listening history with a streaming service
Non-Believer	Primary: Does not trust music recommendation generators and prefers finding music through other methods
	Secondary: Does not think an app or service can pick music they would like OR Does not use the social features of streaming services
Wanderer	Primary: Finds and listens to music from many sources and is always looking for something new
	Secondary: Spends more time browsing rather than searching for music OR Takes satisfaction in discovering new artists others have not heard of

Table 1. Filtering mechanism for assigning personas.

4.3 Similarity among Personas

Through the filtering method used to create sample subsets for each persona, 155 respondents (15.4%) were classified as Active Curators, 184 (18.2%) as Addicts, 47 (4.7%) as Guided Listeners, 121 (12.0%) as Music Epicureans, 119 (11.8%) as Music Recluses, 120 (11.9%) as Non-Believers, and 263 (26.1%) as Wanderers. The filtering method used to classify respondents did not return mutually exclusive results in all cases, leading to a number of respondents to be classified with multiple music personas. This is consistent with the observation of Lee and Price that "any user may exhibit a combination of these personas as they are not mutually exclusive (p. 478)" [18]. In total, 732 respondents (71.2%) were identified as exhibiting one or more personas. Of the total number of responses, 493 respondents (67.3%) were classified as exhibiting one persona, 204 respondents (27.9%) as two personas, 32 respondents (4.4%) as three personas, and only 3 respondents (0.4%) were classified as having four personas. These results suggest that enough distinctness exists between at least some of these personas that they tend to not be exhibited by the same person. At the same time, it indicates that

a closer relationship exists between particular user personas than with others, and how a respondent's classification as exhibiting more than one persona may represent a fluctuating or shifting of persona association, depending on the user's momentary listening needs or context. It is noteworthy that 28.8% of respondents did not show a distinct persona; rather, they exhibited a range of different behaviors related to a number of different personas.

We calculated the Jaccard similarity coefficient to measure the similarity among personas (Table 2). This statistic represents the degree of overlap between two sets of data ranging from 0 to 1 [28]. Guided Listener and Music Recluse have the largest overlap among classified survey respondents, while the Active Curator and Wanderer, and the Addict and Wanderer have the least.

	AC	AD	GL	ME	MR	NB	WA
AC							
AD	0.61						
GL	0.73	0.68					
ME	0.66	0.58	0.77				
MR	0.67	0.67	0.79	0.69			
NB	0.67	0.67	0.78	0.69	0.76		
WA	0.43	0.43	0.63	0.61	0.54	0.55	

Table 2. Similarity among personas measured by Jaccard similarity coefficient.

We attempted to identify the reasons for these patterns by comparing these coefficient values, examining how the personas were originally defined, and reviewing the qualitative data reported in the principal study including direct user quotes. In regards to the personas with the largest overlap, the major similarity between the two is that they prefer not to engage or interact with the service very much. Guided Listener is passive and minimally invested when it comes to engagement with a music service. Music Recluse also tends to limit their engagement with the system by not actively sharing their information or using social features. The difference between these personas is that the Guided Listener trusts a music service enough to let the service select music on their behalf while the Music Recluse has little to no interest in sharing their music preferences or listening history with a music service because they are private listeners. The fact that the coefficient values between the Non-Believer and Music Recluse (0.76) and between the Guided Listener and Non-Believer (0.78) also tend to be high is noteworthy. This may be stemming from the commonality among these personas that they do not like to share their personal information although their reasons may be different (e.g., lack of interests, distrust with recommendation algorithms, privacy concerns).

When looking at personas with the least overlap and their characteristics, the level of willingness to explore and the level of engagement with their own collection seem to be the core reasons for this difference. According to the data reported in the principal study [18], the Wanderer is primarily concerned with finding new music across platforms and genres and is generally exploratory in their music listening while the Addict tends to repeatedly listen to a few songs they know and like. Also the Active Curator shows a high level of engagement with their own music collection which takes the form of playlist creation and collection organization/management whereas the Wanderer is focused on new music discovery which often occurs outside of their own collection.

4.4 Persona Classification and Preferred Services

When examining the distribution of the seven user personas across the most popular streaming services, several notable trends emerge. Personas were assigned to respondents using the filtering method described above in Section 4.2, and respondents' preferred service choice was determined using their responses to the survey question asking for the primary streaming service they use. Overall, the three most popular music streaming services selected by survey respondents were: Spotify (28.8%), YouTube (25.2%), and Pandora (17.2%). These were followed by iTunes (6.0%), SoundCloud (5.7%), and Google Play (4.2%), and several other services that were selected by less than 2% of respondents. We conducted a chi-square analysis to identify statistically significant differences between the persona distribution across the three most popular services (i.e., Spotify, YouTube, and Pandora) based on respondents' stated preference of primary streaming service.

When looking at those respondents who selected Spotify as their primary service (Figure 1), the Active Curator persona had the greatest representation with 28.1%, much more than other two services (X^2=28.37, df=2, p=0.000). Spotify includes a range of features making it easy for users to save songs for playlist creation as well as discover new music based on their listening history, which are design elements that the Active Curator would find valuable. Conversely, the Guided Listener persona was under-represented in its indications of a preference for Spotify, as the service itself requires a certain amount of curation to take advantage of its features (e.g. making an account, importing one's library, and understanding the robust interface) that the user is presented with upon first interacting with the service. For the less engaged personas, the initial process of familiarizing oneself with Spotify may not be worth the effort.

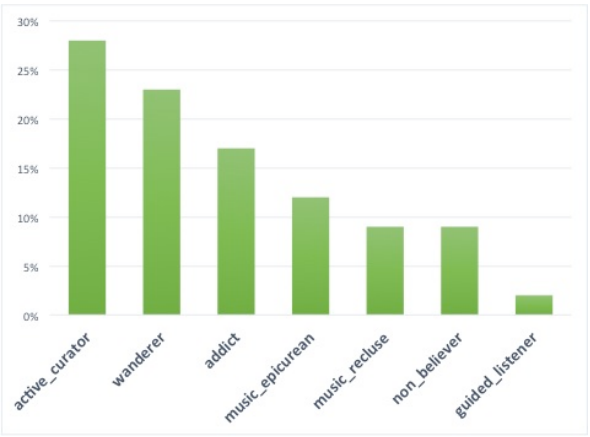


Figure 1. Persona distribution of Spotify users.

For those respondents who selected YouTube as their primary service (Figure 2), the Addict persona had the greatest representation with 24.1%, significantly different

from other services (X^2=13.40, df=2, p=0.001). Given that YouTube is an ideal platform for known-item searches, and allows for the easy replay of videos and songs, in the context of the Addict persona's user needs, YouTube would appear to be a realistic primary service choice. Music Epicureans were also most likely to choose YouTube as their primary service. The staggering amount of content, including concert footage, interviews, and rare tracks, often not available on any other services, allows for the endless discovery of new musical content. The ability to navigate through an extensive range of content is an appealing quality of the service for even the Non-Believer, which, although a small sample of the survey, was most likely to choose YouTube as their primary service. Conversely, the heavily self-directed nature of YouTube is an unattractive option to the Guided Listener, who was the least likely of any persona to choose YouTube as their preferred service.

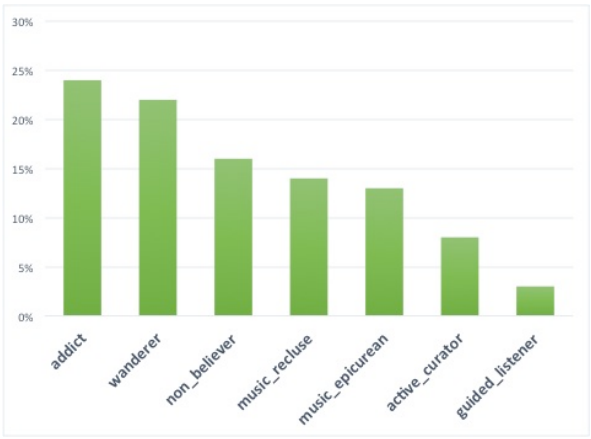


Figure 2. Persona distribution of YouTube users.

For those respondents who selected Pandora as their primary service (Figure 3), the Wanderer persona had the greatest representation with 29.7% (X^2=1.01, df=2, p=0.601), although the difference with the other two services was not significant. The most notable difference was the proportion of Guided Listeners (X^2=25.17, df=2, p=0.000). Overall, Guided Listeners were an underrepresented group in the sample, comprising only 4.7% of our population. While the respondents classified as Guided Listeners represented only 13.3% of those individuals that indicated Pandora as their preferred music service, Pandora was the primary service selected by Guided Listeners at 46.8%. Compared with other music services, Pandora provides a limited number of user features and will recommend and play content closely aligned to the user's "seed" artists. The user may be less likely to discover new or unexpected songs or artists but will also not have to interact with the service extensively during use. Because of these characteristics, Lee and Price [18] also expected that the Guided Listener persona would be the majority persona for Pandora which was indeed the case.

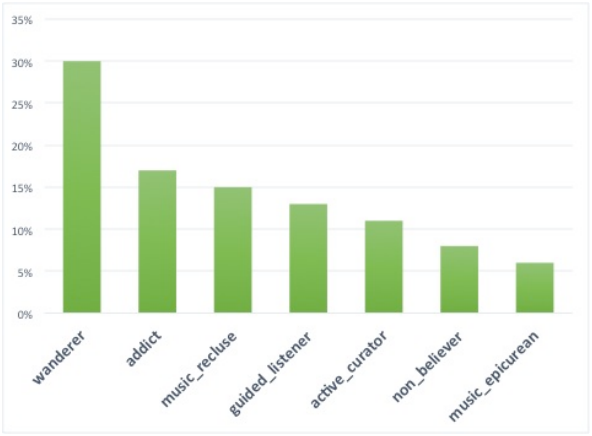


Figure 3. Persona distribution of Pandora users.

4.5 Gender Differences

Chi-squared tests revealed statistically significant differences in the way that users of different genders categorize their interactions with music services. When asked to indicate behaviors that described their music listening habits (X^2=30.40, df=4, p=0.000), 31.4% of males said they liked to listen to an album from start to finish, as opposed to only 16.4% of females. Females, on the other hand, were more likely to say that they enjoy listening to songs and artists from many different sources and are always looking for something new (32.6% for F, 30.7% for M). When asked what they typically do after hearing a new song that they enjoy (X^2=14.48, df=4, p=0.006), males were most likely to answer that they seek out the album the song came from (23.7% for F, 32.4% for M), while females were most likely to respond that they listen to the song over and over (34.1% for F; 23.5% for M).

Participants were also asked whether they find themselves more often searching or browsing for music (X^2=12.35, df=2, p=0.002); 48.9% of females identified themselves as known-item searchers and an equal share of 25.5% identified themselves as browsers and as spending equal amounts of time both searching and browsing. On the other hand, males' responses were more evenly distributed; 37.6% identified themselves as searchers, 33% browsers, and 29.4% responded that they spend approximately equal amounts of time doing both.

The tendency of females to identify as searchers is reflected in the classification of the Addict persona, which frequently relies on known-item searching in music seeking, encompassing 22.1% of females versus 15.8% of males (X^2=6.23, df=1, p=0.013). Females also more often identified as Music Recluse (X^2=5.16, df=1, p=0.023). The higher percentage of male browsers as opposed to female browsers supports the gender distribution of the Music Epicurean (8.9% for F, 14.6% for M; X^2=7.01, df=1, p=0.008) and Non-Believer (7.4% for F, 14.7% for M; X^2=12.28, df=1, p=0.000) personas. Table 3 shows the overall distribution of personas by gender. This pattern of males as the more exploratory gender in their music-seeking behavior is also exhibited in the breakdown of preferred music services. When asked to indicate their most preferred service, the top three services (Spotify, YouTube, and Pandora) accounted for 67.8% of males' top

selection, whereas 77.6% of females named one of these top three services as their most preferred one. Overall, males selected 18 different options, while females chose 14.

Persona	Female		Male		X^2	*df*	*p*
Active Curator	69	17.5%	82	14.2%	1.94	1	0.163
Addict	87	22.1%	91	15.8%	6.23	1	0.013
Guided Listener	20	5.1%	24	4.2%	0.46	1	0.499
Music Epicurean	35	8.9%	84	14.6%	7.01	1	0.008
Music Recluse	57	14.5%	56	9.7%	5.16	1	0.023
Non-Believer	29	7.4%	85	14.7%	12.28	1	0.000
Wanderer	97	24.6%	155	26.9%	0.61	1	0.433
Total	394	100.0%	577	100.0%	-	-	-

Table 3. Persona distribution by gender.

Overall, males seem to exhibit more exploratory nature in their music discovery endeavors. When participants were asked whether they take satisfaction in finding or discovering new artists that few people are aware of, 78% of males answered affirmatively versus 67.9% of females (X^2=12.16, *df*=1, *p*=0.000). Males also indicated that they more frequently curate playlists, with 33.9% of males saying that they like to make playlists at least once a week, compared to 23.9% of females (X^2=11.06, *df*=3, *p*=0.011). Males were also more likely to claim that listening to music occupies their full attention (14.2% for F, 21.2% for M; X^2=7.55, *df*=1, *p*=0.006).

These findings suggest that it may be fruitful for developers of commercial music streaming services to consider gender-specific approaches in the design and function of system features. Services targeted at males should account for the desire of discovering novel musical content, including robust features that consistently stimulate the encounter of unfamiliar artists and songs. Females are more likely to require features that allow for the easy replayability of content and support passive engagement.

There are also some notable similarities across genders. Neither males nor females tend to care much about the organization of their music collections. Of the respondents, 43.5% of males and 42.2% of females responded that they care only a little and are willing to spend minimal time organizing their collections (X^2=33.58, *df*=3, *p*=0.000). Additionally, both males (41.8%) and females (44.1%) responded that they prefer to get recommendations from those with similar tastes or listening habits rather than friends, family, music experts and curators, or streaming services (X^2=21.80, *df*=5, *p*=0.000). Therefore, services should incorporate measures of user profile and listening history similarity as a prominent feature for music recommendation. Interestingly, of all the respondents, 78.3% said that they were either somewhat likely or very likely to recommend a song or artist to a friend they thought would like it, but a mere 8.8% of all respondents said that they typically use the social features provided by a streaming service. The lack of use of social features was consistent across both genders (91.4% for F, 90.4% for M; X^2=0.059, *df*=1, *p*=0.808). This indicates that while users want to share music discoveries or new artists, they prefer to do so through channels not associated with the streaming service itself. This represents a potential area of innovation for commercial streaming services in evaluating how they incorporate social features into their platforms.

5. CONCLUSION AND FUTURE WORK

By testing the applicability of personas with a larger stratified user sample, we were able to determine the relative uniqueness of some personas and the closeness of others. In calculating the Jaccard coefficient, we found the Active Curator and Wanderer, and Addict and Wanderer personas to be the most dissimilar, while the Guided Listener and Music Recluse personas had the most overlap among the survey respondents. Going forward, we may consider reevaluating those less distinctive personas by further examining the overlapping characteristics of each, and redefine them accordingly.

When looking at the persona distributions for major commercial streaming services, patterns emerged between users' classified personas and their preferred services. While Spotify tends to draw in a high representation of more engaged personas like the Active Curator, it seemingly repels others, such as the Guided Listener persona. Similarly, YouTube, the service most preferred by Music Epicureans, was not popular among Guided Listeners, who instead prefer the more self-guided service, Pandora.

A further breakdown revealed significant differences between genders with regard to their persona distribution and preferred services. While the classification of the Addict persona skewed female, the Music Epicurean persona was male skewed. The Music Recluse and Non-Believer personas were also significantly different, with males more often identifying with the Non-Believer persona, and females comprising a higher representation of the Music Recluse persona. In regards to the preferred service distribution, a significant proportion of males and females reported Spotify and YouTube as their most preferred services. However, it was found that females are much more likely to favor Pandora, while males more often opt for peripheral services, such as Google Play and SoundCloud. The significant differences found between male and female users' preferences, characteristics, and expectations of music services suggest a need for further research. Future work will focus on obtaining a deeper understanding of the reasons for these gender differences and behaviors when engaging with music streaming services.

A revised model may be developed for classifying respondents to the defined user personas, incorporating the self-reported persona classification data from users. Asking users to identify themselves by the sets of traits represented by personas may help us further verify the validity of our filtering mechanism. In addition, further investigation through a qualitative study should be conducted to investigate those individuals that are classified into more than one persona to determine the factors at play, such as shifts across different contexts.

6. REFERENCES

[1] J. M. Abramo, "Popular music and gender in the classroom," Ed.D. dissertation, Teachers College, Columbia University, 2009.

[2] L. Baltrunas, M. Kaminskas, B. Ludwig, O. Moling, F. Ricci, A. Aydin, K.H. Lüke, and R. Schwaiger: "InCarMusic: Context-Aware Music Recommendations in a Car," *EC-Web*, Vol. 11, pp. 89-100, 2011.

[3] L. Barrington, O. Reid, and G. Lanckriet: "Smarter than Genius? human evaluation of music recommender systems," *Proc. of ISMIR*, pp. 357-362, 2009.

[4] M. J. Bates: "The design of browsing and berrypicking techniques for the online search interface." *Online Review*, Vol. 13, No. 5, pp. 407-424, 1989.

[5] A. Bonneville-Roussy, P. J. Rentfrow, M. K. Xu, and J. Potter: "Music through the ages: Trends in musical engagement and preferences from adolescence through middle adulthood." *Journal of Personality and Social Psychology*, Vol. 105, No. 4, pp. 703, 2013

[6] Ò. Celma: "Music recommendation and discovery in the long tail," Ph.D. dissertation, Dept. Information & Communication Tech., UPF, 2008.

[7] A. Cooper: *The Inmates Are Running the Asylum*, Sams, Indianapolis, 1999.

[8] S. Cunningham, S. Caulder, and V. Grout: "Saturday night or fever? context-aware music playlists," *Proceedings of Audio Mostly*, pp. 1-8, 2008.

[9] S. J. Cunningham, M. Jones, and S. Jones: "Organizing digital music for use: an examination of personal music collections," *Proc. of ISMIR*, pp. 447-454, 2004.

[10] B. Ferwerda, E. Yang, M. Schedl, and M. Tkalcic.: "Personality traits predict music taxonomy preferences," *Proc. of CHI EA '15*, pp. 2241-2246, 2015.

[11] X. Hu and J. H. Lee: "A cross-cultural study of music mood perception between American and Chinese listeners," *Proc. of ISMIR*, pp. 535-540, 2012.

[12] S. Köelsch, B. Maess, T. Grossmann, and A. D. Friederici: "Electric brain responses reveal gender differences in music processing," *NeuroReport*, Vol. 14 No. 5, pp. 709-713, 2003.

[13] S. Komulainen, M. Karukka, and J. Häkkilä: "Social music services in teenage life: a case study," *Proc. of OZCHI' 10*, pp. 364-367, 2010.

[14] K. Kosta, Y. Song, G. Fazekas and M. B. Sandler: "A study of cultural dependence of perceived mood in Greek Music," *Proc. of ISMIR*, pp. 317-322, 2013.

[15] P. A. Kostagiolas, C. Lavranos, N. Korfiatis, J. Papadatos, and S. Papavlasopoulos: "Music, musicians and information seeking behaviour: a case study on a community concert band," *Journal of Documentation*, Vol. 71, No. 1, pp. 3-24, 2015.

[16] A. Laplante and J. S. Downie: "Everyday life music information-seeking behaviour of young adults," *Proc. of ISMIR*, pp. 381-382, 2006.

[17] A. LeBlanc, Y. C. Jin, L. Stamou, and J. McCrary: "Effect of age, country, and gender on music listening preferences," *Bulletin of the Council for Research in Music Education*, pp. 72-76, 1999.

[18] J. H. Lee and R. Price: "Understanding users of commercial music services through persona: design implications," *Proc. of ISMIR*, pp. 476-482, 2015.

[19] J. H. Lee and R. Price: "User experience with commercial music services: an empirical exploration," *Journal of the Association for Information Science and Technology*, Vol. 67, No. 4, pp. 800-811, 2016.

[20] J. H. Lee and N. M. Waterman: "Understanding User Requirements for Music Information Services." *Proc. of ISMIR*, pp. 253-258, 2012.

[21] L. A. Liikkanen and P. Åman: "Shuffling Services: Current Trends in Interacting with Digital Music," *Interacting with Computers*, iwv004, 2015.

[22] H. Liu, J. Hu, and M. Rauterberg: "Music playlist recommendation based on user heartbeat and music preference," *International Conference on Computer Technology and Development*, pp. 545-549, 2009.

[23] T. Magrini: *Music and Gender: Perspectives from the Mediterranean*, University of Chicago Press, 2003.

[24] P. Moisala and B. Diamond: *Music and Gender*, University of Illinois Press, 2000.

[25] B. Moens, L. van Noorden, and M. Leman: "D-jogger: Syncing music with walking," *Sound and Music Computing Conference*, pp. 451-456, 2010.

[26] S. A. O'Neill and M. J. Boultona: "Boys' and girls' preferences for musical instruments: A function of gender?" *Psychology of Music*, Vol. 24, No. 2, pp. 171-183, 1996.

[27] M. Schedl A. Flexer: "Putting the user in the center of music information retrieval," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 385-390, 2012.

[28] R. Toldo and A. Fusiello: "Robust multiple structures estimation with j-linkage" *European Conference on Computer Vision*, pp. 537-547, 2008.

[29] D. M. Weigl and C. Guastavino: "User studies in the music information retrieval literature," *Proc. of ISMIR*, pp. 335-340, 2011.

EXPLORING THE LATENT STRUCTURE OF COLLABORATIONS IN MUSIC RECORDINGS: A CASE STUDY IN JAZZ

Nazareno Andrade
Universidade Federal de Campina Grande

Flavio Figueiredo
IBM Research - Brazil

ABSTRACT

Music records are largely a byproduct of collaborative efforts. Understanding how musicians collaborate to create records provides a step to understand the social production of music. This work leverages recent methods from trajectory mining to investigate how musicians have collaborated over time to record albums. Our case study analyzes data from the Discogs.com database from the Jazz domain. Our analysis examines how to explore the latent structure of collaboration between leading artists or bands and instrumentists over time. Moreover, we leverage the latent structure of our dataset to perform large-scale quantitative analyses of typical collaboration dynamics in different artist communities.

1. INTRODUCTION

Collaboration is a major component of musical creation. Examining who has collaborated in a record is a common method to understand their style, content, and process of creation. Collaborators leave a mark in the music, and may affect the style of the leading artists themselves. For example, the fact that Miles Davis collaborated with Charlie Parker in the beginning of his career can be seen as an important influence in the development of his style.

Looking at a larger picture, understanding the string of collaborators of a musician over his or her career is also prolific source of information to understand the career itself. Reusing the same example, it is possible to partly describe changes in Miles Davis' style in the 70s by describing how he changed the musicians recording with him. At the same time, similarities in the sequence of collaborators for two artists may denote similarities in the artists themselves. Complementarily, identifying common sequences of leading artists with which different instrumentists have performed also helps understanding how styles and communities of musical creation evolve.

From a quantative standpoint, collaboration patterns have often been studied through the use of methods from graph analysis to large-scale collaboration networks (e.g. [1, 9, 16, 18, 21]). However, although these methods provide valuable insights, they fail to focus on longitudinal views of collaboration. This way, they do not allow for examining patterns in collaboration trajectories.

This work leverages recent methods proposed for mining trajectories in object consumption to study collaborations trajectories among musicians. We use TribeFlow [7], a method recently shown to accurately and expressively discover latent spaces of consumption sequences in the Web domain [7]. Our work explores how this model can also be used to discover latent structures in the trajectories of musicians as they collaborate with the leading artists or bands in records. This exploration is done through a case study with Jazz records. Collaborators and musicians as extracted from the Discogs.com collaborative database of discographic information.

The rest of this paper is organized as follows. In the next section we discuss related work. An overview of the TribeFlow model is described in Section 3. This is followed by a description of our datasets in Section 4. Our main results are discussed in Sections 5 to 7. Section 5 discusses the latent trajectories (collaboration spaces) extracted with TribeFlow. Here, we discuss how the method extracts a semantically meaningful latent representation of our datasets. In Section 6 we compare the collaboration spaces of different artists. Section 7 discusses how artists move between collaboration spaces over time. Finally, in Section 8 we conclude the paper.

2. RELATED WORK

Our cultural products, music being no exception, are strongly tied of our social interactions, and in particular to the dynamics of such interactions. Realizing the importance of understanding networks of interacting collaborators, various research efforts have looked into large-scale creation, dissemination and curation of information by groups of individuals [2,3,5,10–12,15,17,20,21]. Some efforts have also specifically focused on understanding musical recordings as a collaborative effort [1,8,9,16,18,19]. Nevertheless, much less attention has been given to the dynamics of collaborations trajectories as we do.

With regards to musical production, very recently Bae et. al. [1] looked into the network properties and community structure of the ArkivMusic [1] database. This database, contains meta-data on classical music records. The authors looked into complex network properties such as power-law distributions and the small world effect [5] that exist

[1] http://www.arkivmusic.com

© Nazareno Andrade, Flavio Figueiredo. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).
Attribution: Nazareno Andrade, Flavio Figueiredo. "Exploring The Latent Structure of Collaborations in Music Recordings: A Case Study in Jazz", 17th International Society for Music Information Retrieval Conference, 2016.

in such networks. Similar studies based on complex networks has also been performed for Brazilian music [9, 18], as well as contemporary popular music [16].

Regarding the community structure of collaboration networks, again the work of Bae et. al. [1] performed an analysis of the ego-network of contributors. Their analysis uncovered strength of social connections from performers of classical musical. In a similar note, Gleiser and Danon also looked into communities of jazz musicians [8].

Our approach in this paper complements the above by viewing collaborations as dynamic trajectories, not static networks as was done previously. This novel way of looking into collaboration employs recent advances in trajectory mining [7]. Viewing collaborations as trajectories can aid practitioners in understanding latent structures that reflect on the evolving career of a musician.

Finally, we point out that various previous efforts looked into the listening behavior of users as trajectories [7, 13, 14]. To perform our study, we employ the TribeFlow method [7]. This method has been shown to be more accurate and interpretable than state-of-the-art baselines in user trajectory mining [7]. We describe the method in the next section.

3. MINING COLLABORATION TRAJECTORIES WITH TRIBEFLOW

Music records (albums) represent a collaborative effort from different individuals such as band members, producers, hired instrumentists, and even graphical artists responsible for the artwork. Whenever individuals collaborate they leave a trail in their career. For example, in *2005 Lucas Dos Prazeres* collaborated with *Naná Vasconcelos* to create the album *Chegada*. Our goal in this study is to understand the collaboration trajectory of individuals. In this context, a trajectory is an ordered sequence of collaborations by a collaborator.

For the sake of clarity, we differentiate between the *artist* or *band* leading a record and the *collaborators* who participate in this album. These collaborators can themselves be the leading artists in other records. Paul, John, Ringo and George are all viewed as collaborators in an album by the artist The Beatles.

More formally, each trajectory defines the sequence of artists (ordered by time) that the collaborator contributed with. Let us define an album as a timestamp, artist/band, and a set of collaborators. That is, $r = (t_r, n_r, a_r, \mathcal{L}_r)$, where r is a record or album, t_r is a timestamp, n_r is the name of the album, a_r is the artist/band and $\mathcal{L}_r$ is the set of collaborators which contributed to r. The subscript r identifies the release for each element of the tuple. Let $\mathcal{R}$ be the set of records. Also, let us define that records are identified by integers $[1, |\mathcal{R}|]$, as well as that for any pair of records $t_i \leq t_{i+1}$. That is, records are ordered by the release timestamp and their ids correspond to the position of the record on the defined ordering.

With the definitions above, the trajectory of a collaborator c is defined as $T_c =< ..., a_i, a_{i+j}, ... >$, where for any pair a_i, a_{i+j} with $j \geq 1$, $t_i \leq t_{i+j}$ (by definition, albums ids are defined by their ordered timestamps). Also, $c \in L_i$ as well as $c \in L_{i+j}$. More importantly, we focus our study on the changes in collaborations over time. That is, we enforce $a_i \neq a_{i+j}$. With this choice, trajectories represent the changes in artists chosen by a collaborator over time.

To exemplify a trajectory, let's us look into John Coltrane as a collaborator. In 1955 to 1956, Coltrane collaborated on various recordings by Miles Davis. Later, in 1957 Coltrane collaborated with Thelonious Monk, again, in various recordings. Afterwards, John Coltrane returned to collaborate with Miles Davis in 1958. Taking this small slice of time as an example, the trajectory of John Coltrane would be represented as: $<$ Miles Davis, Thelonious Monk, Miles Davis $>$. Notice that, regardless of partaking in many records with Miles Davis in 1955 and 1956, the trajectory only captures the change in collaboration from Davis to Monk.

It is important to notice that there exists a variety of latent factors that lead to a collaboration. That is, while some collaborations will emerge due to geographical constraints, others may exist because of musical genres, temporal influence, or social network factors. The trajectory, as defined above, will essentially exist because of choices by the collaborator to collaborate with the artist motivated by these factors. Thus, the end result, regardless the of the underlying factors, is always the same a trajectory of trails/choices (artists) that as collaborator has worked with.

3.1 The TribeFlow Model

To extract the latent structure of trajectories, we employ TribeFlow [7], a recent method proposed to mine sequential data. TribeFlow has recently been shown to discover a meaningful latent structure in a variety of different settings. Here, we apply the method to understand musical collaborations based on the trajectories T_c.

In our setting, TribeFlow models collaborations as random choices over random environments by a collaborator. One example of a random environment can be: Jazz Artists from New Orleans in the 1960s. Due to various constraining factors, as explained above, a collaborator will choose to play with an artist from this environment over a set of albums. After recording these albums, the collaborator will again choose an environment (in some cases, the same as before) and move on to record more albums with different artists. Thus, trajectories are captured as random choices (or random walks) over random environments. Each environment captures a latent factor that leads to collaborations between collaborators and artists.

TribeFlow works using as input a set of trajectories. Given the set of collaborators $c \in \mathcal{C}$, the set of artists $a \in \mathcal{A}$, as well as the set of records $r \in \mathcal{R}$, TribeFlow will explore as input the total set of trajectories $T_c \in \mathcal{T}$. r was defined above. a and c can be defined as the names of artists and collaborators, respectively. A single parameter is required to execute the model. This parameter $k = |\mathcal{Z}|$ captures the number of latent environments $z \in \mathcal{Z}$.

TribeFlow defines a Bayesian graphical model (ommited due to space, see [7]) that learns by performing

Gibbs sampling for each entry (a_{i+j}) in every trajectory $\mathcal{T}$ from the following posterior:

$$P[z|c, a_{i+j}, a_i] \propto \frac{P[z|c]P[a_i|z]P[a_{i+j}|z]}{1 - P[a_i|z]} \quad (1)$$

$P[a|z]$, $P[z|c]$ and $P[z]$ are all probability distributions estimated with TribeFlow. After the model is trained, the above probabilities can be exploited to answer various queries, as we now describe.

3.2 Answering Questions with TribeFlow

First, we point out TribeFlow's model is highly interpretable, since it represent probabilities over $\mathcal{C}$, $\mathcal{A}$, and $\mathcal{Z}$.

By employing the graphical model described in [7], we can re-arrange the probability equations to answer various questions using the TribeFlow model. More specifically:

What is the probability that a collaborator goes from on to collaborate with artist a after choosing environment z? This initial likelihood is captured by the probability $P[a|z]$, learned by the model. It captures the importance of artists in a given environment, that is, artists with high $P[a|z]$ will likely attract more collaborators within that environment.

What is the probability that a collaborator goes from collaborating with artist a to artist a'? Given that a collaborator will collaborate with various artists over a trajectory, this first question captures the importance of artists as links to other artists. That is, how likely is a collaborator to follow-up on his/her career with a' after playing with a. This question is assessed by:

$$P[a'|a] = \sum_{z \in \mathcal{Z}} P[a'|z]P[a|z]P[z] \quad (2)$$

What is the probability of collaborating with environment z' after collaborating with environment z? This question is similar to the previous one. However, it captures the notion of transitions between environments. For instance, how likely is it that a collaborator will go from playing with Dixieland artists to playing with Bebop artists. $P[z'|z]$ is thus defined as:

$$P[z'|z] = \sum_{a \in \mathcal{A}} P[a|z']P[z']P[a|z] \quad (3)$$

What is the probability that an environment z caused collaborator to go from playing with a to playing with a'? This final question can be used to explain trajectories. It capture's how likely is an environment z to cause a change in collaboration from a to a'. This final question is answered with the posterior equation above (Eq. 1).

3.3 Learning the Model

Finally, we point out that our results were achieved by executing the method with the same parameters as discussed by the authors in [7]. More importantly, we made use of the TribeFlow **without** employing the inter-event time heuristics discussed in [7]. We found that employing such heuristics had little to no effect on our results. This effect most likely happens because timestamps t are usually expressed in years (e.g., 2005) on the Discogs dataset. For this reason, on the data we analyzed from 15% to 30% of collaborations happen within a single year, making the time between collaborations useless in such cases (they are all zero). Also, frequent collaborations can happen both over short and large periods of times, such as individuals that take hiatuses on their careers.

Given the exploratory nature of our work, for the sake of interpretability, all analysis in this work use $|\mathcal{Z}| = 30$. This number of latent environments provides an expressive range of latent factors for our purposes while keeping sensemaking easily tractable. To understand how to fine tune $|\mathcal{Z}|$ for other tasks (e.g., prediction) see [6, 7].

Table 1. Summary of the dataset used.

Releases	#		54,466
Artists	#		23,890
	in degree	median	5
		mean	14.8
		max	3,052
Collaborators	#		70,320
	out degree	median	1
		mean	5
		max	830
Collaborations	#		352,932

4. DATA USED

To investigate collaboration among musicians, we leverage the Discogs.com database. Discogs is a collaborative site to register and annotate discographies which makes its database freely available. At the time of writing, the database registers approximately six million record releases, including multiple releases of a same record (eg. CD and LP or LPs releases in different countries). Part of this data is annotated with genre and more specific style tags, and with information about which collaborators participated in a record and in which capability. For example, it is registered in the database that Ron Carter played the bass in Charles Mingus's record Three of Four Shades of Blues, initially released in 1977. As this data is collaboratively created, it is likely biased towards the interests of its contributors, and it is naturally incomplete. Nevertheless, due to its sheer volume and to the community verification of its information, this database provides a promising data source for investigating how collaboration patterns can be understood through trajectory mining.

For this purpose, we use a dataset extracted from the Discogs database as a case study in collaboration. The dataset is comprised of all records tagged with the Jazz genre and which possess metadata about instrumentists in

Table 2. Six exemplary latent environments found learned through TribeFlow from the data. For each environment, we present the first collaborators and artist most strongly associated with the environment. Collaborators are listed first; artists are in italics. All labels were given by the authors based on the artists and collaborators listed.

(a) Bebop	***(b) Bebop 2***	***(c) Free Jazz***
Dizzy Gillespie, Miles Davis, J.J. Johnson, Charlie Rouse, Lucky Thompson, Charlie Parker	John Coltrane, Freddie Hubbard, Donald Byrd, Hank Mobley, Lee Morgan	Steve Lacy, Don Cherry, Archie Shepp, Roswell Rudd Lester Bowie
Dizzy Gillespie, Charlie Parker, Miles Davis, Thelonious Monk, Dizzy Gillespie And His Orchestra	*Art Blakey & The Jazz M., Miles Davis, John Coltrane, Art Blakey, Charles Mingus, Max Roach*	*Archie Shepp, The Sun Ra Arkestra, Cecil Taylor, Steve Lacy Anthony Braxton, Ornette Coleman*
(d) Improvisation UK Big Bands	***(e) Italian***	***(f) Fusion***
Paul Rutherford, Evan Parker, John Edwards, Johannes Bauer, Paul Rogers, Malcolm Griffiths	Giovanni Maier, Gianni Basso, Lauro Rossi, Dino Piana, Giancarlo Schiaffini	Don Alias, David Liebman, Dave Holland, Joe Lovano, Bob Berg, Mino Cinelu
London Improvisers Orchestra, Chris McGregor's Brotherhood Of Breath, Globe Unity Orchestra	*Giorgio Gaslini, Italian Instabile Orchestra, Enrico Rava, Nexus, Chet Baker Nicola Conte*	*Miles Davis, Jaco Pastorius, John Scofield, Chick Corea, Mike Stern, Herbie Hancock*

the Oct 2015 database dump. After removing duplicated releases and releases with no collaboration metadata, the release name, year, and collaboration data were extracted from each release. Furthermore, we focus on instrumentists in which the metadata associated with his/her role in the record contained one of the following words: *bass, guitar, drum, vocal, voic, percuss, keyboard, trumpet, sax, saxophon, trombon, flute, synthes*.

The data resulting from this process is summarized in Table 1. Considering a collaboration as a *(collaborator, time, leading artist, record)* tuple, the in degree of an artist a denotes the number of distinct tuples where a is present in the data. Similarly, the out degree of a collaborators c denotes the number of distinct tuples in which c is present. Ron Carter and Miles Davis are the collaborators and artist with the largest degree in the data. Inspecting the releases, it is possible to note that the numbers of popular artists are slightly inflated due to compilation releases.

5. LATENT TRAJECTORY SPACES

Each latent environment found by TribeFlow can be seen as the result of a set of latent factors that influence the movement of collaborators between artists. Inspecting the collaborators and artists most associated with each environment thus sheds light on what are the relevant factors affecting collaboration sequences in our dataset.

The latent environments found in our case study reveal clear loadings on the environments of stylistic, geographical and chronological latent constraints that shape collaboration trajectories. Table 2 shows six exemplary latent environments as described by the collaborators and artists (in italics) most associated with them [2].

It is possible to clearly distinguish in the first four environments the styles of jazz most often associated with the artists, and to note that in several cases collaborators have notoriously recorded with multiple artists in that environment. It is worthwhile noting that we remove collaborations where the collaborator and artist are equal. For example, there are recordings in the data both of Dizzy Gillespie collaborating in Miles Davis albums and vice-versa.

As for the chronology, it is possible to see in the examples that collaborations of a same period are loaded in different environments. For example, environments *(a)*, *b* and *f* are all associated with Miles Davis. However, his collaborators are divided in these three spaces according to the period of the collaboration. Most markedly, it is possible to distinguish his collaborators from the 70s in environment *(f)* versus earlier collaborators in environment *(b)* and even earlier on *(a)*.

Similarly, the set of top collaborators listed in the *(d)* environment is a core part of the three bands listed as artists. Moreover, in this case, as in the environment *e*, there is a relation of collaborators, artists and geography: the three artist groups listed in environment *(d)* are largely based in the UK, while artists in environment *(e)* are mostly Italians. Along the same lines, there are latent environments not listed in Table 2 that group Brazilian or Scandinavian jazzists, among others.

Before continuing, we point out that various efforts looked into the trajectories of users listening to music [7, 13, 14]. In our study, we make use of TribeFlow [7], a recent trajectory mining technique that has been shown to be both accurate and interpretable when compared to other state-of-the-art methods. We describe the method in the next section.

Besides looking at the intuitive similarities of collaborators or artists associated with an environment, a second possibility for sensemaking is to look for less obvious associations. For example, the presence of Chet Baker, an artist mostly known for his work in the USA, in envi-

[2] Access to the list environments and probabilities is available online at: `https://github.com/flaviovdf/tribeflow/` on the folder `scripts/ismir2016annotated`.

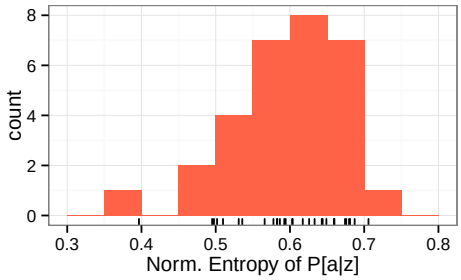


Figure 1. Normalized entropy of $P[a|z]$. The normalized entropy was computed for every z

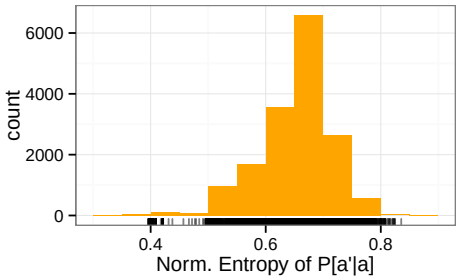


Figure 2. Normalized entropy of $P[a'|a]$. The normalized entropy was computed for every $a' \neq a$.

ronment *(e)* is an example of a relationship less obvious for some. An investigation of Chet Baker's collaborators shows that during his work and late life in Europe, Chet Baker recorded more than one album with a cast of mostly Italian musicians. Several of these musicians (eg. Enrico Rava) have trajectories of collaboration that touch on other Italian artists in Table 2.

6. DIFFERENCES IN COLLABORATION SPACES

Because TribeFlow has a probabilistic interpretation, it allows an analyst to investigate differences in probabilities learned for transitioning to/from different artists and latent environments. A relevant tool for doing so in our context is to examine the entropy of the probability distributions extracted with TribeFlow.

Along those lines, we first investigate which latent environments have a high/low entropy [4] in $P[a|z]$. Recall that, entropy captures the expected uncertainty in a probability distribution. Higher values of entropy indicate that a discrete distribution, our case, is closer to being uniform. Lower values of entropy indicate that the distribution is skewed to a subset of artist in our case.

In other words, the entropy of $P[a|z]$ captures the notion that after choosing to collaborate an artist from z, what is the uncertainty of choosing an artist. Environments with higher entropy indicate that most collaborations remain within a small subset of the artists.

In the following, we use the normalized entropy. Normalization is performed dividing the entropy $\sum_a -P[a|z]log(P[a|z])$ for each space z in the model by that of a Uniform distribution over the same artists. Figure 1 displays the distribution of the values of the normalized entropies for each latent environment.

In our data, the latent environments associated with the highest values of entropy display a normalized entropy of approximately 0.7. These environments seem to be either associated with free or experimental jazz, or to be mostly formed by collaborators and artists from a specific region outside the USA. For example, environments *(d)* and *(e)* are among the highest entropy environments, together with an environment associated with North-European jazz and another one associated with experimental jazz fused with World Music. As for the environments with least entropy,

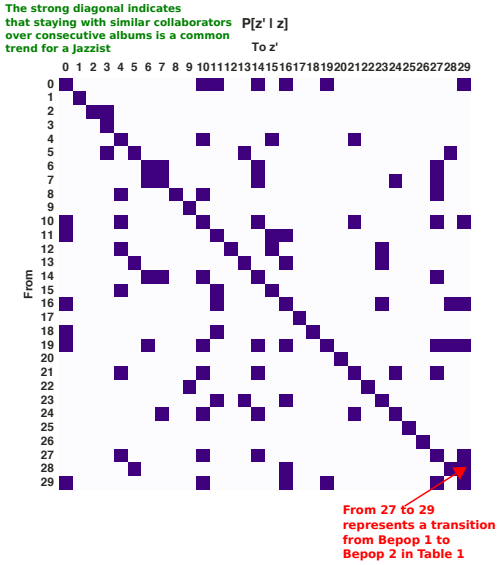


Figure 3. The $P[z'|z]$ highlighting transitions that are above uniform chance $(1/\mathcal{Z})$

the apparent pattern for top-3 environments is the combination of artists from older periods with the presence of a seminal figure. Namely, these three environments have Duke Ellington, Count Basie and Louis Armstrong as their top artists. This suggests that collaborators in our data associated with these latent environments had their trajectories gravitating around these artists with not much collaboration with the other artists in the same environments.

A similar approach can be used to identify who are the artists which have the highest entropy considering the probabilities of collaborating with other artists afterwards ($P[a'|a]$ - Eq. (2)). The five highest-entropy artists in this view are all jazzists often associated with avant-garde or free jazz: Anthony Braxton, Peter Brötzmann, Franz Koglmann, Herb Robertson, and Gerry Hemingway. In a sense, our model points that collaborators who record with these artists are follow no clear pattern in the following collaborations. This can be seen as a sign of the openess in the choice of these artists in collaborating. On the other end of the spectrum, musicians collaborating with artists in the former Czech Republic (eg. Czechoslovak Radio Jazz

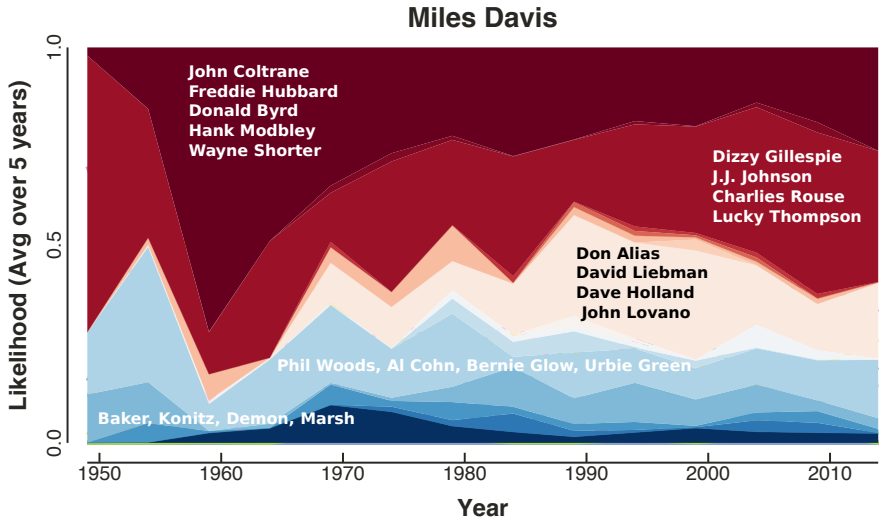


Figure 4. The career of Miles Davis as captured by the model.

Orchestra, Prague Big Band, Ji Vlek) are the ones displaying the least entropy. This last example is likely caused by geographical constraints.

7. MOVEMENT ALONG LATENT SPACES

Considering the probabilities of a collaborator transitioning between distinct spaces $P[z'|z]$ (Eq. (3)) gives us a view of which latent environments are likely to be adjacent in collaborators' trajectories. In Figure 3, we depict this transition matrix. For the sake of clarity, we highlight in this matrix only transitions that were above uniform random chance ($\frac{1}{Z}$). In this plot, latent spaces are numbered from 0 to $k-1$.

Observing the model fit to the data, we see that the pairs of latent environments with the highest probabilities of transition between them will be on the diagonals. That is, artists will likely remain collaborating within the same style after the previous records. Although this is expected, there interesting examples in the non-diagonals as well. For example, the highest probability in a non-diagonal happens between an environment where the most likely artist to collaborate with is Stan Kenton and a second environment where the correspondent artist is Woody Herman. These are two Big Band leaders who led popular bands during the first half of the 20th century. The following five highest probabilities follow a similar pattern, with the fifth being between environments *(b)* and *(a)* in Table 2.

A final frame in which we explore how to use TribeFlow in the context of collaboration trajectories is to inspect the trajectories associated with a prominent artist. Figure 4 shows the likelihood of an environment ($P[z|c, a_{i+j}, a]$ - Eq. (1)) of all collaborator transitions to reach Miles Davis over a period to be associated with each latent environment. We averaged this likelihood over every five years.

For this case, there is a marked change in the likely source of collaborators from the environments *(a)* to *(b)* from Table 2. This change correlates with a major change in the Miles Davis Quintet to the group that would compose it during the first half of the 60s. Wayne Shorter is one of the collaborators strongly associated with environment *(b)* who also recorded with multiple other artists associated with this environment, such as Freddie Hubbard.

8. DISCUSSION

Examining collaboration patterns is an important endeavor in understanding artists' influences and creations. Through a case study of collaboration in Jazz records, we have explored how to use TribeFlow to unveil latent structures in the trajectories of collaborators. The latent environments found in this case study are expressive and were able to help sense making in both popular and more niche collaboration groups in the data. Moreover, these environments seem to express at least stylish, chronological and geographical factors shaping collaboration trajectories. Due to the Bayesian approach of TribeFlow, it is possible to employ a direct probabilistic approach to investigate questions of association at different levels of relationship, such as artist to artist, and artist to environment.

Future work may extend the approach of this paper in deepening the link of the analysis conducted here in its musicological and historical aspects, employing a similar approach to other datasets, and extending both our modeling approach and the tools used so far to compare collaboration in different communities. Moreover, understanding how collaboration trajectories related to musical features (e.g., beat or tempo) can also help researchers better understand collaboration in recordings.

Acknowledgments: This research was supported by grant 460154/2014-1 from CNPq, by the EU-BR BigSea project (MCTI/RNP 3rd Coordinated Call), as well as the EUBrazilCC (grants FP7 614048 and CNPq 490115/2013-6) project.

9. REFERENCES

[1] Arram Bae, Doheum Park, Yong-Yeol Ahn, and Juyong Park. The multi-scale network landscape of collaboration. *PloS one*, 11(3):e0151784, 2016.

[2] Albert-László Barabási, Hawoong Jeong, Zoltan Néda, Erzsebet Ravasz, Andras Schubert, and Tamas Vicsek. Evolution of the social network of scientific collaborations. *Physica A: Statistical mechanics and its applications*, 311(3):590–614, 2002.

[3] Alex Beutel, B Aditya Prakash, Roni Rosenfeld, and Christos Faloutsos. Interacting viruses in networks: can both survive? In *Proc. KDD*. ACM, 2012.

[4] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[5] David Easley and Jon Kleinberg. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

[6] Flavio Figueiredo, Bruno Ribeiro, Jussara M Almeida, Nazareno Andrade, and Christos Faloutsos. Mining Online Music Listening Trajectories. In *Proc. ISMIR*, 2016.

[7] Flavio Figueiredo, Bruno Ribeiro, Jussara M Almeida, and Christos Faloutsos. TribeFlow: Mining & Predicting User Trajectories. In *Proc. WWW*, 2016.

[8] Pablo M Gleiser and Leon Danon. Community structure in jazz. *Advances in complex systems*, 6(04):565–573, 2003.

[9] Charith Gunaratna, Evan Stoner, and Ronaldo Menezes. Using network sciences to rank musicians and composers in brazilian popular music. In *Proc. ISMIR*, 2011.

[10] Paul Heymann and Hector Garcia-Molina. Collaborative creation of communal hierarchical taxonomies in social tagging systems. 2006.

[11] Brian Karrer and Mark EJ Newman. Competing epidemics on complex networks. *Physical Review E*, 84(3):036106, 2011.

[12] Paul Lamere. Social tagging and music information retrieval. *Journal of new music research*, 37(2):101–114, 2008.

[13] Andryw Marques, Nazareno Andrade, and Leandro Balby Marinho. Exploring the Relation Between Novelty Aspects and Preferences in Music Listening. In *Proc. ISMIR*, 2013.

[14] Joshua L. Moore, Shuo Chen, Thorsten Joachims, and Douglas Turnbull. Taste Over Time: The Temporal Dynamics of User Preferences. In *Proc. ISMIR*, 2013.

[15] Mark EJ Newman. Scientific collaboration networks. *Physical review E*, 64(1):016131, 2001.

[16] Juyong Park, Oscar Celma, Markus Koppenberger, Pedro Cano, and Javier M Buldú. The social network of contemporary popular musicians. *International Journal of Bifurcation and Chaos*, 17(07):2281–2288, 2007.

[17] Maximilian Schich, Chaoming Song, Yong-Yeol Ahn, Alexander Mirsky, Mauro Martino, Albert-László Barabási, and Dirk Helbing. A network framework of cultural history. *Science*, 345(6196):558–562, 2014.

[18] D de Lima Silva, M Medeiros Soares, MVC Henriques, MT Schivani Alves, SG de Aguiar, TP de Carvalho, G Corso, and LS Lucena. The complex network of the Brazilian Popular Music. *Physica A: Statistical Mechanics and its Applications*, 332, 2004.

[19] T Teitelbaum, P Balenzuela, P Cano, and Javier M Buldú. Community structures and role detection in music networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 18(4):043105, 2008.

[20] Jennifer Trant. Studying social tagging and folksonomy: A review and framework. *Journal of Digital Information*, 10(1), 2009.

[21] Brian Uzzi and Jarrett Spiro. Collaboration and creativity: The small world problem. *American journal of sociology*, 111(2):447–504, 2005.

GLOBAL PROPERTIES OF EXPERT AND ALGORITHMIC HIERARCHICAL MUSIC ANALYSES

Phillip B. Kirlin
Department of Mathematics and Computer Science, Rhodes College
kirlinp@rhodes.edu

ABSTRACT

In recent years, advances in machine learning and increases in data set sizes have produced a number of viable algorithms for analyzing music in a hierarchical fashion according to the guidelines of music theory. Many of these algorithms, however, are based on techniques that rely on a series of local decisions to construct a complete music analysis, resulting in analyses that are not guaranteed to resemble ground-truth analyses in their large-scale organizational shapes or structures. In this paper, we examine a number of hierarchical music analysis data sets — drawing from Schenkerian analysis and other analytical systems based on *A Generative Theory of Tonal Music* — to study three global properties calculated from the shapes of the analyses. The major finding presented in this work is that it is possible for an algorithm that only makes local decisions to produce analyses that resemble expert analyses with regards to the three global properties in question. We also illustrate specific similarities and differences in these properties across both ground-truth and algorithmically-produced analyses.

1. INTRODUCTION

Music analysis refers to a set of techniques that can illustrate the ways in which a piece of music is constructed, composed, or organized. Many of these procedures focus on illustrating relationships between certain types of musical objects, such as *harmonic analysis*, which can show how chords and harmonies in a composition function in relation to each other, or *voice-leading analysis*, whose purpose is to illustrate the flow of a melodic line through the music. Some types of analysis are explicitly *hierarchical*, in that their purpose is to construct a hierarchy of musical objects illustrating that some objects occupy places of higher prominence in the music than others. Different kinds of hierarchical analysis have different methods for determining the relative importance of objects in the hierarchy. The most well-known of these hierarchical procedures is *Schenkerian analysis*, which organizes the notes of a composition in a hierarchy according to how much each note contributes to the overall musical structure. The procedure also illustrates how notes at one level of the hierarchy function in relation to surrounding notes at higher and lower levels. While Schenkerian analysis is the most common type of hierarchical analysis in the music theory community, there are a number of similar procedures that were developed in the music and linguistics community, specifically the procedures put forth in the book *A Generative Theory of Tonal Music* by Lerdahl and Jackendoff [10], abbreviated here as *GTTM*. GTTM applies an explicitly hierarchical view to multiple aspects of a musical composition, leading to two types of analysis known as *time-span reductions* and *prolongational reductions*, both of which are similar to Schenkerian analysis in that all three analytical methods organize the pitches of the music in hierarchies of relative importance, allowing a person to view a musical composition at multiple levels of abstraction.

None of these types of musical analysis were originally developed as computational algorithms, and so all contain certain ambiguities in their definitions. Schenkerian analysis, in particular, is known for originally being defined primarily through examples and not via a step-by-step procedure. Similarly, the two GTTM reductional analysis systems contain preference rules that can conflict with each other; the authors explicitly state that there is not enough information in GTTM to provide a "foolproof algorithm" for analyzing a composition. Nevertheless, there are now a number of automated computational systems that can construct analyses in a Schenkerian fashion [7, 11] or by following the rules of time-span or prolongational reductions in the GTTM formalism [4]. The most common computational technique underlying these systems and others like them is the *context-free grammar*: such a formalism is widely-adopted because such grammars are easily applied to musical objects, are inherently rule-based, can be adapted to work with probabilities, and admit a computationally-feasibly $O(n^3)$ parsing algorithm that can be used to find the best analysis for a piece of music.

The largest downside to context-free grammars is precisely that they are *context-free*: there are restrictions on how much musical context can be used when applying the rules of a grammar to "parse" a piece of music into an analysis. Most decisions made during the analysis process under the context-free paradigm have to be made somewhat "locally," and are unable to consider many important "global" properties that are critical to producing a high-


© Phillip B. Kirlin. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Phillip B. Kirlin. "Global Properties Of Expert and Algorithmic Hierarchical Music Analyses", 17th International Society for Music Information Retrieval Conference, 2016.

quality musical analysis. For instance, certain compositional techniques, such as identical repetitions of an arbitrarily melodic sequence, are impossible to describe satisfactorily with a context-free grammar [13]. Other common practices, such as the rules of musical form, manifest themselves as certain shapes and structures in the hierarchy [9, 10, 15], and it is unclear whether a purely context-free system could identify such structures. In this work, we show (a) evidence that context-free grammars can reproduce certain global structures in music analyses, and (b) similarities and differences in those global structures across various types of hierarchical music analysis.

2. REPRESENTATION OF HIERARCHICAL ANALYSES

For simplicity, we assume we are analyzing a monophonic sequence of notes. The two most common ways to do this in a hierarchical manner are (a) to create a hierarchy directly over the notes, and (b) to create a hierarchy over the melodic intervals *between* the notes. Each representation has distinct advantages and disadvantages [2, 12], but Schenkerian analysis is more easily represented by a hierarchy of melodic intervals. We explain this through the following example. Imagine we wish to analyze the five-note descending passage in Figure 1, which takes place over G-major harmony. Schenkerian analysis is guided by the concept of a *prolongation*: a situation where a note or pair of notes controls a musical passage even though the governing note or notes may not be sounding throughout the entire passage. For instance, in Figure 1, the sequence D–C–B contains the passing tone C, and we say the C prolongs the motion from the D to the B. The effect is similar for the notes B–A–G. However, there is another level of prolongation at work: the entire five-note span is governed by the beginning and ending notes D and G. This two-level structure can be represented by the binary tree shown in Figure 2(a), which illustrates this hierarchy of prolongations through the melodic intervals they encompass. A more succinct representation is shown in Figure 2(b): this structure is known as a *maximal outerplanar graph* or *MOP*, and illustrates the same hierarchy as the binary tree [14].

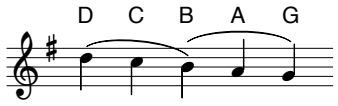


Figure 1. An arpeggiation of a G-major chord with passing tones. The slurs are a Schenkerian notation used to indicate the locations of prolongations.

A MOP is a graph representing a complete triangulation of a polygon. Like their binary tree equivalents, MOPs are rooted, but by an edge, rather than a vertex; this edge represents the most abstract level of the melodic hierarchy. Every triangle within a MOP corresponds to a hierarchical relationship among the three notes that form the triangle, with the middle note taking on a subservient role in relation

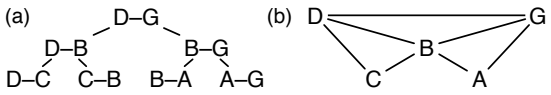


Figure 2. The prolongational hierarchy of a G-major chord with passing tones represented as (a) a tree of melodic intervals, and (b) a MOP.

to the left and right notes. It is equivalent to say that every triangle has a parent edge and two child edges, or has two parent vertices and a child vertex. Triangles closer to the root of the MOP express more abstract relationships than those farther away. Though originally developed to represent Schenkerian prolongations, we will use MOPs later in this paper to work with the GTTM reductional systems as well.

2.1 Large-Scale Organization of MOPs

Like binary trees, MOPs can be described by a variety of global attributes that are determined from their overall shape. We explore three such attributes and how common music composition and analysis practices affect these attributes.

Height: The *height* of a MOP is defined to be the number of triangles in the longest possible sequence of triangles from the root edge moving through subsequent child edges to the bottom of the MOP. It is analogous to the height of the equivalent binary tree. For a MOP with a fixed number of triangles, there are a certain range of possible heights; for instance, Figure 3 shows two MOPs with five triangles, one with a height of 5 and one with a height of 3. Because a MOP with n vertices will always contain $n-2$ triangles, we can say the maximum height of such a MOP is $n-2$, whereas the minimum height is $\lceil \log_2(n-1) \rceil$.

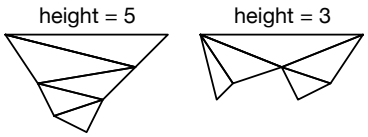


Figure 3. Two MOPs, each with five triangles, but different heights.

Investigating the heights of the MOPs that result from hierarchical analysis gives us insight into the compositional structure of the music from which the MOPs were created. MOPs with large heights result from situations where the notes of critical importance in a hierarchy are positioned towards the beginning and end of a musical passage, with importance decreasing monotonically as one moves towards the middle of the passage in question (as in the left MOP of Figure 3. In contrast, MOPs with small heights result from hierarchies where the structural importance does not increase or decrease monotonically over time during a passage, but rather rises and falls in a pattern similar to how strong and weak beats fall rhythmically

within a piece.

Average Path Length: While height provides a coarse measure of the shape of a MOP, a different metric, *average path length*, provides finer-grained detail about the balancedness or skewedness of the structure [5]. Measuring the level of balancedness or unbalancedness in a tree gives an overall sense of whether the leaves of the tree are all at roughly the same level or not; when applied to a MOP, this determines whether the leaf edges are all roughly the same distance away from the root edge. Towards that end, the average (external) path length in a MOP is calculated by averaging the lengths of all the paths from the root edge to all leaf child edges. For a MOP with n vertices, the minimum average path length is very close to $\log_2 n + 1$, while the maximum average path length is $\frac{(n-1)(n+2)}{2n}$ [1, 8].

GTTM and related literature posits that music is constructed around structures that are largely balanced, that is, tending towards shorter average path lengths. One reason for this, from a prolongational standpoint, is that a melodic hierarchy expressed as a MOP illustrates patterns of tension and relaxation: each triangle represents tension rising from the left parent vertex to the child, then relaxing from the child to the right parent. Balanced MOP structures imply waves of tension and relaxation roughly on the same level, whereas unbalanced MOPs imply tension and relaxation patterns that may seem implausible to a listener: "it is most unlikely that a phrase or piece begins in utmost tension and proceeds more or less uniformly towards relaxation, or that it begins in relaxation and proceeds toward a conclusion of utmost tension." [9, 10].

Left-Right Skew: An important consideration not accounted for by the concepts of height or average path length is determining whether the MOP has more left-branching structures or more right-branching structures. To this end, we define a variant of path length by choosing to count a left-branch within a path as -1 and a right-branch as $+1$. It is clear that this metric assigns negative numbers to all MOP edges that lie on paths from the root edge with more left branches than right branches, and positive numbers to edges in the opposite situation. Using this measure of distance, we define the *left-right skew* of a MOP to be the sum of these numbers for all paths in a MOP from the root edge to a leaf edge, giving us an overall sense of whether the MOP is skewed to the left or to the right. Due to the organization of leaf edges within a MOP, a fully right-branching MOP with n vertices will achieve a left-right skew of

$$\sum_{i=-1}^{n-4} i + (n-2) = \frac{n^2}{2} - \frac{5n}{2} + 3$$

and a fully left-branching MOP will achieve a corresponding negative value.

3. FIRST EXPERIMENT

Our first experiment is intended to answer the question, "Can a fully-automated algorithm for music analysis based on context-free parsing techniques produce MOPs with global structural attributes matching those of ground-truth MOPs?" Note that we are assuming that the three global attributes — MOP height, average path length, and left-right skew — are not randomly distributed; this assumption is based on the previous work described earlier detailing that the overall shape of a hierarchical music analysis is most decidedly not random, but influenced by the way compositions are constructed and the manner in which listeners hear and interpret them.

We used the PARSEMOP system in concert with the SCHENKER41 data set to conduct this experiment. SCHENKER41 is a data set of 41 common practice period musical excerpts along with corresponding Schenkerian analyses in MOP form for each excerpt [6]. The excerpts are homogeneous: they are all in major keys, written or arranged for a keyboard instrument or voice with keyboard accompaniment, and do not modulate. The corresponding analyses of the excerpts are all derived from textbooks or other expert sources, and can be regarded as ground truth. SCHENKER41 serves as training data for the PARSEMOP machine-learning system, which learns the rules of Schenkerian analysis by inferring a probabilistic context-free grammar from patterns extracted from SCHENKER41 [7]. After training, PARSEMOP can produce MOP analyses for new, previously-unseen pieces of music.

There are three variants of PARSEMOP, which vary only in treatment of the *Urlinie*, a uniquely Schenkerian concept. According to Schenker, all tonal music compositions should have, at the most abstract level of the melodic hierarchy, one of three possible background structures. These three structures, representing Schenker's fundamental conception of melody, consist of a stepwise descent from the third, fifth, or eighth scale degree to the tonic below, and the entire melodic content of the piece serves as an elaborate prolongation of this descending melodic line. PARSEMOP-A, when trained on the SCHENKER41 corpus, does not have any *a priori* knowledge of the *Urlinie*: it does not even know that such a concept exists in music. Therefore, PARSEMOP-A produces output MOPs that usually do not contain an *Urlinie*, except if by chance. PARSEMOP-B, on the other hand, is given information about the *Urlinie* for the pieces of music it is analyzing. PARSEMOP-B produces output MOPs that always contain the correct *Urlinie* (the structure is copied from the input music). Clearly, PARSEMOP-A and PARSEMOP-B represent two opposite ends of the spectrum with regard to the *Urlinie*. PARSEMOP-C is a compromise between the two: it uses extra rules in the context-free grammar to guarantee that an *Urlinie* will be produced in the output MOP analyses, but it may not match the notes of the correct *Urlinie* exactly.

We ran the three PARSEMOP variants using leave-one-out cross-validation on each of the 41 excerpts in the SCHENKER41 corpus, leaving us with four sets of MOPs: one ground-truth, and three algorithmically produced. Because the minimum and maximum values for each of the three global MOP attributes are dependent on the number of vertices in a MOP (corresponding to the length of the

musical excerpt in question), we normalized the values for the attributes as follows. MOP height was scaled to always occur between 0 and 1, with 0 corresponding to a MOP of minimum height for a given piece, and 1 corresponding to the maximum height. Average path length was similarly scaled to be between 0 and 1. Left-right skew was scaled to be between -1 and $+1$, with 0 corresponding to a perfectly left-right balanced MOP, and $-1/+1$ corresponding to maximally left- or right-branching MOPs. Finally, we compared the distribution of the attributes obtained for each PARSEMOP variant against corresponding attribute distributions calculated from the ground-truth MOPs; histograms can be seen in Figure 4.

The data illustrate a number of phenomena. Firstly, the histograms for height and average path length suggest that all the data sets, both ground-truth and algorithmically-produced, show a preference for MOPs that are a compromise between balanced and unbalanced, but tending toward balanced: very deep MOPs or MOPs with long path lengths (values close to 1) are avoided in all data sets. PARSEMOP-B and -C have higher average heights and average path lengths than PARSEMOP-A due to the presence of an *Urlinie*, which is always triangulated in a deep, unbalanced fashion. This is most evident in the PARSEMOP-B and -C plots for left-right skew: these show a dramatic left-branching structure that is almost certainly due to the *Urlinie*, especially when compared to the histogram for PARSEMOP-A.

Secondly, it is clear that in some situations, the context-free grammar formalism does a remarkably good job at preserving the overall shape of the distribution of the attributes, at least through visual inspection of the histograms. We can confirm this by computing Spearman's rank correlation coefficient ρ for each pair of data — this calculates the correlation between a list of the 41 pieces sorted by a ground-truth metric and the same metric after having been run through PARSEMOP. All pairs show positive correlation coefficients, with eight of the nine being statistically significant at the $\alpha = 0.005$ level (obtained via the Šidák correction on 0.05). In particular, rank correlation coefficients for all three PARSEMOP-B comparisons are all greater than 0.9, indicating a strong correlation.

However, a high Spearman ρ coefficient does not necessarily imply the distributions are identical. We ran two-sample two-tailed t-tests on each pair of data to determine if the means of the two data sets in question were different (the null hypothesis being that the means of each data set within a pair were identical). Two cases resulted in p-values significant at the $\alpha = 0.005$ level, indicating rejection of the null hypothesis: the height comparisons for PARSEMOP-B, and the left-right skew comparisons for PARSEMOP-C. This implies a situation where PARSEMOP-B is apparently very good preserving relative ranks of MOP heights in the data set (this rank correlation between algorithmic MOP heights and ground-truth MOP heights was revealed above), but there is also a statistically significant, though small, difference in the means of the distribution of these heights.

4. SECOND EXPERIMENT

Our first experiment suggests that there may be some bias in our calculations being introduced by the *Urlinie*, namely because it has a particular structure that is always present in the resulting analyses. This situation is further complicated by the presence of a number of short pieces of music in the SCHENKER41 data set, where, for instance, the music may consist of ten notes, five of which constitute the *Urlinie*. In a situation like this, the MOP structure is already likely determined by the locations of the notes of the *Urlinie*, and so the PARSEMOP algorithm has very little effect on the final shape of the MOP analysis. In short, we suspect that these two factors may be artificially increasing the height and average path length of the algorithmically-produced MOPs.

To address this, we replicated the first experiment but only calculated the global MOP attributes for pieces with at least 18 notes (leaving 23 pieces out of 41), hypothesizing that having more notes in the music would outweigh the effects of the *Urlinie*. The leave-one-out cross-validation step was not altered (this still used all the data). Figure 5 illustrates the new histograms compiled for this experiment. In short, these new data support our hypothesis: removing short pieces largely eliminates very deep MOPs and those with very long average path lengths.

We can again address similarities and differences using tests involving Spearman's correlation coefficient ρ and paired t-tests. All of the statistically significant results for ρ relating to PARSEMOP-B still remain: all three global MOP structure attributes calculated on the PARSEMOP-B MOPs are strongly positively rank-correlated ($\rho > 0.8$) with the ground-truth MOP attributes. There is a weaker rank correlation ($\rho \approx 0.583$) between the left-right skew attribute calculated on the PARSEMOP-C data and its ground-truth that is also statistically significant ($p < 0.005$). In contrast, the two statistical significances identified via the t-tests in the first experiment both disappear when run on only the pieces of at least 18 notes, suggesting that these association may have spurious, caused by noise in the shorter pieces.

5. THIRD EXPERIMENT

In our third experiment, we branched out from Schenkerian analysis to explore *A Generative Theory of Tonal Music*'s time-span and prolongational reductions. These are two forms of music analysis that, like Schenkerian analysis, are designed to illustrate a hierarchy among the notes of a musical composition.

Time-span reduction is introduced in GTTM as grounded in the concept of pitch stability: listeners construct pitch hierarchies based primarily on the relative consonance or dissonance of a pitch as determined by the principles of Western tonal music theory. However, pitch stability is not a sufficient criteria upon which to found a reductional system, because pitches do not occur in a vacuum, but take place over time: there are temporal and rhythmic considerations that are required. Lerdahl and

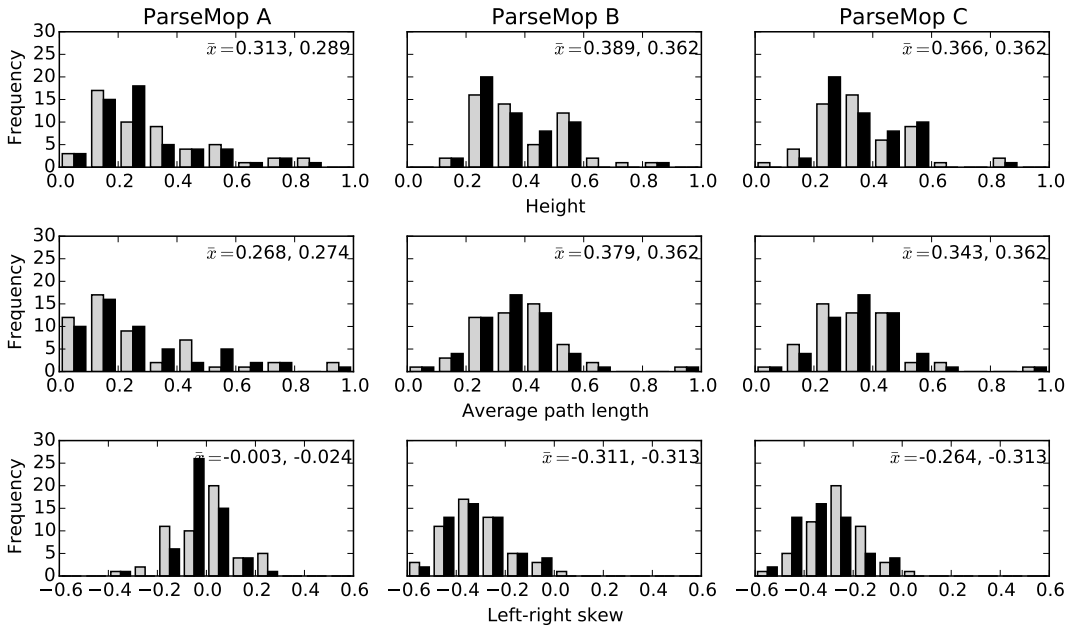


Figure 4. Histograms displaying the distribution of global MOP attributes comparing algorithmically-generated MOPs (grey bars) and ground-truth MOPs (black bars). Sample means are shown for algorithmic and truth MOPs, respectively. The ground-truth bars for PARSEMOP-A are different from -B and -C because PARSEMOP-A has no conception of the *Urlinie*, and therefore the *Urlinie* in the ground-truth MOPs is triangulated slightly differently in the training data for PARSEMOP-A.

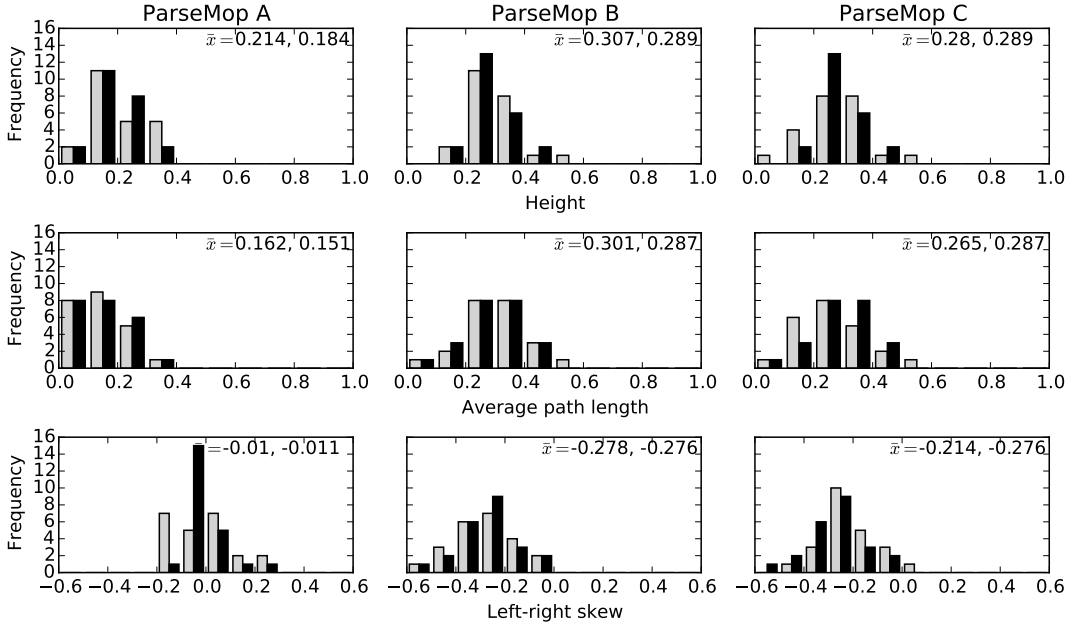


Figure 5. Histograms of the same variety as in Figure 4, but only for excerpts containing 18 or more notes.

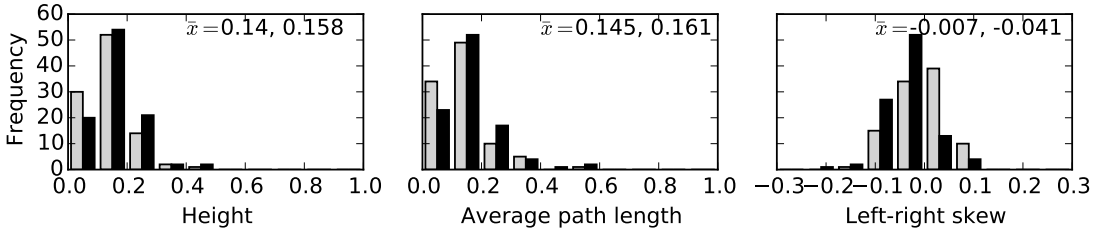


Figure 6. Histograms of the global MOP structure attributes comparing prolongational reductions (grey bars) and time-span reductions (black bars).

Jackendoff address this by basing time-span reductions on their conceptualizations of metrical and grouping structure, where metrical structure is determined from analyzing the strong and weak beats of a composition, while the grouping structure comes from a listener perceiving notes grouped into motives, phrases, themes, and larger sections.

Though similar to time-span reduction, prolongational reduction adds the concepts of tension and relaxation to the criteria that are used to form a musical hierarchy. The motivation for the need for two types of reduction is that time-span reductions cannot express some structural relationships that take place across grouping boundaries, which determine the overall form of a time-span analysis. In contrast, prolongational reductions are not tied to grouping boundaries, and therefore can represent rising and falling tension across such boundaries. The two types of trees produced by the reductional systems are often similar in branching structure at the background levels, but become more dissimilar at lower levels of the hierarchies [10].

Because time-span and prolongation reductions seem similar on the surface, it is appropriate to address their similarities and differences through a study of the three global attributes calculated for MOPs in the previous section. We perform this study by using the GTTM database developed by Hamanaka et al. [3], through their research on the feasibility of automating the analytical methods described in GTTM [4]. This database consists of 300 eight-bar excerpts of music from the common practice period, along with time-span reductions and prolongational reductions for certain subsets of the pieces. Specifically, there are 99 excerpts that include both time-span and prolongational reductions. Note that these reductions in the database were produced by a human expert, not an algorithm.

Our first task was to convert the time-span and prolongation reductions into MOPs. This is necessary because although time-span and prolongational reductions are expressed through binary trees (which are structurally equivalent to MOPs), the GTTM reductions use binary trees created over the notes of a composition, whereas MOPs are equivalent to binary trees created over melodic intervals between notes, as shown earlier in Figure 2. Therefore, we require an algorithm to convert between these two fundamentally different representations.

Time-span and prolongational reductions are represented by trees with primary and secondary branching, like that of Figure 7(a). Phase one of the conversion algorithm converts these trees into an intermediate representation: a multi-way branching tree where all children of a note are represented at the same level, as in Figure 7(b). Phase two converts this intermediate representation to a MOP by adding edges in appropriate places, as in Figure 7(c). This conversion algorithm is guaranteed to preserve all hierarchical parent-child relationships present in the original time-span or prolongational tree. It may introduce other relationships through adding additional edges, however.

Once all the time-span and prolongational reductions were converted into MOPs, we computed histograms of the MOP height, average path length, and left-right skew for

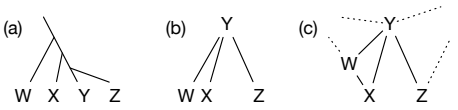


Figure 7. Illustration of a time-span/prolongational tree structure converted into a MOP.

both the time-span reductions and prolongational MOPs. These are shown in Figure 6. Spearman's rank coefficient test reveals positive rank-correlations between MOP height ($\rho = 0.660$), average path length ($\rho = 0.762$), and left-right skew ($\rho = 0.300$) calculated from time-span analyses and the corresponding attribute for prolongational analyses. At the same time, paired t-tests suggest that the sample means have statistically significant differences for all three attributes as well, when comparing time-span and prolongational reductions. Lastly, though the paired histograms for height and average path length may appear similar, the left-right skew paired histograms seem more visually different. This is confirmed via a two-sample Kolmogorov-Smirnov test, which indicates the left-right skew values for time-span versus prolongational reductions are drawn from different distributions. All of these statistical significances account for multiple comparisons using the Šidák correction ($\alpha = 0.05 \rightarrow \alpha = 0.017$).

6. CONCLUSIONS

The data presented here suggest a number of conclusions. The first two experiments involving PARSEMOP imply that when PARSEMOP makes mistakes in analyzing music, the mistakes do not drastically change the overall shape or structure of the corresponding ground-truth analysis. This information is challenging to reconcile with the fact that PARSEMOP, like any music analysis algorithm derived from context-free parsing techniques, does no global calculations related to shape or structure during the analytical process. One explanation is that the notes of a music composition imply an overall shape and structure that the analytical process simply reveals, in that the shape is inherently present in the music and does not have to be given explicitly to the grammar. If this were true, then using a formal grammar class higher in the Chomsky hierarchy (e.g., a grammar with some amount of context-sensitivity) may not be necessary to create algorithms that can analyze music satisfactorily.

The third experiment comparing time-span and prolongational analyses reveals fundamental differences and similarities in the overall structure of the two analytical forms. For instance, it is clear that both types of reductional system strongly prefer balanced, shallow trees, as is clear from the histograms on height, average path length, and left-right skew. Also, both analysis varieties produce trees that slightly skew to the left. However, our statistical tests also strongly suggest that the underlying distributions of the global MOP structure attributes are different, even if the differences in means happen to be small.

7. REFERENCES

[1] Helen Cameron and Derick Wood. Maximal path length of binary trees. *Discrete Applied Mathematics*, 55(1):15–35, 1994.

[2] Édouard Gilbert and Darrell Conklin. A probabilistic context-free grammar for melodic reduction. In *Proceedings of the International Workshop on Artificial Intelligence and Music, 20th International Joint Conference on Artificial Intelligence*, pages 83–94, Hyderabad, India, 2007.

[3] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. Musical structural analysis database based on GTTM. In *Proceedings of the 15th International Society for Music Information Retrieval Conference*, pages 325–330, 2015.

[4] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. σGTTM III: Learning based time-span tree generator based on PCFG. In *Proceedings of the 11th International Symposium on Computer Music Multidisciplinary Research*, pages 303–317, 2015.

[5] Maarten Keijzer and James Foster. Crossover bias in genetic programming. In *Proceedings of the European Conference on Genetic Programming*, pages 33–44, 2007.

[6] Phillip B. Kirlin. A data set for computational studies of Schenkerian analysis. In *Proceedings of the 15th International Society for Music Information Retrieval Conference*, pages 213–218, 2014.

[7] Phillip B. Kirlin and David D. Jensen. Using supervised learning to uncover deep musical structure. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 1770–1776, 2015.

[8] Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison Wesley Longman, Redwood City, CA, second edition, 1998.

[9] Fred Lerdahl. *Tonal Pitch Space*. Oxford University Press, Oxford, 2001.

[10] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, Cambridge, MA, 1983.

[11] Alan Marsden. Schenkerian analysis by computer: A proof of concept. *Journal of New Music Research*, 39(3):269–289, 2010.

[12] Panayotis Mavromatis and Matthew Brown. Parsing context-free grammars for music: A computational model of Schenkerian analysis. In *Proceedings of the 8th International Conference on Music Perception & Cognition*, pages 414–415, 2004.

[13] Mariusz Rybnik, Wladyslaw Homenda, and Tomasz Sitarek. *Foundations of Intelligent Systems: 20th International Symposium, ISMIS 2012*, chapter Advanced Searching in Spaces of Music Information, pages 218–227. 2012.

[14] Jason Yust. *Formal Models of Prolongation*. PhD thesis, University of Washington, 2006.

[15] Jason Yust. *Organized Time*, chapter Structural Networks and the Experience of Musical Time. 2015. Unpublished manuscript.

HUMAN-INTERACTIVE OPTICAL MUSIC RECOGNITION

Liang Chen
Indiana University Bloomington
chen348@indiana.edu

Erik Stolterman
Indiana University Bloomington
estolter@indiana.edu

Christopher Raphael
Indiana University Bloomington
craphael@indiana.edu

ABSTRACT

We propose a human-driven Optical Music Recognition (OMR) system that creates symbolic music data from common Western notation scores. Despite decades of development, OMR still remains largely unsolved as state-of-the-art automatic systems are unable to give reliable and useful results on a wide range of documents. For this reason our system, *Ceres*, combines human input and machine recognition to efficiently generate high-quality symbolic data. We propose a scheme for human-in-the-loop recognition allowing the user to constrain the recognition in two ways. The human actions allow the user to impose either a pixel labeling or model constraint, while the system re-recognizes subject to these constraints. We present evaluation based on different users' log data using both *Ceres* and *Sibelius* software to produce the same music documents. We conclude that our system shows promise for transcribing complicated music scores with high accuracy.

1. INTRODUCTION

Optical Music Recognition (OMR), the musical cousin of Optical Character Recognition (OCR), seeks to convert score images into symbolic music representations. Success in this endeavor would usher music into the 21st century alongside text, paving the way for large scale symbolic music libraries, digital music stands, computational musicology, and many other important applications.

Research in Optical Music Recognition (OMR) dates back to the 1960s with efforts by a large array of researchers on many aspects of the problem [3, 5, 10–14, 17, 18, 20, 21, 25, 29] including several overviews [6, 23], as well as well-established commercial efforts [2] [1]. While evaluation of OMR is a challenging task in its own right [8], it seems fair to say that the problem remains largely unsolved, in spite of this long history. The reason is simply that OMR is very hard, representing a grand challenge of document recognition.

One reason OMR is so difficult stems from the heavy tail of symbols used in music notation. While a small core of symbols account for the overwhelming majority of ink on the printed page, these are complemented by a long list of familiar symbols that may be absent in many or most pages. These include repeat signs, D.S. and D.C. directives, a wide variety of possible ornaments and articulations, harmonics, fingerings, pedaling, arpeggiation, fermati, double sharps and flats, pedaling, 1st and 2nd endings, repeats, etc. While each of these symbols can be recognized with reasonable accuracy by fairly standard means, the symbols are rare enough that the unavoidable false positives they produce often outweigh the value of the correct detections we may find. This constitutes one of the the essential paradoxes of OMR: we cannot simply ignore unusual symbols, though their inclusion often leads to worse performance overall.

We sometimes refer to the heavy tail described above as the "sprawl of OMR." This sprawl is not limited to the range of symbols, but also includes the many exceptions to familiar notational rules. For instance, in standard notation beamed groups, notes and chords carry the majority of musical content, thus their recognition must be central to any OMR effort. The construction of these symbols is highly rule-bound, arguing strongly for recognition approaches that exploit the symbols' grammatical nature. The difficulty here comes from the many special cases we must account for. For example, note heads usually lie on one side of the stem, though chords with note heads on adjacent staff positions are usually rendered with "wrong side" note heads; beamed groups usually have closed note heads though measured alternations between two pitches is often abbreviated with two open note heads in a beamed group; beam groups usually have stems that go in a single direction from the beam, though one occasionally sees both directions from a single beamed group; beamed groups are usually associated with a single staff, but can span both staves of a grand staff when necessary; augmentation dots are generally placed to the right of the note heads they belong to, though dense voicing can force them to wander far off their nominal positions. As with the heavy tail of symbols, these special cases can all be modeled and recognized, though the results are not what one would hope for. The majority of notation will *not* contain these rarer cases; allowing for these exceptions when they do not occur begs for trouble, reliably degrading the recognized results. However, these exceptions are common enough that we doubt a useful OMR system can be developed without accounting for them somehow. This is essentially the same paradox as that posed by the heavy tail: we must recognize these unusual cases but cannot allow them to result in degraded performance overall. Dealing with this sprawl is a

© Liang Chen, Erik Stolterman, Christopher Raphael. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Liang Chen, Erik Stolterman, Christopher Raphael. "Human-Interactive Optical Music Recognition", 17th International Society for Music Information Retrieval Conference, 2016.

central issue we address in this paper.

In light of these (and other) obstacles we doubt that any fully automatic approach to OMR will ever deal effectively with the wide range of situations encountered in real life recognition scenarios. For this reason we formulate the challenge in terms of *human-in-the-loop computing*, developing a *mixed-initiative system* [15, 19, 27, 28] fusing both human and machine abilities. The inclusion of the human adds a new dimension to the OMR challenge, opening a vast expanse of unexplored potential.

However, there is another reason for the human-computer formulation we favor. While there may be some uses for moderate quality symbolic music data, we believe the most interesting applications require a level of accuracy near that of published scores. The human will not only play the important role of guiding the system toward the required level of accuracy, but must also "bless" the results when they are complete. We expect symbolic music data lacking this human imprimatur will be of dubious value.

Commercial systems often deal with this issue by pipelining their results into a score-writing program, thus allowing the user to fix the many recognition problems. This approach creates an artificial separation into the two phases of recognition and correction. Rather, since we require a human to sign-off on the end result, we propose to involve her in the heart of the process as well.

In what follows we present the view of human-in-the-loop OMR taken in our *Ceres* system. Our essential idea is to allow the user two axes of control over the recognition engine. In one axis the user chooses the *model* that can be used for a given recognition task, specifying both the exceptions to the "rules" discussed above as well as the relevant variety of symbols to be used. In the other, the user labels misrecognized pixels with the correct primitive type, allowing the system to re-recognize subject to user-imposed constraints. This provides a simple interface in which the user can provide a wealth of useful knowledge without needing to understand the inner-workings and representations of the system. Thus we effectively address the *communication* issue of mixed-initiative literature. Our work has commonalities with various other human-in-the-loop efforts such as [26, 30], though most notably with other approaches that employ constrained recognition as driven by a user [4, 7, 24].

2. HUMAN-INTERACTIVE SYSTEM

The main part of our *Ceres* OMR system deals with symbol recognition, which occurs after the basic structure of the page has been identified. For each type of symbol or group of symbols we use a different *graphical model*. Here we depict only the isolated chord graph in Fig. 1 which serves as a template for the others, and refer the more detailed discussions to our previous papers [9, 22].

Each individual music symbol (beamed group, chord, clef-key-signature, slur, etc.) is grammatically constrained based on a generative graph, enabling automatic, model-driven, symbolic recognition. Perhaps more difficult than

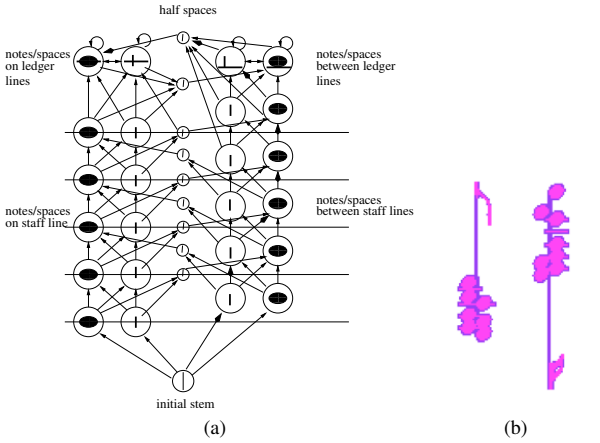


Figure 1: (a) Graphical model for isolated chord. (b) Symbol samples generated via random walk over the graphical model shown in (a).

individual symbol recognition is the challenge of decomposing the image into disjoint, grammatically-consistent symbols. To address this problem, our system identifies *candidates* for the different symbol types such as chords, beamed groups, dynamics, slurs, etc. The user chooses some of these candidates for interactive recognition. After each recognition task is completed the resulting symbol will be saved, thus constituting a constraint for future candidate detection and recognition — we constrain our system to identify mostly non-overlapping symbols.

Our current human-driven system performs recognition in a symbol-by-symbol fashion as opposed to the measure-based version proposed in [9]. The symbol-based scheme allows for a responsive and efficient interface, which functions with the symbol recognizers implemented in our system . Here the human is allowed to interact with all three steps: candidate identification, interactive recognition, and post-processing. In the candidate identification and post-processing steps, the user directs the decision-making process by either selecting a system-proposed candidate, adding a missing candidate, or deleting an incorrectly saved symbol. In the interactive recognition step, the user actions impose extra labeling or model constraints to the recognizer, while the system automatically invokes re-recognition *subject to these constraints*.

The interactive recognition begins by performing fully automatic recognition of the selected candidate. In many cases the result will be correct and will be saved. Otherwise the process iterates between human action and machine re-recognition until it yields the correct result. In each iteration of this process the user will click on a particular pixel to input the corresponding *primitive* label (solid note head, stem, beam, etc.) or change the model settings to an appropriate choice. The whole process requires only basic knowledge of music notation and thus extends the range of potential users.

Our recognizers seek hypotheses that give the highest probability to the pixel intensities, $g(x)$, where x is a particular location. More explicitly, we regard a hypothesis, h,

as a labeling of pixel sites, x, with models, $M_h(x)$, where $M_h(x) \in \mathcal{L} = \{b, w, t, 0\}$. The labels in $\mathcal{L}$ correspond to a probability models for black, white, transitional, and background pixels (P_b, P_w, P_t, P_0). We measure the quality of a hypothesis, h, by

$$S(h) = \sum_{x \in D(h)} \log \frac{P_{M_h(x)}(g(x))}{P_0(g(x))} \quad (1)$$

where $D(h)$ is the support of the hypothesis.

As mentioned, the hypotheses are highly constrained and we express these constraints through graphs as in Figure 1. We denote the possibility that a graph G generates a hypothesis h by $G \Rightarrow h$. Thus our recognizers seek to compute

$$H^*_G = \underset{\{h|G \Rightarrow h\}}{\arg\max}\, S(h) \quad (2)$$

By normalizing by the background model, P_0, in Eqn. 1, the result $S(h) = 0$ means that h explains the pixels no better or worse than the background model (iid sample from the overall gray-level distribution), thus calibrating the interpretation of our scoring function and providing a natural threshold for detection. This data model is more explicitly explained in our previous works [9, 22].

2.1 Label Constraints

When the user labels a pixel, x, with a primitive (the most fundamental unit that constitutes a symbol), l, such as stem, flag, open note head, etc., we create a constraint of the form (x, l). If several such user labelings are required to correctly recognize a symbol we have a collection of constraints of the form $L = \{(x_i, l_i) : i = 1, \ldots, n\}$. Each time we get a new constraint the system re-recognizes the current candidate *subject* to these user-imposed constraints. Thus we modify our original scoring function to be

$$S(h) = \sum_{x \in D(h)} \log \frac{P_{M_h(x)}(g(x))}{P_0(g(x))} + t(x, Q_h(x)) \quad (3)$$

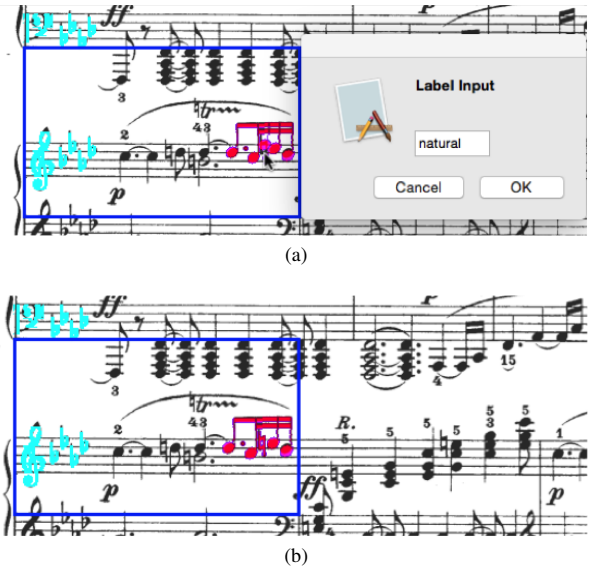
where

$$t(x, Q_h(x)) = \begin{cases} -\infty & x = x_i, Q_h(x) \neq l_i \text{ some } i \\ 0 & \text{otherwise} \end{cases}$$

Here $Q_h(x)$ represents the primitive label of hypothesis h at location x, thus $t(x, Q_h(x))$ disallows h if the pixel label is inconsistent with the hypothesis.

Fig. 2 illustrates a use case for a label constraint. In this example the original recognition, shown in the top panel of the figure, misidentifies the natural sign modifying the 'e' as an additional note head. In the top panel the user clicks on this pixel, labeling it with the "natural" primitive type. The bottom panel shows that re-recognition subject to the constraint fixes the problem.

In addition to primitive labels, the system also allows the user to label a rectangle of pixels as "white space", thus disallowing the recognizer from covering these pixels with any primitive symbol. In practice this turns out to be one of the most powerful constraints as it addresses the common case in which our recognition "spills over" into adjacent symbols.



(a)

(b)

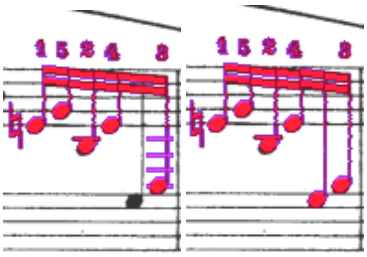
Figure 2: (a) During the initial recognition the natural was misidentified as a note head. The user is adds the correct primitive label to any location within the the natural. (b) Re-recognition correctly identifies the whole symbol by using the user-imposed label constraint.

2.2 Model Constraint

Model constraints change the graph, G, over which we optimize. Our interface contains a number of toggle switches each enabling a special case of recognition thus enlarging the graph. In general, the recognition works best when the minimal possible graph is chosen, though in many cases an overly permissive graph structure still produces the desired result without help from the user. In this case we still optimize Eqn. 2, though with a new graph, G', playing the role of G.

Fig. 3 gives an example in which an inappropriately restrictive graph generates a result with many primitive errors (top panel). In this case the original recognition was done without allowing note and chords to span the grand staff, as they do in this example. The result completely misses the penultimate note in the beamed group, while recognizing the last note with extraneous ledger lines which would be syntactically necessary when the note belongs to the upper staff. After enabling the grand staff ability we get the correct result in the bottom panel of the figure.

After the interactive recognition of a symbol is complete the user can save the symbol. When this occurs, we reset the list of pixel constraints placed by the user — these do not carry forward to future recognition tasks. The pixels involved in the recognized hypothesis are considered *unavailable* in subsequent symbol recognition, thus express-



(a) (b)

Figure 3: (a) Recognition using a beamed group graph that does not allow notes to span the grand staff. (b) Recognition using grand-staff beamed group graph.

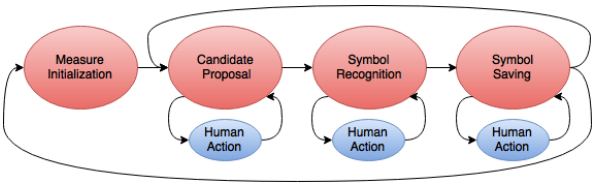


Figure 4: Human-driven (blue) and Machine-driven (red) actions in *Ceres* system. The possible state transitions are shown by arrowed connections.

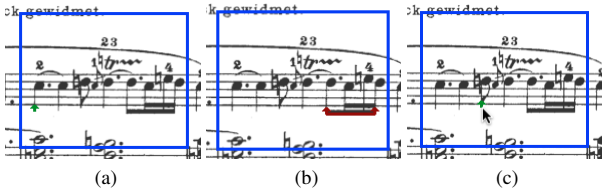
ing the notion that the symbols cannot touch. Of course, symbols do sometimes touch in practice, so we allow the user to label a rectangle of pixels as "reuse," thus allowing the user to override the basic non-overlapping constraint when needed.

3. USER INTERFACE

Our interface allows the user to control and direct the recognition process. The overall process is organized in terms of measures, while the interaction flow within a measure is described in the "action graph" of Fig. 4.

In the *candidate proposal* step (2nd level of the figure), the user can switch between the different candidate types (beamed group, isolated chord, slur, dynamics, etc.). Within each candidate type the user is presented with a left-to-right sequence of candidates she may choose to recognize or skip over. The color of the highlighted candidate reflects the candidate's type, also showing the direction of the stem, beam, and slurs with arrow signs, as this information is needed by the recognizers. The interface for this phase is shown in Fig. 5.

After a candidate is chosen, the system moves to the *symbol recognition* step. In this step, the system collaborates with user to improve the recognition in an iterative process. In each iteration the user either accepts the current recognition or imposes a new constraint (label or model), as discussed in Section 2. For a labeling constraint, the user inputs the pixel labels through a message box after clicking on the desired pixel position, as in Fig. 2a. For a model constraint, the user changes the current settings on the checkboxes or pull-down menu. The interface is shown in Fig. 6.



(a) (b) (c)

Figure 5: (a) The system detects and indicates a stem-up chord candidate for the user; (b) The system detects and indicates a stem-up beamed group candidates for the user; (c) The user adds a missing chord candidate.

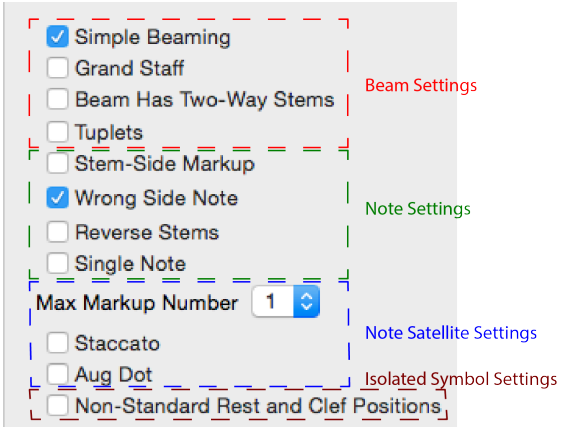


Figure 6: Checkboxes and pull-down menu for different model settings.

The system uses different colors to distinguish the current symbol from saved symbols. When the user wants to revisit an incorrectly saved symbol, she can select and delete the symbol before redoing the recognition.

Ceres has shortcut keys designed for the user to move from one step to another one, and also has a "cancel" key that allows the user to exit this process while moving to a "default" state. These interface units together constitute the visible part of *Ceres'* human-in-the-loop system.

4. EVALUATION

We evaluate our system both in terms of accuracy and speed. Both criteria are important since we believe the most interesting applications of OMR require accuracy on par with published scores, while it won't be possible to create large quantities of such data through OMR unless the process is highly efficient.

We measure accuracy here at the primitive level (beams, flags, note heads, accidentals, etc.), rather than, say, in terms of pitch and rhythm as in [16]. We prefer primitive evaluation because it is generic (all symbols are evaluated in the same way), it allows for all symbols our recognizer treats — not just those carrying pitch and rhythm information, and it is easy to relate primitive error analysis to specific aspects of the recognition engine.

The test set consists of first three pages of the *Breitkopf and Härtel* 1862-90 edition of Beethoven's Piano

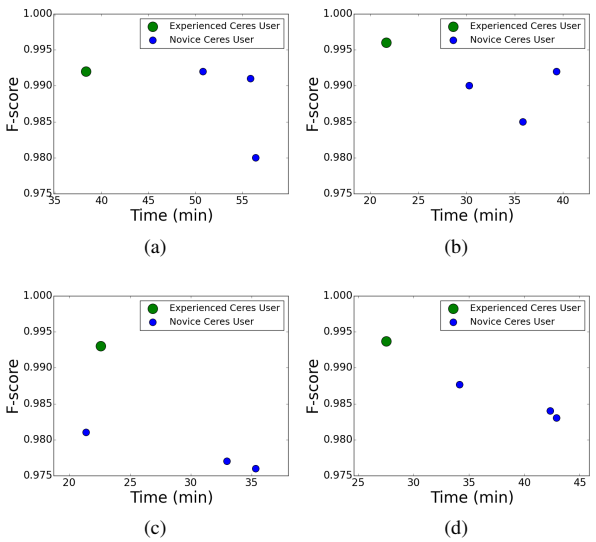


Figure 7: Clock time versus accuracy for novice and experienced *Ceres* users to generate one page: (a) page 1, (b) page 2, (c) page 3, (d) average performance.

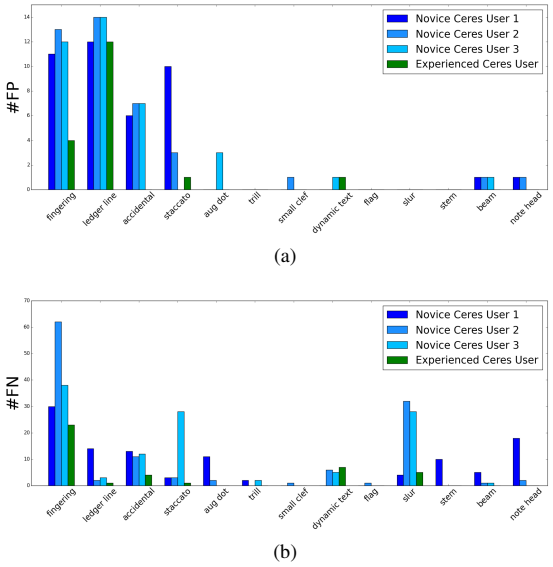


Figure 8: Distribution of *Ceres*-user-generated (a) false positive and (b) false negative errors with respect to their primitive labels on all the three pages.

Sonata No. 23, (the "Appassionata"), having 1606, 1501, and 1651 primitives respectively. We annotated the symbol primitives by hand with an interactive tool, thus creating our ground truth. In doing this we left out a few symbols appearing in the document that our system does not yet handle: grace notes, text and *fermati*. Our ground truth accounts for the overwhelming majority of what appears on the pages.

After recognition we decompose our structured results into an unstructured list of primitives, counting both false positives and false negatives. A recognized symbol counts as a false positive if its distance to all ground truth primitives of the same type is greater than some threshold. Analogously, a false negative occurs if a ground truth primitive is not sufficiently close to any recognized primitive. All symbol-to-symbol distances are measured in terms of a fixed reference point on each symbol.

Our subjects contained both novice users having about an hour of training, and more experienced users who were involved in the development of the user interface. Figure 7 shows both clock time and accuracy (an F-score) measured for these test subjects separately on each page. The figure shows overall error rates in the range of 1%, short of our eventual goal, but also showing that highly accurate results are within reach. The effect of user experience is evident both in accuracy and speed, though it is worth noting that the novice users were still able to get usable results from *Ceres*.

A primitive-level breakdown of errors is detailed in Figure 8, which counts both false positives and negatives by each class and user. A number of the errors are due to small symbols, such as augmentation dots, *staccato* markings, and short slurs. As illustrated in Section 3, our system superimposes the recognized results over the original image, usually making recognition errors obvious, though they are occasionally hard to see with small symbols. This highlights the need to explore better ways of visualizing results. The fingering errors were mostly due to our system's inability to recognize markups in non-standard positions such as to the side of a note head — an issue we have since accounted for. One can also see that a number of the errors come from ledger lines. This is due to a bug causing our system to occasionally produce syntactically impossible configurations of these primitives. These observations show the virtue of primitive-based evaluation since the errors are easily traced to their root causes.

We wanted to compare with a system other than our own, though between-system comparisons in OMR are challenging due to differences in the representations of both intermediate and end results. While commercial score-writing programs have different goals than OMR systems, both create symbolic representations of music documents that can be used to generate score images. Aside from these basic similarities there are a great many differences that may call comparisons into question. Still, in order to gain a point of reference for evaluation we compared our results with the commercial score-writing program, *Sibelius*.

Due to the steep learning curve involved with this program we engaged a professional *Sibelius* user with many years of professional experience, charging him with the task of recreating the original notation from scratch according to our test images. Even when directed otherwise, music copyists can substitute equivalent or nearly equivalent notation making it impossible to find a one-to-one correspondence between primitives. Thus we could not make meaningful accuracy comparisons with *Sibelius*.

However, we can compare the time to create the symbolic results as in Figure 9. This figure shows the necessary clock time to create the various pages. The results vary

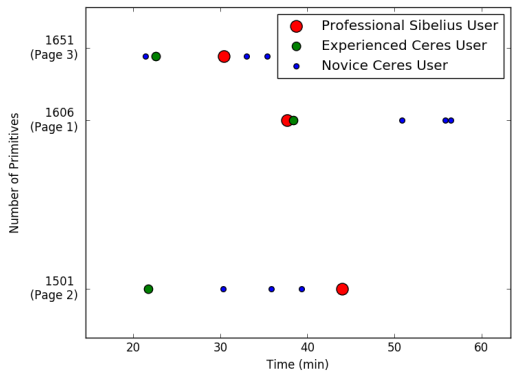


Figure 9: Clock time versus number of primitives for each page. Each point represents a single user from one of the following three groups: professional Sibelius user, experienced Ceres user or novice Ceres user.

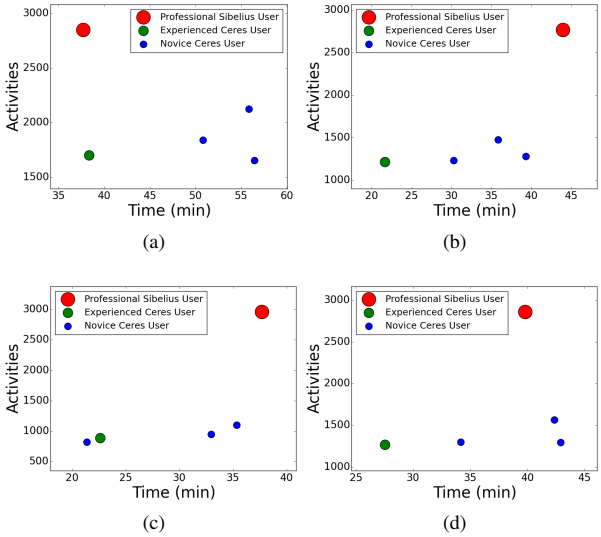


Figure 10: Clock time versus number of mouse and key activities used for each page: (a) page 1, (b) page 2, (c) page 3, (d) average performance. Each point represents a single user from one of the following three groups: professional Sibelius user, experienced Ceres user or novice Ceres user.

significantly from page to page, but show *Ceres* as competitive in all cases, and on two out of three pages showing significantly faster results than *Sibelius* with far less experienced users. Fig. 10 shows a similar comparison using keystrokes and mouse clicks as the measure of user effort. Here *Ceres* is seen to be considerably more efficient than *Sibelius* producing the results with much less user activity. This is because notation programs require detailed supervision while our system offloads as much work as possible to the machine. We also see from this figure that the experienced *Ceres* user makes more efficient strategies by minimizing the number of mouse and key actions.

The representations produced by these two systems have little in common, with different strengths and weaknesses for generating notation. *Sibelius* understands more of the inherent relations between symbols which must be preserved under renotation. *Ceres* captures a great deal of information about notational decisions (groupings, stem directions, spacing, etc.) which is also useful in renotation. The images at the website below [1] show notation generated from the two representations thus allowing for a subjective comparison of the two representations for renotation.

5. CONCLUSION

We have proposed a human-in-the-loop scheme for OMR that addresses several of the core difficulties of OMR. By allowing the user to select parameters of the models and symbol vocabulary, we deal with the heavy tail of rare symbols and notational exceptions. We also address the fundamental challenge of segmentation through recognition — now facilitated by a human guide. Finally, we demonstrate a feasible means to achieve the level of accuracy we believe is essential for successful application of OMR. The experiments show that our system has the potential to produce high-quality symbolic data more efficiently than a score-writing system, though we believe we must still improve significantly on this benchmark for our system to gain acceptance.

One promising application of *Ceres*-generated data is *renotation*. The current renotated pages are basically a one-to-one reconstruction of the original score, essentially denoising the image. We continue to explore more general renotation problems allowing various transformations of existing notation such as reformatting into arbitrarily-sized rectangles, transposition, and construction of parts from a score. This line of work will facilitate the flexible rendering of scores for electronic music readers.

We also remain engaged with improving the performance of our system. On one hand, we must continue to refine the core recognition abilities of our system, as these promise to improve both accuracy and speed of our system by handling a greater proportion of the work through automatic means. On the other hand we see considerable room for improvement and creativity in constructing the interface. We are interested in intelligent interactive approaches that increase the efficiency of the approach, e.g. automatic planning of human-machine-collaborated actions to minimize time cost, active prediction of human labeling and model selection, intelligent aggregation of multiple constrains through MIDI keyboard input, or adaptive learning to better recognize new documents. These interesting discussions are a part of our future plan.

[1] http://music.informatics.indiana.edu/papers/ismir16/

6. REFERENCES

[1] Sharpeye. `http://www.music-scanning.com/sharpeye.html`.

[2] Smartscore. `http://www.musitek.com`.

[3] P. Bellini, I. Bruno, and P. Nesi. Assessing optical music recognition tools. *Computer Music Journal*, 31(1):68–93, 2007.

[4] Taylor Berg-Kirkpatrick and Dan Klein. Improved typesetting models for historical ocr. In *ACL (2)*, pages 118–123, 2014.

[5] H. Bitteur. Audiveris. `https://audiveris.kenai.com`, 2014.

[6] D. Blostein and H. S. Baird. A critical survey of music image analysis. In H. Bunke H. S. Baird and K. Yamamoto, editors, *Structured Document Image Analysis*, pages 405–434. 1992.

[7] Nicholas J Bryan, Gautham J Mysore, and Ge Wang. Source separation of polyphonic music with interactive user-feedback on a piano roll display. In *ISMIR*, pages 119–124, 2013.

[8] Donald Byrd and Megan Schindele. Prospects for improving omr with multiple recognizers. In *ISMIR*, pages 41–46, 2006.

[9] Liang Chen and Christopher Raphael. Human-directed optical music recognition. *Electronic Imaging*, 2016(17):1–9, 2016.

[10] G. S. Choudhury, T. DiLauro, M. Droettboom, I. Fujinaga, B. Harrington, and K. MacMillan. Optical music recognition system within a large-scale digitization project. In *ISMIR*, 2000.

[11] B. Couasnon and B. Retif. Using a grammar for a reliable full score recognition system. In *ICMC*, pages 187–194, 1995.

[12] H. Fahmy and D. Blostein. A graph-rewriting paradigm for discrete relaxation: Application to sheet-music recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(6):763–799, 1998.

[13] I. Fujinaga. Optical music recognition system which learns. In *Proceedings of the SPIE - The International Society for Optical Engineering*, volume 1785, pages 210–17, 1993.

[14] I. Fujinaga. Exemplar-based learning in adaptive optical music recognition system. In *ICMC*, volume 12, pages 55–56, 1996.

[15] Ferguson G. and Allen G. Trips: An intelligent integrated problem-solving assistant. In *Proc. of the 15th National Conf. on Art. Intel.*, pages 567–573, 1998.

[16] Rong Jin and Christopher Raphael. Interpreting rhythm in optical music recognition. In *ISMIR*, pages 151–156, 2012.

[17] G. Jones, B. Ong, I. Bruno, and K. Ng. Optical music imaging: Music document digitisation, recognition, evaluation, and restoration. *Interactive Multimedia Music Technologies*, pages 50–79, 2008.

[18] G. E. Kopec, P. A. Chou, and D. A. Maltz. Markov source model for printed music decoding. In *Proceedings of the SPIE - The International Society for Optical Engineering*, volume 2422, pages 115–25, 1995.

[19] Myers K. L., Jarvis P. A., Tyson W. M., and Wolverton M. J. A mixed-initiative framework for rubust plan sketching. In *Proc. 13th Int. Conf. on Automated Planning and Scheduling*, pages 256–265, 2003.

[20] K. C. Ng and R. D. Boyle. Recognition and reconstruction of primitives in music scores. *Image and Vision Computing*, 14(1):39–46, 1996.

[21] L. Pugin, J. A. Burgoyne, and I. Fujinaga. Map adaptation to improve optical music recognition of early music documents using hidden markov models. In *ISMIR*, pages 513–516, 2007.

[22] Christopher Raphael and Jingya Wang. New approaches to optical music recognition. In *ISMIR*, pages 305–310, 2011.

[23] A. Rebelo, G. Capela, and J. S. Cardoso. Optical recognition of music symbols. *International Journal on Document Analysis and Recognition*, 13:19–31, 2009.

[24] Verónica Romero, Alejandro H Toselli, Luis Rodríguez, and Enrique Vidal. Computer assisted transcription for ancient text images. In *Image Analysis and Recognition*, pages 1182–1193. 2007.

[25] F. Rossant and I. Bloch. Robust and adaptive omr system including fuzzy modeling, fusion of musical rules, and possible error detection. *EURASIP Journal on Applied Signal Processing*, 2007.

[26] Branson S., Wah C., Babenko B., Schroff F., Welinder P., and B̃elongie S. Perona P. Visual recognition with humans in the loop. In *ECCV*, 2010.

[27] Cox M. T., Edwin G., Balasubramanian K., and Elahi M. Multiagent goal transformation and mixed-initiative plannng using prodigy/agent. In *Proc. 4th Int. Multiconf. on Systemics, Cybernetics and Informatics*, volume 7, pages 1–6, 2001.

[28] Cox M. T. and Veloso M. M. Supporting combined human and machine planning: An inteface for planning by analogical reasoning. In *Proc. 2nd Int. Conf. on Case-Based Reasoning*, pages 531–540, 1997.

[29] V. Viro. Peachnote: Music score search and analysis platform. In *ISMIR*, pages 360–363, 2011.

[30] Alexander Waibel, Rainer Stiefelhagen, Rolf Carlson, J Casas, Jan Kleindienst, Lori Lamel, Oswald Lanz, Djamel Mostefa, Maurizio Omologo, Fabio Pianesi, et al. Computers in the human interaction loop. In *Handbook of Ambient Intelligence and Smart Environments*, pages 1071–1116. 2010.

I SAID IT FIRST: TOPOLOGICAL ANALYSIS OF LYRICAL INFLUENCE NETWORKS

Jack Atherton and **Blair Kaneshiro**
Center for Computer Research in Music and Acoustics, Stanford University
{lja, blairbo}@ccrma.stanford.edu

ABSTRACT

We present an analysis of musical influence using intact lyrics of over 550,000 songs, extending existing research on lyrics through a novel approach using directed networks. We form networks of lyrical influence over time at the level of three-word phrases, weighted by tf-idf. An edge reduction analysis of strongly connected components suggests highly central artist, songwriter, and genre network topologies. Visualizations of the genre network based on multidimensional scaling confirm network centrality and provide insight into the most influential genres at the heart of the network. Next, we present metrics for influence and self-referential behavior, examining their interactions with network centrality and with the genre diversity of songwriters. Here, we uncover a negative correlation between songwriters' genre diversity and the robustness of their connections. By examining trends among the data for top genres, songwriters, and artists, we address questions related to clustering, influence, and isolation of nodes in the networks. We conclude by discussing promising future applications of lyrical influence networks in music information retrieval research. The networks constructed in this study are made publicly available for research purposes.

1. INTRODUCTION

Lyrics have been used to study many topics in music information retrieval (MIR) including genre classification [6], hit prediction [9], similarity searching [10], cultural studies [4], and computational musicology [5]. One approach to lyrical analysis is the *bag-of-words* model, which considers word frequencies in a text irrespective of order. In 2004, Logan et al. used this approach to produce promising preliminary results for measuring artist similarity through topic models, and observed that some genres naturally group with others based on shared vocabulary [9]. Fell and Sporleder later found that some genres (e.g. Rap, Metal) have relatively unique vocabularies, while others (e.g. Folk, Blues, and Country) cluster into groups [6]. Most recently, Ellis et al. computed bag-of-words novelty of lyrics and found that top-100 music was less lexically novel than less popular music [5].

In contrast, *n-gram* models consider ordered phrases of n words. A. Smith et al. used trigrams (n-grams with $n = 3$) and rhyme structures to develop a metric for cliché in lyrics, finding that number-one hits were more clichéd than average songs [14]; here, trigrams proved to be the better metric for measuring cliché. They also inspected their data by genre and found that genres had generally unique most-frequent phrases, though some phrases were shared by many genres. Later, Fell and Sporleder developed a suite of lexical features and used these, along with n-grams, to achieve performance gains in various classification tasks, but confirmed that n-grams alone achieved satisfactory baseline performance [6].

Networks—or graphs—are a natural and increasingly prevalent tool for analyzing structure between musical entities such as artists, songwriters, and genres. Networks comprise sets of nodes and sets of edges, or relations, between nodes. Most networks use unweighted, undirected edges, whereby edges are binary measures of whether two nodes are connected. Weighted edges ascribe varying importance to the relationships, and directed edges give each relationship a direction or flow. A 2006 study by R. Smith revealed the community structure of rappers by constructing a network between rappers who collaborated [15]. This study weighted edges by frequency of collaboration and found that different groupings such as large communities, music labels, and groups such as the famous Wu-Tang Clan emerged when varying percentages of the least significant edges were removed from the graph. We will refer to this process here as *edge reduction*. Collins later considered the flow of musical influence in synth-pop music [3], while Gunaratna et al. built a collaboration network for Brazilian musicians and composers [8]. Finally, Bryan and Wang constructed a directed graph connecting genres based on sampling of musical content, showing that Funk, Soul, and Disco heavily influence many modern popular genres [2].

Both lyrics and graphs allow us to ask deep questions about similarity, popularity, and interconnectedness in the music landscape. To our knowledge, no study has formed lyrics-based networks to analyze musical relationships formed directionally over time, although Fujihara et al. created a system for hyperlinking between lyrics and from lyrics to audio [7]. The current study combines lyrical analysis with graphs to observe influence of genres, artists, and writers, through networks formed by re-use of

© Jack Atherton and Blair Kaneshiro. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jack Atherton and Blair Kaneshiro. "I Said It First: Topological Analysis of Lyrical Influence Networks", 17th International Society for Music Information Retrieval Conference, 2016.

lyrics. Our networks' directional edges highlight lyrical influence over time, allowing us to address questions like that raised by A. Smith et al. of whether Pop music makes use of existing clichés or creates new ones [14]. We examine the topology of influence networks of genres, artists, and writers to quantifiably assess grouping behavior, robustness of connections, and centrality within the networks.

The remainder of this paper is structured as follows: first, we explain the formation of the influence networks and the computation of several of their properties. Next, we demonstrate and visualize the topology of the networks. Finally, we show how the basic building blocks of network properties can be used to address many outstanding musicological and MIR research questions.

2. METHODS

2.1 Data Sources

Past research has shown that n-grams (phrases) are superior to bag-of-words (vocabulary) for lyrical MIR tasks such as classification and computational musicology [6, 14]. With this in mind, we obtained intact lyrics data and artist/writer metadata via a signed research agreement with LyricFind, whose data were used previously in the Ellis et al. bag-of-words study on lexical novelty [5]. We additionally obtained primary genre and release date at the album level through the free iTunes Search API. [1] After filtering out songs with no lyrics, as well as those with no entry in the sparse iTunes dataset, we were left with 554,206 songs, collectively representing 42,802 artists, 95,349 writers, and 214 genres.

2.2 Constructing the Networks

The first step was to exhaustively measure the trigrams present in every song. Phrases were considered equivalent if they were cleaned to the same base phrase. We cleaned the lyrics using a procedure previously validated by Ellis et al. [5], avoiding stemming the words and using their rules for misspellings, alternate spellings, hyphens, and slang (modified to avoid expanding contractions and to correct a few inaccurate slang terms). We also filtered out *stopwords*—words too common to impart any lexical significance (e.g. pronouns and articles). We used a list of English stopwords from the Natural Language Toolkit, [2] augmented with many of the contractions ignored in the cleaning phase and misspellings of stopwords common to the LyricFind data. If a phrase was reduced in size due to stopword removal, we added an additional word and repeated the process until we obtained a cleaned trigram with no stopwords. This process allowed us to consider and match phrases originally longer than three words on the basis of only semantically significant words; for example, two four-word phrases that differed only by a specific pronoun (e.g. "he" / "she") would be matched after stopword removal. After initial results revealed spurious phrases created across lines of lyrics, we modified the algorithm to search for phrases only within lines of lyrics.

Next, songs were separated by year of release date in order to compute their phrases' term frequency-inverse document frequency (*tf-idf*). Tf-idf is a robust measure of significance common to information retrieval that increases an item's weight if it is common within its document and decreases its weight if it is common across the dataset [11]. Past MIR studies have used tf-idf for automatic mood classification of lyrics [16]—also in conjunction with rhyme information [17]—and to measure lexical novelty [5]. To capture the changing significance of phrases over time, we treated each individual year as a separate dataset. This way, the first person to use a phrase would have a significantly higher idf for that phrase than a person using it when it is already popular. Tf-idf is computed as in equation (1), where n_p is the number of occurrences of a phrase p in a song, n_s is the number of phrases in that song, s_y is the number of songs in a year y, and s_p is the number of songs in that year containing p.

$$\text{tf-idf}(p) = \frac{n_p}{n_s} \cdot \log(\frac{s_y}{s_p}) \tag{1}$$

We then constructed the three influence networks, one each for genres, artists, and writers. For every phrase in the dataset, we generated a list of all pairs of songs sharing that phrase. Pairs of songs released in the same year were ignored. This limit sets a minimum on the time difference necessary before a repeated phrase is considered influential, and also avoids forming links between potential duplicate entries in the dataset. For pairs of songs occurring in different years, we formed an edge from the earlier song to the later one. The edge's weight was the product of the tf-idfs of the phrase in both songs in order to capture the significance of the phrase in both years. Using song metadata, we then added the edge to the genre, artist, and writer graphs. For example, if a phrase was used in a Rock song in 1990 and in a Pop song in 1991, the resulting edge was drawn from the Rock node to the Pop node in the genre graph. If either song had multiple artists or writers, we added edges between all possible pairings. If multiple edges were added between two nodes, they were combined by summing their weights.

Next, *influence* scores were computed for each node of each graph. The influence I_i of node i is defined as the ratio of the sum of its outgoing edge weights (e_{ij}) to the sum of its incoming edge weights (e_{ji}):

$$I_i = \frac{\sum_j e_{ij}}{\sum_j e_{ji}} \tag{2}$$

Influence is thus a measure of the degree to which a node impacted future work or quoted previous work, but does not depend on the node's total volume of work.

Genre diversity scores were computed for each artist and writer as the number of genres they are credited in, divided by their total number of songs. This gives a measure

[1] http://apple.co/1qHOryr
[2] http://www.nltk.org/

of how many genres the person has contributed to without being skewed by their total number of contributions. [3]

Finally, each node's *self-reference* score was computed as the weight of the edge pointing from that node to itself, divided by that node's total number of contributions. This normalization again avoids skewing the score by volume, as edge weights were formed by summing over all possible pairings of phrases shared between two nodes.

The graphs were constructed with Graph-tool. [4] We make the graph data from this study publicly available for research purposes [1].

2.3 Network Analyses

The graphs were first assessed for clustering behavior. Adapting the method used by R. Smith, we analyzed the graphs in stages while removing increasing percentages of the least significant edges [15]. The first, *global* edge reduction method removed the $X_g\%$ lowest-weighted edges across the entire the graph, with X_g ranging from 0 to 99. The second, *local* method removed edges from each node that have a weight less than $X_l\%$ of the strongest weight at that node. At each stage of both procedures, the graphs were analyzed for their strongly connected components in order to examine the grouping behavior of the nodes.

Next, we performed *multidimensional scaling* (MDS) on the genre graph in order to embed its nodes in two dimensions for visualization. MDS converts a set of pairwise dissimilarities among objects into coordinates that can be used to visualize the objects in a low-dimensional space [13]. Here, the dissimilarities were computed using mutual influence D_{ij} below, where e_{ij} is the weight between nodes i and j. Graph visualizations were performed using Gephi. [5]

$$D_{ij} = \log(e_{ij} * e_{ji})^{-1} \tag{3}$$

Finally, we computed a series of correlations between the various metrics defined above, as well as the in-degree, out-degree, and average tf-idf of incoming and outgoing edges of each node. The r and p values were computed with the Scipy Statistics *pearsonr* function. [6]

3. RESULTS

3.1 Most Common Phrases

Table 1 shows the phrases used across the highest number of years. Many of these phrases are considered timeless, and all are semantically significant. Also, no phrase occurred in every year. The repetition of "dream(s) come true" does suggest that using word stems might improve performance. We note also that pronouns and other stopwords are absent from all phrases, which allowed the consideration of longer phrases with internal stopwords. For example, "makes feel like" is a combination of "makes (me) feel like," "makes (you) feel like," and other similar phrases. Overall, we treat these results as verification that our cleaning procedure was adequate.

Phrase	Years	Phrase	Years
dreams come true	49	one two three	43
never let go	48	late last night	42
new york city	46	whole wide world	42
long time ago	46	come back home	42
dream come true	45	makes feel like	41

Table 1: The most common phrases, ordered by number of years in which they appeared (maximum possible is 62).

3.2 Network Components

In our global edge reduction analysis, we expected that the graphs would split into several components. Instead, each graph remained concentrated in one large strongly connected component, with a few negligible side components. The size of the central component at a few points in the global edge reduction process is shown in Table 2.

The writer graph was the most robust: its central component was largest at nearly every point in the edge reduction process, and with only the top 1% of edges remaining in the graph ($X_g = 99$), it still contained nearly 200 components of 5 or more writers. We believe this result arose because many songs have multiple writers, while few songs have multiple artists; therefore, more relationships among writers would emerge from the same songs.

Using the local edge reduction method, a few small, significant components did break off of the main component. For example, with $X_l = 2$, the pairs {Celtic, Contemporary Celtic} and {Folk-Rock, Contemporary Folk} split off the main genre component. Table 3 shows the Brazilian and Spanish-speaking Latin American components formed with $X_l = 3$ and $X_l = 4$, respectively.

The graphs' strongly connected components split apart much more quickly using local edge reduction than with global edge reduction. At $X_l = 4$, the main component consisted of 21 genres; at $X_l = 20$, the only components remaining of size more than 1 were the two in Table 4. At $X_l = 25$, the main component had reduced to the pair {Rock, Pop}, but the Latin American component remained unchanged from $X_l = 20$. The answer to why the Latin American component was so robust probably lies in our data preparation method: since we did not filter out the stopwords of any language but English, the connections between genres of other languages were strengthened by spurious connections with no lexical significance. This result shows the importance of a cleaning procedure that works uniformly across the dataset.

Graph	$\mathbf{X_g = 0}$	$\mathbf{X_g = 90}$	$\mathbf{X_g = 99}$
Genre	95%	54%	13%
Artist	99%	52%	12%
Writer	98%	66%	28%

Table 2: Percentage of nodes in the central component with $X_g\%$ edge reduction.

[3] Un-normalized genre count did not significantly interact with any other variables in the study.

[4] https://graph-tool.skewed.de/

[5] https://gephi.org/

[6] http://docs.scipy.org/doc/scipy-0.16.0/reference/

$\mathbf{X_l = 3}$	$\mathbf{X_l = 4}$
MPB	Latin Pop
Sertanejo	Latin Alternative & Rock
Samba	Latino
Pagode	Salsa y Tropical
Axé	Regional Mexicano
	Baladas y Boleros
	Latin Urban

Table 3: Genres contained in two components that split from the main component with $X_l\%$ edge reduction.

Main Component	Latin American Component
Pop	Latin Pop
Rock	Latin Alternative & Rock
Alternative	Latino
R&B/Soul	Salsa y Tropical
Country	Regional Mexicano

Table 4: Components with $X_l = 20\%$ edge reduction.

3.3 Multidimensional Scaling and Visualizations

To explore the seemingly central nature of the graphs, we performed MDS and visualized the genre graph. [7] The visualization (Figure 1) promisingly showed that nearly all edges in the graph were focused toward the center.

To more closely observe the behavior at the center of the graph, we next visualized only the 28 genres that appear in the central component with global edge reduction ($X_g = 99$), and displayed only each node's top three incoming edges. Using dissimilarities computed from all edges (not just those present in the visualization), we performed MDS again to obtain the node positions for Figure 2 and Figure 3. Jazz, Pop, and Rock are firmly at the center of the genre network. Here, 21 of the 28 nodes in the central component include Jazz among their top three influences, while 16 include Pop and 13 include Rock.

3.4 Influence and Self Reference

We next sought a statistical explanation for the centrality of the highlighted nodes in Figure 1. Turning first to influence, the ratio of a node's outgoing to incoming edge weights, we expected that central genres would have high influence, meaning that many genres draw from the phrases used in the central genres. In fact, the extremes of the influence metric are dominated by outliers, including rare genres as well as artists and writers who appear only very early or very late in the dataset. These groups do not have much opportunity for incoming or outgoing edges, respectively. In contrast, central genres are referenced at about the same rate that they reference previous material, having influence scores close to 1.0. Figure 2 and Table 5 show the influence of central genres.

We then turned to the self-reference score, a measure of how much a genre re-uses lyrics from its past. Our intuition here was that genres that refer to themselves frequently create a particular subculture that is ripe for other genres to draw influence from. Table 5 shows the top 10

[7] Because of their sheer size, visualizations of the artist and writer graphs were not feasible at the time of the study.

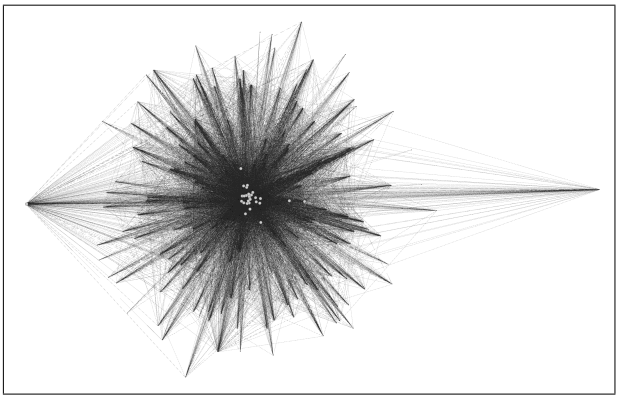


Figure 1: MDS layout of the genre graph, having 214 nodes and 17,438 edges. World, far left, is pushed out of the central area by its extreme dissimilarity to Pop Punk, far right. Light gray nodes are the 28 nodes in the central component with global edge reduction, $X_g = 99\%$.

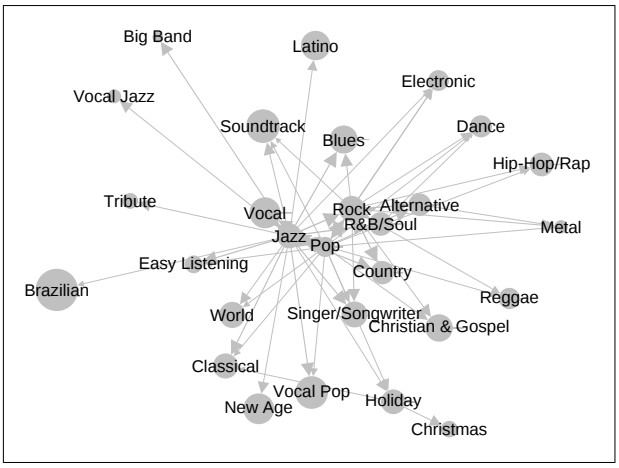


Figure 2: MDS layout of the central component of the genre graph. Node size denotes node influence, and arrow size denotes edge weight. Here, the sensitivity of the influence metric to outliers is shown (for example, with the large size of the Brazilian node).

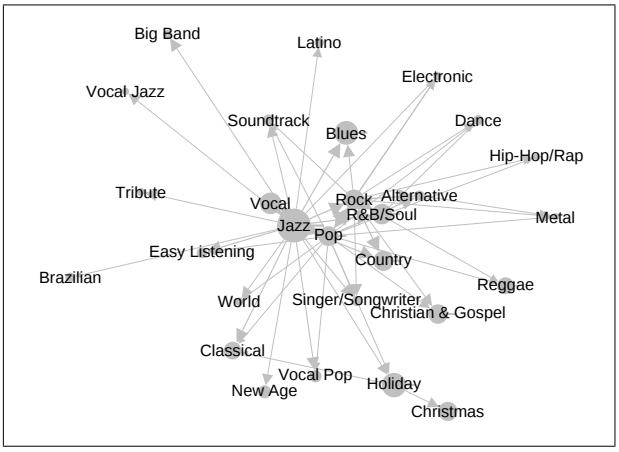


Figure 3: MDS layout of the central component of the genre graph. Node size denotes self-reference, and arrow size denotes edge weight. Self-reference, more than influence (Figure 2), correlates with centrality in the network.

Genre	Influence (%)	Self-reference (%)
Jazz	1.039 (65.3%)	**166.516** (**100**%)
Holiday	1.031 (64.5%)	82.276 (99.6%)
Blues	1.176 (73.6%)	7.698 (99.1%)
Vocal	**1.301** (**76.6**%)	6.818 (98.6%)
R&B/Soul	0.987 (63.2%)	6.369 (98.1%)
Country	0.974 (60.2%)	6.202 (97.7%)
Rock	1.083 (67.5%)	6.124 (97.2%)
Pop	0.836 (51.9%)	6.074 (96.7%)
Christian & Gospel	1.149 (71.9%)	5.883 (96.3%)
Christmas	**0.690** (**47.6**%)	**5.759** (**95.8**%)

Table 5: Top genres by self-reference. Raw values and percentiles are shown.

genres by self-referential behavior. These genres indeed correspond with genres near the center of the graph, and they are all in the central component of Figure 1. Jazz and Holiday have particularly outlying scores, perhaps reflecting that these genres often consist of *standards*—covers of widely known songs. Across the entire genre graph, log-self-reference correlates with centrality in the MDS graph (measured as the negative sum of euclidean distances from a node to all others), with $r = 0.610$, $p < 0.0001$.

After seeing the importance of self-reference in the genre graph, we computed this metric for the other two graphs. The list of top artists by self-reference is dominated by Jazz artists. This could reflect more the outlying nature of the genre than it does anything about Jazz artists. However, viewing the top artists by decade yields some interesting results when Jazz artists are ignored. For example, Sam Smith, Two Door Cinema Club, and Owl City are among the top-10 most self-referential artists of the 2010s.

3.5 Self-Reference and Volume

To better understand the metrics of self-reference and influence, we assessed their correlations with other aspects of the data. First, we determined the extent to which genre volume (number of songs in a genre) affects its self-reference. Using the un-normalized value of self-reference (i.e. the genre's raw self edge weight, not divided by the genre's volume), log-self-reference correlates highly with log-volume ($r = 0.846$, $p < 0.0001$). This is because the raw edge weight is a sum of all connections between songs of that genre, and more songs allow more connections. But, when self-reference is normalized by genre volume (see § 2.2), log-self-reference still correlates highly with log-volume ($r = 0.780$, $p < 0.0001$). Our intuition for this is that as the number of songs in a genre increases, the average quality of self-references increases, and so the normalized contribution from each song increases.

3.6 Genre Diversity, Influence, and Connectedness

Having investigated self-reference as a measure of phrase sharing within genres, we next assessed sharing of phrases across genres. We expected that people with high genre diversity are able to transfer phrases between genres, which would increase their influence score as the transferred phrases are referenced by people with less genre diversity.

Name	Influence (%)	Self-Reference (%)
Paul McCartney	0.970 (55.4%)	114.061 (99.8%)
John Lennon	0.974 (55.6%)	**119.660** (**99.8**%)
Max Martin	**0.421** (**31.7**%)	**0.739** (**70.4**%)
Mariah Carey	1.039 (59.6%)	3.595 (88.9%)
Barry Gibb	**1.111** (**62.4**%)	7.661 (95.0%)

Table 6: Top writers, ordered by number-one singles. Raw values and percentiles are shown.

However, we found that influence has a low, though statistically significant correlation with genre diversity ($r = 0.079$ for artists, $r = 0.155$ for writers, $p < 0.0001$).

Surprised by this result, we investigated further the effect of genre diversity on connections in the network. First, we investigated whether drawing influence from many genres correlates with more complex references and found a low, though statistically significant, correlation between writers' log-average incoming tf-idf value and genre diversity ($r = -0.167$, $p < 0.0001$). Next, we examined the degree to which writing in many genres correlates with directional influence forward and backward in time. We actually found a negative correlation between log-out-degree and genre diversity ($r = -0.572$, $p < 0.0001$), as well as between log-in-degree and genre diversity ($r = -0.563$, $p < 0.0001$). Thus, although influence (the ratio of outgoing to incoming edges) explains little genre diversity, increased genre diversity correlates moderately with less robust connections within the graph in both future and past directions. This could suggest that writers who contribute to a wider variety of genres use complicated phrases that are less likely to be shared with other writers, or that they use more stopwords that are filtered by the algorithm.

Artists showed similar correlation behavior to writers, but with lower correlation magnitudes, perhaps reflecting that artists are often a step removed from writing lyrics and may perform lyrics written by a variety of writers.

3.7 Top Genres, Writers, and Artists

We showed in § 3.5 that the volume of a genre in the dataset correlates highly with its self-reference score. Compared to other popular music genres, Rap has a particularly low self-reference score: it is the 6th most numerous genre in the data, but ranks 48th in self-reference. Similarly, Metal is the 8th most numerous genre in the data, but ranks 40th in self-reference. Rap's low self-reference score may reflect a particular subculture within this genre that values lyrical originality over references to past material.

Having analyzed lyrical influence between genres over time, we can now address whether Pop music is more clichéd than other genres because it draws from many sources or because it popularizes new phrases [14]. Rock and Pop are the two most common genres in our dataset. Rock's influence score is 1.083, while Pop's is only 0.836. Since Pop's influence is less than 1.0, Pop music quotes phrases from other genres more often than it influences them. This suggests that Pop music draws on existing clichés more than it creates new ones, especially when compared to other popular genres such as Rock.

Name	Influence	Self-Reference
The Beatles	**0.140** (**21.5**%)	2.699 (95.4%)
Elvis Presley	0.847 (56.1%)	**9.964** (**99.6**%)
Mariah Carey	1.688 (73.0%)	3.362 (96.6%)
Rihanna	0.381 (36.9%)	**1.114** (90.2%)
Michael Jackson	0.869 (56.8%)	6.527 (98.9%)
The Supremes	0.762 (53.6%)	5.068 (98.2%)
Madonna	**3.901** (**87.1**%)	1.344 (91.3%)
Whitney Houston	2.010 (76.5%)	2.369 (94.7%)
Stevie Wonder	0.247 (29.0%)	2.297 (94.5%)
Janet Jackson	1.162 (64.2%)	1.723 (92.8%)

Table 7: Top artists, ordered by number-one singles. Raw values and percentiles are shown.

The all-time top writers of number-one hits have influence scores close to 1, with more successful writers having slightly lower influence scores (Table 6). This result mirrors that found for central genres and suggests that these writers were well connected in a lyrical culture that they both contributed to and drew from. The exception is Max Martin, whose lower influence score perhaps reflects the less quotable nonsense phrases he often uses in songs [12]. Martin is also the least self-referential of the top writers, which might be explained by noting that he writes for a variety of artists with different styles, whereas the other writers are most famous for writing for themselves.

Table 7 shows the top 10 artists by number-one hits. In contrast to writer position, artist position does not seem strongly affected by influence score. However, all top artists fall into the top decile (10%) of self-reference. Also, female artists in the list have much higher influence scores than males, with the exception of The Supremes and Rihanna. We note also that Mariah Carey has a much higher influence score as an artist than as a writer. Further analysis of this phenomenon is complicated by the fact that many people use a pseudonym as an artist but not as a writer.

4. DISCUSSION

We explored topologies of genre, artist, and songwriter influence networks constructed from links between trigrams over time. Through edge reduction and strongly connected component analyses, we showed that all three graphs are highly centralized around a large component with robust links. We confirmed this organization with an MDS visualization of the genre graph based on mutual influence. Alternative methods of edge reduction revealed separate components, but primarily along language differences. We found that the best predictor of a genre's centrality to the influence network was the degree to which it referenced itself, and that the network especially centered around three popular and self-referential genres: Jazz, Pop, and Rock.

Our current metrics are useful building blocks for studying relationships between genres, artists, and writers. The centrality of our influence networks supplements earlier findings showing clustering between some genres and isolation of others [6, 9]. However, our data do not produce the Folk, Country, and Blues cluster observed by Fell and Sporleder [6]; rather, we find these genres have similar influences but do not draw significant influence from each other. Fell and Sporleder also found Rap and Metal to be lyrically isolated when analyzed with a bag-of-words model [6]. Our trigram analysis shows Rap and Metal to be well connected in the network, though their self-reference scores are low compared to other popular genres. Furthermore, the notion of lyrical novelty [5] can be approximated with influence, as the influence network incorporates novelty into the edge weights with tf-idf; cliché [14] can be understood as the inverse of novelty. Overall, our analyses do not suggest the same degree of genre segmentation suggested in prior studies. We conclude that significant differences between genres may not occur at the phrase level, but instead arise from key vocabulary differences [6, 9] as well as musical and sociocultural factors [2, 15].

There are several potential areas for improvement in the present study. First, a phrase shared across songs does not necessarily signify direct influence. Next, albums were occasionally labeled with incorrect years, which would impact the temporal dimension of our networks. We used primary iTunes album genre for our analysis, but acknowledge that such labels may not adequately characterize the songs, especially for non-Western music or for songs that conceivably belong to more than one genre. Also, the decrease in connection robustness from cleaning may not be uniform across genres. In particular, the present results could be refined by analyzing English lyrics only or by including stopwords and cleaning rules for other languages. The cleaning procedure we followed [5] may benefit from stemming. Finally, the decision to ignore phrases spanning lines of lyrics reflects the organization of phrases for most genres, but may have broken up phrases from genres with more complicated lyrics (such as Rap).

Our findings highlight exciting possibilities for future research. Recall that at one point in our component analyses, Jazz was excluded from the central component comprising Pop and Rock—not because it was less influential, but because it drew much less influence from Pop and Rock than they drew from it, leaving no path back to Jazz from Pop or Rock when only the highest-magnitude edges remained in the graph. Future analyses could examine whether the edge reduction component analysis aligns more closely with self-reference when graphs are treated as undirected. Future work could also find cliques, which would be more robustly interconnected than strongly connected components. Assessing changes in network structure when higher-order n-grams are used is another topic that can be explored in future research. Finally, future studies could examine further the notion of robustness of connection; differences in influence when people act as artists versus writers; trends that emerge when people are grouped by gender, race, or geolocation; network topologies when rare genres are grouped into categories; and could contribute visualizations to help understand the structure of artist and writer networks.

Acknowledgements: The authors thank Will Mills from LyricFind for facilitating access to the lyrics data; and Casey Baker and Ge Wang for their helpful feedback.

5. REFERENCES

[1] J. Atherton and B. Kaneshiro. Lyrical influence networks dataset (LIND). In *Stanford Digital Repository*, 2016. http://purl.stanford.edu/zy061bp9773.

[2] N. J. Bryan and G. Wang. Musical influence network analysis and rank of sample-based music. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 329–334, 2011.

[3] N. Collins. Computational analysis of musical influence: A musicological case study using MIR tools. In *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pages 177–182, 2010.

[4] C. N. DeWall, R. S. Pond Jr, W. K. Campbell, and J. M. Twenge. Tuning in to psychological change: Linguistic markers of psychological traits and emotions over time in popular US song lyrics. *Psychology of Aesthetics, Creativity, and the Arts*, 5(3):200–207, 2011.

[5] R. J. Ellis, Z. Xing, J. Fang, and Y. Wang. Quantifying lexical novelty in song lyrics. In *Proceedings of the 16th International Society for Music Information Retrieval Conference*, pages 694–700, 2015.

[6] M. Fell and C. Sporleder. Lyrics-based analysis and classification of music. In *Proceedings of the International Conference on Computational Linguistics*, pages 620–631, 2014.

[7] H. Fujihara, M. Goto, and J. Ogata. Hyperlinking lyrics: A method for creating hyperlinks between phrases in song lyrics. In *Proceedings of the 9th International Conference on Music Information Retrieval*, pages 281–286, 2008.

[8] C. Gunaratna, E. Stoner, and R. Menezes. Using network sciences to rank musicians and composers in Brazilian popular music. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 441–446, 2011.

[9] B. Logan, A. Kositsky, and P. Moreno. Semantic analysis of song lyrics. In *IEEE International Conference on Multimedia and Expo*, volume 2, pages 827–830, 2004.

[10] J. P. G. Mahedero, A. Martínez, P. Cano, M. Koppenberger, and F. Gouyon. Natural language processing of lyrics. In *Proceedings of the 13th annual ACM International Conference on Multimedia*, pages 475–478, 2005.

[11] G. Salton and M. J. McGill. *Introduction to modern information retrieval*. McGraw-Hill, Inc., New York, 1986.

[12] J. Seabrook. Blank space: What kind of genius is Max Martin? *New York Times*, Sep. 30, 2015.

[13] R. N. Shepard. Multidimensional scaling, tree-fitting, and clustering. *Science*, 210(4468):390–398, 1980.

[14] A. Smith, C. Zee, and A. Uitdenbogerd. In your eyes: Identifying clichés in song lyrics. In *Australasian Language Technology Workshop*, pages 88–96, 2012.

[15] R. D. Smith. The network of collaboration among rappers and its community structure. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(02):2–15, 2006.

[16] M. van Zaanen and P. Kanters. Automatic mood classification using TF*IDF based on lyrics. In *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pages 75–80, 2010.

[17] X. Wang, X. Chen, D. Yang, and Y. Wu. Music emotion classification of Chinese songs based on lyrics using TF*IDF and rhyme. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 765–770, 2011.

IMPACT OF MUSIC ON DECISION MAKING IN QUANTITATIVE TASKS

Elad Liebman
Department of Computer Science
The University of Texas at Austin
eladlieb@cs.utexas.edu

Peter Stone
Department of Computer Science
The University of Texas at Austin
pstone@cs.utexas.edu

Corey N. White
Department of Psychology
Syracuse University
cnwhite@syr.edu

ABSTRACT

The goal of this study is to explore which aspects of people's analytical decision making are affected when exposed to music. To this end, we apply a stochastic sequential model of simple decisions, the drift-diffusion model (DDM), to understand risky decision behavior. Numerous studies have demonstrated that mood can affect emotional and cognitive processing, but the exact nature of the impact music has on decision making in quantitative tasks has not been sufficiently studied. In our experiment, participants decided whether to accept or reject multiple bets with different risk vs. reward ratios while listening to music that was chosen to induce positive or negative mood. Our results indicate that music indeed alters people's behavior in a surprising way - happy music made people make better choices. In other words, it made people more likely to accept good bets and reject bad bets. The DDM decomposition indicated the effect focused primarily on both the caution and the information processing aspects of decision making. To further understand the correspondence between auditory features and decision making, we studied how individual aspects of music affect response patterns. Our results are particularly interesting when compared with recent results regarding the impact of music on emotional processing, as they illustrate that music affects analytical decision making in a fundamentally different way, hinting at a different psychological mechanism that music impacts.

1. INTRODUCTION

There is plentiful evidence that one's mood can affect how one processes information. When the information being processed has emotional content (words, for instance), this phenomenon is referred to as mood-congruent processing, or bias, and it's been found that positive mood induces a relative preference for positive emotional content and vice versa [2, 7]. However, what effect does music have on non-emotional decision making? This study focuses on the impact of music on risky decision behavior which requires quantitative reasoning. To this end, we design an experiment in which participants decide whether to accept or reject gambles with different win-loss ratios (meaning they have different expected payoff).

Previous work in this area shows robust effects of loss aversion, whereby participants put more weight on potential losses than potential gains. Loss aversion in this context manifests as subjects being unwilling to accept gambles unless the potential gain significantly outweighs the potential loss (e.g., only accepting the gamble if the gain is twice as large as the loss [12, 13]). The present study focuses on whether and how emotional music influences such risky decision behavior.

Not much work has studied the direct connection between music and risky decision making. Some previous work has studied the general connection between gambling behavior and ambiance factors including music [1, 3, 11] in an unconstrained casino environment. Additionally, Noseworthy and Finlay have studied the effects of music-induced dissociation and time perception in gambling establishments [6]. In this paper, we take a deeper and more controlled look at how music impacts decision making in this type of risk-based analytical decision making. To this effect, we use a popular model of simple decisions, the drift-diffusion model (DDM; [8]), to explore how music-induced mood affects the different components of the decision process in such tasks. Our results indicate that music indeed has a nontrivial and unexpected effect, and that certain types of music led to better decision making than others.

The structure of the paper is as follows. In Section 2 we outline the characteristics of the drift-diffusion model, which we use in this study. In Section 3 we discuss our experimental design and how data was collected from participants. In Section 4 we present and analyze the results of our behavioral study. In Section 5 we analyze how individual auditory components correlate with the behavioral patterns observed in our human study. In Section 6 we recap our results and discuss them in a broader context.

2. THE DRIFT-DIFFUSION MODEL

This study employs the Drift Diffusion Model (DDM) of simple decisions to decompose the observed decision behavior into its underlying decision components. The DDM, shown in Figure 1, belongs to a family of evidence accumulation models which frame simple decisions

© Elad Liebman, Peter Stone, Corey N. White. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Elad Liebman, Peter Stone, Corey N. White. "Impact of Music on Decision Making in Quantitative Tasks", 17th International Society for Music Information Retrieval Conference, 2016.

in terms of gradual sequential accumulation of noisy evidence until a decision criterion is met. In the model, the decision process starts between the two boundaries that correspond to the response alternatives. Evidence is accumulated over time to drive the process toward one of the boundaries. Once a boundary is reached, it marks the choice of a specific response. The time taken to reach the boundary represents the decision time. The overall response time is the sum of the time it takes to make a decision plus the time required for processes outside the decision process like encoding and motor execution. The model includes a parameter for this nondecision time (Ter), to account for the duration of these processes.

The primary components of the decision process in the DDM are the boundary separation, the starting point, and the drift rate. Boundary separation represents response caution or the speed vs. accuracy tradeoff exhibited by the participant. Wide boundaries indicate a cautious response style where more evidence needs to be accumulated before a decision is reached. The need for more evidence makes the decision process slower, but also more accurate, since it is less likely to reach the wrong boundary by mistake. The starting point of the diffusion process (z) reflects response expectancy. If z is closer to the top boundary, for instance, it means less evidence is required to reach that boundary, so "positive" responses will be faster and more probable than "negative" responses. Finally, the drift rate (v) represents the processing of evidence from stimulus which drives the accumulation process. Positive values indicate evidence for the top boundary, and negative values for the bottom boundary. Furthermore, the absolute value of the drift rate represents the strength of the stimulus evidence, with larger values indicating strong evidence and leading to fast and accurate responses.

In the framework of the DDM, there are two mechanisms that can drive behavioral bias. Changes in the starting point (z) reflect a response expectancy bias, whereby there is an a-priori preference for one response even before the stimulus is shown [5, 14]. Experimentally, response expectancy bias is observed when participants have an expectation that one response is more likely to be correct and/or rewarded than the other. In contrast, changes in the drift rate (v) reflect a stimulus evaluation bias, whereby there is a shift in how the stimulus is evaluated to extract the decision evidence. Experimentally, stimulus evaluation bias is observed when there is a shift in the stimulus strength and/or the criterion value used to classify the stimuli. Thus response expectancy bias, reflected by the starting point in the DDM, indicates a shift in how much evidence is required for one response relative to the other, whereas stimulus evaluation bias, reflected by a shift in the drift rates in the DDM, indicates a shift in what evidence is extracted by the stimulus under consideration. Importantly, both mechanisms can produce behavioral bias (faster and more probable responses for one choice [14]), but they differentially affect the distribution of response times. In brief, response expectancy bias only affects fast responses, whereas stimulus evaluation bias affects both fast and slow responses

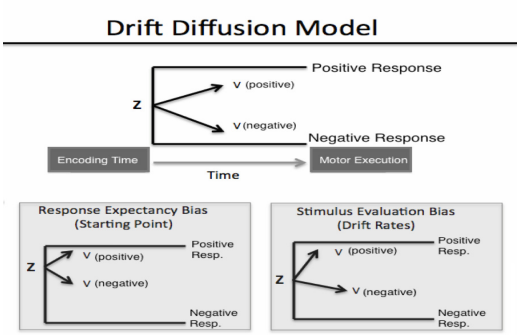


Figure 1. An Illustration of the Drift-Diffusion Model.

(see [14]). It is this differential effect on the response time (abbreviated RT) distributions that allows the DDM to be fitted to behavioral data to estimate which of the two components, starting point or drift rates, is driving the bias observed in the RTs and choice probabilities. The DDM has been shown to successfully differentiate these two bias mechanisms from behavioral data in both perceptual and recognition memory tasks [14].

This study used the DDM approach described above to investigate how music-induced mood affects the different decision components when performing a quantitative task. Participants listened to happy or sad music while deciding whether to bet or fold as bets with different win-loss ratios were proposed to them. The DDM was then fitted to each participant's behavioral data to determine whether the mood induction affected response expectancy bias, stimulus evaluation bias, or both.

3. METHODS

Participants were presented with simple binary gambles and were asked whether to accept (bet) or reject them (fold). Each gamble had a 50%-50% chance of success, with varying win to loss ratio, reflecting how much was to be gained vs. lost. For example, a 15:5 win-loss ratio reflect a 50% chance to win 15 points and a 50% chance of losing 5 points. After a fixation cue was shown for 500 ms, each gamble was presented in the center of the screen and remained on screen until a response was given. If no response was given after 3.5 seconds, the trial ended as a "no response" trial. Responses were indicated with the "z" and "/" keys, and mapping between the key and response was counterbalanced across participants.

The gamble stimuli were partitioned to very negative (win-loss ratio in range $[0.33, 0.66)$), negative (win-loss ratio in range $[0.66, 1)$), positive (win-loss ratio in range $[1, 2)$), and very positive (win-loss ratio in range $[2, 3]$). The actual values of the bets were randomized in the range of $[3, 60]$. Each experiment comprised 20 batches of 20 gambles, such that in each batch each stimuli condition was repeated 5 times (gamble order was randomized). Subjects were not shown the outcome of their gambles immediately as that would be distracting. Instead, between each batch subjects were shown the overall score they ac-

crued for the previous batch (whereas each batch score starts as 0). To encourage competitive behavior, they were also shown the expected score for that batch. A different song was played during each block of 5 batches, alternating from positive to negative music across blocks. The order of the songs was counterbalanced across subjects. The entire experiment lasted less than 30 minutes. To ensure that the results were not specific to the particular choice of songs, the entire experiment was repeated with a large sample of participants ($N = 84$), and two separate sets of songs to assess result reliability.

The music used for this experiment is the same as that used in [4]. It is a collection of 8 publicly available songs which was surveyed to isolate two clear types - music that is characterized by slow tempo, minor keys and somber tones, typical to traditionally "sad" music, and music that has upbeat tempo, major scales and colorful tones, which are traditionally considered to be typical to "happy" music. The principal concern in selecting these musical stimuli, rather than their semantic categorization as either happy or sad, was to curate two separate "pools" of music sequences that were broadly characterized by a similar temperament (described above), and show they produced consistent response patterns. In [4], it has been shown experimentally that the selected music was effective for inducing the appropriate mood. This was done by selecting a separate pool of 40 participants and having them rate each song on a 7-point Likert scale, with 1 indicating negative mood and 7 indicating positive mood. It was then shown that the songs designated as positive received meaningfully and statistically significantly higher scores than those denoted as sad.

The DDM was fitted to each participant's data, separately for positive and negative music blocks, to estimate the values of the decision components. The data entered into the fitting routine were the choice probabilities and RT distributions (summarized by the .1, .3, .5, .7, and .9 quantiles) for each response option and stimulus condition. The parameters of the DDM were adjusted in the fitting routine to minimize the χ^2 value, which is based on the misfit between the model predictions and the observed data (see [9]). For each participant's data set, the model estimated a value of boundary separation, nondecision time, starting point, and a separate drift rate for each stimulus condition. Because of the relatively low number of observations used in the fitting routine, the variability parameters of the full DDM were not estimated (see [8]). This resulted in two sets of DDM parameters for each participant, one for the positive music blocks and one for the negative music blocks.

4. EXPERIMENTAL RESULTS

The response times and choice probabilities shown in Figure 2 indicate that the mood-induction successfully affected the decision making behavior observed across participants. The left column shows the response proportions, the center column shows normalized response times for betting decisions, and the right panel shows normalized response times for the folding decisions. The betting proportions and response time (or RT) measures for the two conditions - the happy songs and the sad songs - indicate a clear difference between the conditions. Generally speaking, happy music led to more "correct" behavior - participants were more likely to accept good bets and reject bad bets under the happy song condition than the sad song condition. These trends are evident across all gamble proportions and bet-fold decisions, but were only shown to be statistically significant for some of the settings; the difference in betting proportions is shown to be significant for very negative, positive and very positive gambles, whereas the difference in response times is only shown to be significant for folding decisions in very positive gambles. Significance was evaluated using a paired t-test with $p \leq 0.05$.

Figure 3 shows the DDM parameters fitted for the experiment. Although the two bias-related measures (starting point and drift rates) are of primary interest, all of the DDM parameters were compared across music conditions. It is possible that the different music conditions could affect response caution and nondecision time. For example, the slower tempo of the sad songs could lead participants to become more cautious and have slower motor execution time. Thus all parameters were investigated. As the top-left and top-center panels of Figure 3 show, the music conditions did not differentially affect response caution or encoding/motor time, as neither boundary separation nor nondecision time differed between happy and sad music blocks. Of primary interest were the starting point and drift rate parameters, which provide indices of response expectancy and stimulus evaluation bias, respectively. Interestingly, as apparent in the top-right and bottom-right panels of Figure 3, overall, we did not observe any stimulus (evidence processing) bias nor starting point (response expectancy) bias in the two music conditions. However, the key difference lied in the drift rates themselves. Fitting parameters for the drift rates for the four gamble types indicate an overall change in evidence processing in the happy vs. the sad music conditions, which is statistically significant for all gamble proportions. This outcome is shown in the bottom-left panel of Figure 3. In other words, people were faster to process the evidence and make betting decisions for good gambles and folding decisions for bad gambles in happy vs. sad music. This difference is summarized in the bottom-center panel of Figure 3, which presents the discriminability factor in the happy vs. the sad condition. Discriminability is defined as the sum of the drift rates for good bets minus the sum of the drift rates for the bad bets, $(d_{positive} + d_{very-positive} - d_{negative} - d_{very-negative})$. This measure represents the "processing gap" between good evidence (good bets) and bad evidence (bad bets). The discriminability was dramatically higher for happy songs compared to sad songs.

The DDM results show that the music-based manipulation of mood affected the overall processing of information in the quantitative task of deciding when to bet and when to fold, rather than any single bias component. There were no effects of music on response caution, nondecision time, or response or stimulus bias, meaning that people weren't

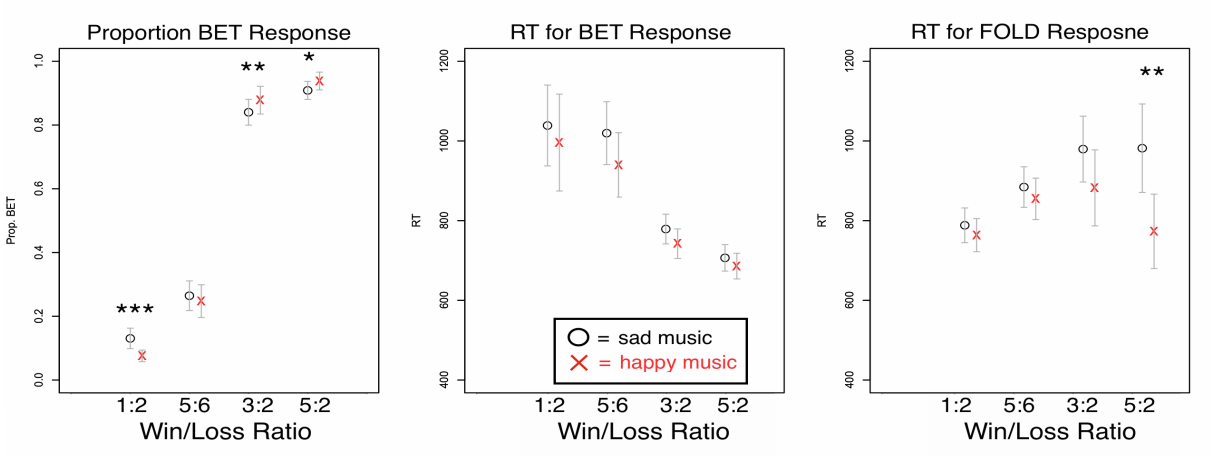


Figure 2. Response patterns in terms of response times and bet-fold proportions for the behavioral experiment. A statistically significant difference between the happy song and the sad song conditions is evident for betting proportions given the four clusters of betting ratios (very negative, negative, positive and very positive). There is also a large statistically significant difference between response times for folding in the two different conditions. Error bars reflect 95% confidence intervals. * $= p < .05$; ** $= p < .01$; *** $= p < .001$.

more likely to accept bets or reject them in one condition or the other, but rather the change impacted the entire decision process. In other words, the mood change induced by music neither affected the a-priori inclination of people to bet or to fold, nor has it led to a relative difference in processing one type of bet vs. the other, but rather simply made people make better decisions (more likely to accept good bets and reject bad ones).

5. CORRELATING RESPONSES AND MUSICAL FEATURES

The partition between "positive" and "negative" mood-inducing songs is easy to understand intuitively, and in itself is enough to induce the different behavioral patterns discussed in the previous section. However, similarly to the analysis performed in [4], we are interested in finding a deeper connection between the behavior observed in the experiment and the different characteristics of music. More exactly, we are interested in finding the correspondence between various musical features, which also happen to determine how likely a song is to be perceived as happy or sad, and the gambling behavior manifested by participants. To this end, we considered the 8 songs used in this experiment, extracted key characterizing features which we assume are relevant to their mood classification, and examined how they correlate with the subject gambling behavior we observed.

5.1 Extracting Raw Auditory Features

We focused on three major auditory features: a) overall tempo; b) overall "major" vs. "minor" harmonic character; c) average amplitude, representing loudness. Features (a) and (c) were computed using the Librosa library [10]. To compute feature (b), we implemented the following procedure, similar to that described in [4]. For each snippet of 20 beats an overall spectrum was computed and individual pitches were extracted. Then, for that snippet, according to the amplitude intensity of each extracted pitch, we identified whether the dominant harmonic was major or minor. The major/minor score was defined to be the proportion of major snippets out of the overall song sequence. Analysis done in [4] confirms these three features were indeed associated with our identification as "positive" vs. "negative". Having labeled "positive" and "negative" as 1 and 0 respectively, a Pearson correlation of $0.7 - 0.8$ with p-values ≤ 0.05 was observed between these features and the label. Significance was further confirmed by applying an unpaired t-test for each feature for positive vs. negative songs (p-values $< .05$).

5.2 Processing Observed Gambling Behavior

Given the complexity of the behavioral experiment discussed in this paper, several behavioral breakdowns of participant behavior were extracted. Normalizing the response times (RTs) for each participant, we separately considered the average response times for betting and for folding for all four gamble types and songs (64 values overall). Subsequently, we aggregated these average response times per decision (bet or fold), per gamble type (very negative, negative, positive and very positive), per song (4 happy songs, 4 sad songs overall), to obtain 64 average response times and response time variance per ⟨decision, gamble type, song⟩ configuration. Then we could correlate these values per ⟨decision, gamble type⟩ setting with the features extracted for each song. Similarly, we extracted the average bet-fold ratio and bet-fold variance across all participants for each ⟨decision, gamble type, song⟩ configuration as well. As a result we were also able to examine the relationship between bet-fold ratios per ⟨decision, gamble type⟩ setting with the features extracted for the songs.

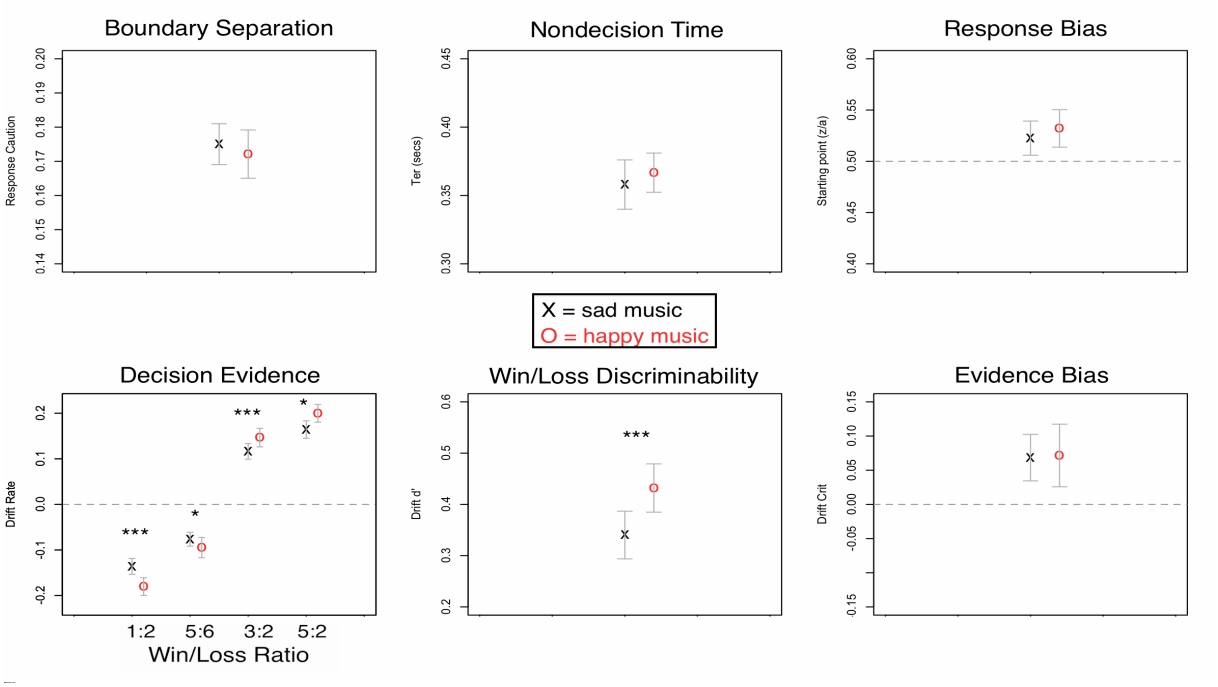


Figure 3. Drift-Diffusion Model parameters fitted for the behavioral experiment. A statistically significant difference between the happy song and the sad song conditions is evident for betting proportions given the four clusters of betting ratios (very negative, negative, positive and very positive). There is also a large statistically significant difference between response times for folding in the two different conditions. Error bars reflect 95% confidence intervals. * $= p < .05$; ** $= p < .01$; *** $= p < .001$.

Decision	Gamble	RT Avg	RT Var.	Avg. p-val	Var. p-val
bet	v. negative	**-0.73**	**-0.61**	**0.03**	**0.1**
bet	negative	**-0.65**	0.48	**0.07**	0.22
bet	positive	**-0.77**	**-0.64**	**0.02**	**0.07**
bet	v. positive	**-0.78**	-0.59	**0.02**	0.11
fold	v. negative	**-0.81**	**-0.65**	**0.01**	**0.07**
fold	negative	**-0.78**	-0.56	**0.02**	0.14
fold	positive	-0.45	-0.45	0.25	0.25
fold	v. positive	**-0.77**	**0.76**	**0.02**	**0.02**

Table 1. Correlation values between tempo and response times (average and variance). Results with p-value ≤ 0.1 are marked in bold.

5.3 Observed Correlations

In this section we discuss how the auditory features corresponded with the normalized response time and bet-fold ratio information extracted from the behavioral experiment. We proceed to analyze the more exact correspondence between the DDM parameters as extracted per song individually and the auditory features of the songs. We note that since we are correlating continuous scalar aggregates across users with continuous auditory features, using the assumptions implicit in a standard Pearson correlation is reasonable.

5.3.1 Correlation with RTs and Bet-Fold Ratio

Examining the relationship between the features extracted per song and the response time and bet-fold ratio data discussed in 5.2 reveals a compound and interesting picture.

Tempo was consistently and in most cases statistically significantly inversely correlated with response times. This was true for all gamble types and decision combinations. Tempo also tended to be inversely proportional to the observed response time variance. Again, this result was consistent across all gamble type and decision combinations. In other words, generally speaking, not only people responded faster (lower response times) the faster the music was, the variance in response times also tended to be reduced. The observed Pearson correlations for average normalized response times and response time variances across the 8 gamble type and decision combinations is provided in Table 1.

Tempo was also inversely correlated with the average bet-fold ratio for very negative gambles ($r = -0.74, p = 0.03$). This also manifested in the correlation with the bet-fold variance ($r = -0.66, p = 0.06$). However, it was linearly correlated with the bet-fold ratio in the very positive gambles case ($r = +0.71, p = 0.04$). Furthermore, in the very positive gambles case, the variance was still reduced, leading to a negative correlation ($r = -0.71, p = 0.04$). In other words, the faster the music, the more people are likely to bet on very good bets, and more consistently (reducing variance). Furthermore, the faster the music, the more likely people are to fold on bad bets, and more consistently (reducing variance). This is a strong signal for how tempo improves the quality of decision making in quantitative tasks.

There is evidence that the major dominance feature (de-

termining the major to minor chord proportion in each song) is inversely correlated to the average bet-fold ratio and the bet-fold ratio variance in the very negative gambles case (average: $r = -0.6, p = 0.11$, variance: $r = -0.61, p = 0.10$). Similarly, there is some evidence that major dominance is linearly correlated with the average bet-fold ratio and inversely correlated to the bet-fold variance in the strong-positive case, but this result wasn't as convincing (average: $r = +0.42, p = 0.29$, variance: $r = -0.58, p = 0.14$). This result, though inconclusive, hints at the possibility that the more major chords there are in a song, the better the analytical decision making that subjects manifest.

Interestingly, the major dominance feature (determining the major to minor chord proportion in each song) was inversely proportional to the variance in response times when folding on a very positive bet ($r = -0.71, p = 0.04$). Major dominance was also inversely proportional to variance in response times betting on a very negative bet ($r = -0.65, p = 0.07$). In other words, the more major chords appeared in a song, the less variability people displayed in the time it took them to make a poor decision. This could be a side effect of people making fewer such mistakes in these gamble - decision combinations, as was documented in previous sections.

The average amplitude was inversely correlated to the average bet-fold ratio and the bet-fold ratio variance for negative and very negative gambles. These observations seem borderline significant (average: $r = -0.59, p = 0.12$ for negative, $r = -0.53, p = 0.17$ for very negative, variance: $r = -0.58, p = 0.13$ for negative, $r = -0.7, p = 0.05$ for very negative). This would imply that the louder the music, the less likely people are to make betting decisions on bad gambles, and variance is also reduced.

5.3.2 Correlation with DDM Decomposition

Finally, we were also interested in examining how the individual DDM parameters fitted for each song separately corresponded with the song features. Comparing the DDM parameters per song with the tempo, major dominance and amplitude data, we observed a statistically significant correlation between the tempo and the drift rate for very positive gambles (Figure 4(a), $r = -0.72, p = 0.04$), tempo and very negative gambles (Figure 4(b), $r = +0.79, p = 0.01$), and, interestingly, between the mean amplitude and the response caution, a connection that was also suggested in [4] (Figure 4(c), $r = -0.67, p = 0.06$). These observations corroborate both the observations in Section 5.3.1, and in Section 4.

6. SUMMARY & DISCUSSION

In this paper, we study how music-induced mood affects decision making in risky quantitative tasks. Subjects were presented with gambles and needed to decide whether to accept or reject these gambles as different types of music were played to them. Our results show that while there is no evidence for music-induced bias in the decision making process, music does have a differential effect on decision

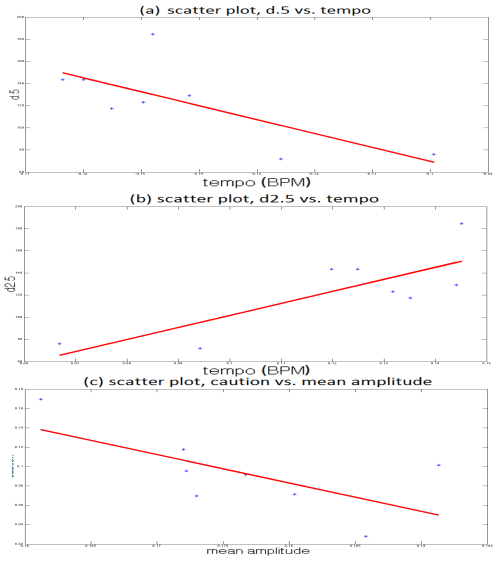


Figure 4. (a) Correlation between tempo and the drift rate for very negative gambles. (b) Correlation between tempo and the drift rate for very positive gambles. (c) Correlation between mean amplitude and the overall response caution (boundary separation).

making behavior. Participants who listened to music categorized as happy were faster to make decisions than people who listened to music categorized as sad. Moreover, the decisions participants made while listening to happy music were consistently better than those made while listening to sad music, implying increased discriminabilty. Further analysis indicates there is a correlation between tempo and the speed and quality of decision making in this setting. Interestingly, previous work on gambling behavior has found a connection between the tempo and the speed of decision making, but was unable to isolate further impact on the quality of decision making, due to a fundamentally different design and different research questions [1].

Of particular note is the comparison between the results of a recent paper studying the connection between music-induced mood and mood-congruent bias [4]. In that paper, participants were requested to classify words as happy or sad as music categorized as happy or sad was played. Results indicated a clear expectancy bias, meaning music affected people's a-priori tendency to classify words as happy or sad. This paper, which uses the exact same set of songs, has reported no such bias effect, or any bias effect, for that matter. This difference suggests the psychological mechanisms involved in emotional classification and risky analytical decision making are inherently different.

This paper is a meaningful step towards a better understanding of the impact music has on commonplace cognitive processes which involve quantitative reasoning and decision making. In future work, additional tasks and other music stimuli should be studied to better understand the relationship between music and this type of cognitive processing.

7. REFERENCES

[1] Laura Dixon, Richard Trigg, and Mark Griffiths. An empirical investigation of music and gambling behaviour. *International Gambling Studies*, 7(3):315–326, 2007.

[2] Rebecca Elliott, Judy S Rubinsztein, Barbara J Sahakian, and Raymond J Dolan. The neural basis of mood-congruent processing biases in depression. *Archives of general psychiatry*, 59(7):597–604, 2002.

[3] Mark Griffiths and Jonathan Parke. The psychology of music in gambling environments: An observational research note. *Journal of Gambling Issues*, 2005.

[4] Elad Liebman, Peter Stone, and Corey N. White. How music alters decision making - impact of music stimuli on emotional classification. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, pages 793–799, 2015.

[5] Martijn J Mulder, Eric-Jan Wagenmakers, Roger Ratcliff, Wouter Boekel, and Birte U Forstmann. Bias in the brain: a diffusion model analysis of prior probability and potential payoff. *The Journal of Neuroscience*, 32(7):2335–2343, 2012.

[6] Theodore J Noseworthy and Karen Finlay. A comparison of ambient casino sound and music: Effects on dissociation and on perceptions of elapsed time while playing slot machines. *Journal of Gambling Studies*, 25(3):331–342, 2009.

[7] Kristi M Olafson and F Richard Ferraro. Effects of emotional state on lexical decision performance. *Brain and Cognition*, 45(1):15–20, 2001.

[8] Roger Ratcliff and Gail McKoon. The diffusion decision model: theory and data for two-choice decision tasks. *Neural computation*, 20(4):873–922, 2008.

[9] Roger Ratcliff and Francis Tuerlinckx. Estimating parameters of the diffusion model: Approaches to dealing with contaminant reaction times and parameter variability. *Psychonomic bulletin & review*, 9(3):438–481, 2002.

[10] Brian McFee ; Matt McVicar ; Colin Raffel ; Dawen Liang ; Douglas Repetto. Librosa. *https://github.com/bmcfee/librosa*, 2014.

[11] Jenny Spenwyn, Doug JK Barrett, and Mark D Griffiths. The role of light and music in gambling behaviour: An empirical pilot study. *International Journal of Mental Health and Addiction*, 8(1):107–118, 2010.

[12] Sabrina M Tom, Craig R Fox, Christopher Trepel, and Russell A Poldrack. The neural basis of loss aversion in decision-making under risk. *Science*, 315(5811):515–518, 2007.

[13] Amos Tversky and Daniel Kahneman. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and uncertainty*, 5(4):297–323, 1992.

[14] Corey N White and Russell A Poldrack. Decomposing bias in different types of simple decisions. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 40(2):385, 2014.

INTERACTIVE SCORES IN CLASSICAL MUSIC PRODUCTION

Simon Waloschek, Axel Berndt, Benjamin W. Bohl, Aristotelis Hadjakos
Center of Music And Film Informatics (CeMFI)
University of Music Detmold, Germany
{waloschek,berndt,hadjakos}@hfm-detmold.de, bohl@edirom.de

ABSTRACT

The recording of classical music is mostly centered around the score of a composition. During editing of these recordings, however, further technical visualizations are used. Introducing digital interactive scores to the recording and editing process can enhance the workflow significantly and speed up the production process. This paper gives a short introduction to the recording process and outlines possibilities that arise with interactive scores. Current related music information retrieval research is discussed, showing a potential path to score-based editing.

1. INTRODUCTION

Classical music generally revolves around the musical score. It is used as fundamental interpretation directions by musicians and conductors. During recording of classical music, scores are used as a means of communication as well as direct working material of record producers. Successive working steps towards a finished music production however utilize additional views upon the recorded audio material while still frequently referring to the score. This media disruption can take a great deal of time since these different views are not synchronized in any way. Although most technologies that are needed to overcome this disadvantage are already present, they have not been used in this specific field of application. This paper therefore summarizes the ongoing efforts of introducing interactive scores to the classical music production process and discusses open issues.

2. CLASSICAL MUSIC PRODUCTION

In order to understand the score-related needs and issues arising during classical music production, a brief overview of the production process will be given. It is neither complete in terms of performed steps taken nor does it claim to comprehensively address every aspect of the production process. Tasks that have implications for further considerations will be outlined in more detail.

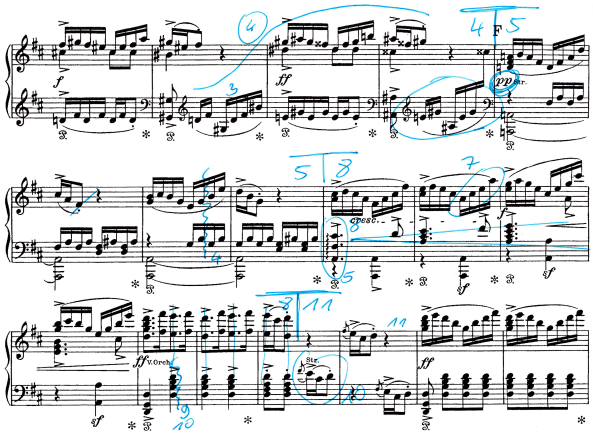


Figure 1. Annotations made in the score during a recording session.

The production of classical music recordings is usually divided into three major phases: *pre-production*, *production* and *post-production* [5]. During *pre-production* an essential goal is to set up a production plan in concordance with the artistic director or conductor. From the record producer's perspective this of course includes analyzing the piece(s) of music to be recorded. This includes several aspects, such as identifying challenging passages. Generally speaking, the record producer's main goal is to familiarize himself with the piece in a way that will allow him to perform his later tasks in a best possible manner, e.g. by listening to existing recordings and studying the score. During this process, the record producer might annotate and mark passages in the score for later consideration. As the score will later be a major means of communication with the conductor or musicians, it should be identical to the conducting score with respect to its appearance, e.g. page-layout or reference points.

Capturing the raw audio material from the musicians' performance in the *Digital Audio Workstation* (DAW) is the main goal of the the *production phase*. This might be done in several recording sessions, depending on the scope and nature of the music. Moreover it is common practice to repeat musically or technically unsatisfying passages multiple times but yet keep all of the recorded *takes*.

© Simon Waloschek, Axel Berndt, Benjamin W. Bohl, Aristotelis Hadjakos. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Simon Waloschek, Axel Berndt, Benjamin W. Bohl, Aristotelis Hadjakos. "Interactive Scores in Classical Music Production", 17th International Society for Music Information Retrieval Conference, 2016.

The responsible record producer has to carefully listen to the played music as well as pay attention to the musical score at the same time. Deviations from the score and other score-related comments—positive and negative—are mostly annotated directly in the score as shown in Figure 1. It is to be noted that there is no standardized set of symbols that is commonly used by record producers, e.g. beginnings or endings of the takes or annotating the quality of a certain passage during a take. Every producer develops his own set of symbols based on his personal experience and specific needs.

Oftentimes, an additional *take list* is manually maintained that reflects the beginning and ending measures of the individual takes in relation to their take number, see Table 1.

On the basis of their observations during individual takes, the record producer's tasks include keeping an overview of whether all passages have been recorded in a satisfying manner. If they have not, they will communicate this to the musicians and ask for additional takes. Communication with the musicians is done orally using dedicated audio lines aka talkback channels. For this purpose the score and its consistency with the conductor's score and the musicians' parts is an essential basis for communication. Page numbers, measure numbers and other reference marks of the score will be used to communicate the precise location to be considered.

The following editing process is dominated by selecting and splicing parts from the takes that best conform to the musical aesthetics and technical demands of the production. This often requires reviewing huge amounts of audio data with identical sections spread across the various takes. With the takes usually being rendered consecutively in the editing environment as *waveforms* (see Figure 2), navigation for auditory comparisons becomes a time-consuming task, see Section 3.4.

A well-organized take list helps to decrease the time needed to identify the takes containing a specific section of the recorded composition. Nevertheless, deciding which takes to splice might be a process of consecutive comparisons of several takes that often cannot be broken down to mere technical aspects (in the sense of the musicians' playing technique, as well as the recording quality) but has to account for aesthetic aspects too, e.g. the quality of a passage's musical performance. When a decision has been made about which takes to splice, it comes to rendering the splice imperceptible. Besides some technical aspects like selecting an adequate zero crossing in the waveforms, adjusting the loudness of the takes, and optimizing the crossfade between the two takes, it is the editor's ears that will allow them to asses the quality of the splice. All in all, this often means specifying the precise edit location by ear and on the basis of a sound music-aesthetic sensitivity of the editor.

Take	Pos. / Measures	Comment
1	α - Ω	
2	α - 17	Beautiful start
3	13 - 31	
4	22 - 31	Quarters not in time
5	29 - 52	
⋮	⋮	

Table 1. Exemplary list of takes from a recording session.

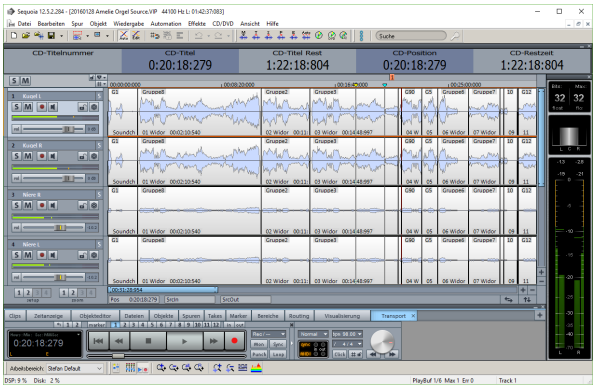


Figure 2. DAW with multiple takes lined up in a row.

The mixing phase ensures that the sound of the recording has the right balance in terms of each instrument's volume levels. Dynamics and panoramic position are manipulated and filters and effects such as reverb may be added to produce a mix that is more appealing to the listener. These tasks as well as the final mastering are not further considered, as the annotated score and take list do not play major roles in them.

3. INTERACTIVE SCORES AS MUSIC PRODUCTION INTERFACES

Replacing conventional paper scores with their interactive counterparts opens up a variety of workflow optimizations for music production and establishes fruitful connections to further research and application areas such as digital music edition. The following sections describe the parts of the workflow that could benefit from a more ubiquitous use of digital scores. The corresponding research topics are mainly located in the field of human-computer interaction (pen and touch interaction, gesture development and recognition, user interface design), audio processing and music information retrieval (audio-to-audio and audio-to-score alignment).

3.1 Sheet Music Interaction and Annotations

Sheet music marks a central work object in the pre-production phase, recording session (here in particular) and in the editing phase as the preceding introduction to classical music production shows. Handwritten annotations in the score are an effective, easily accessible and versatile means to document the recording process and communicate with the musicians. However, the number of remarks increases drastically during a recording session and tends to hamper readability and complicate the assignment of annotations to specific takes.

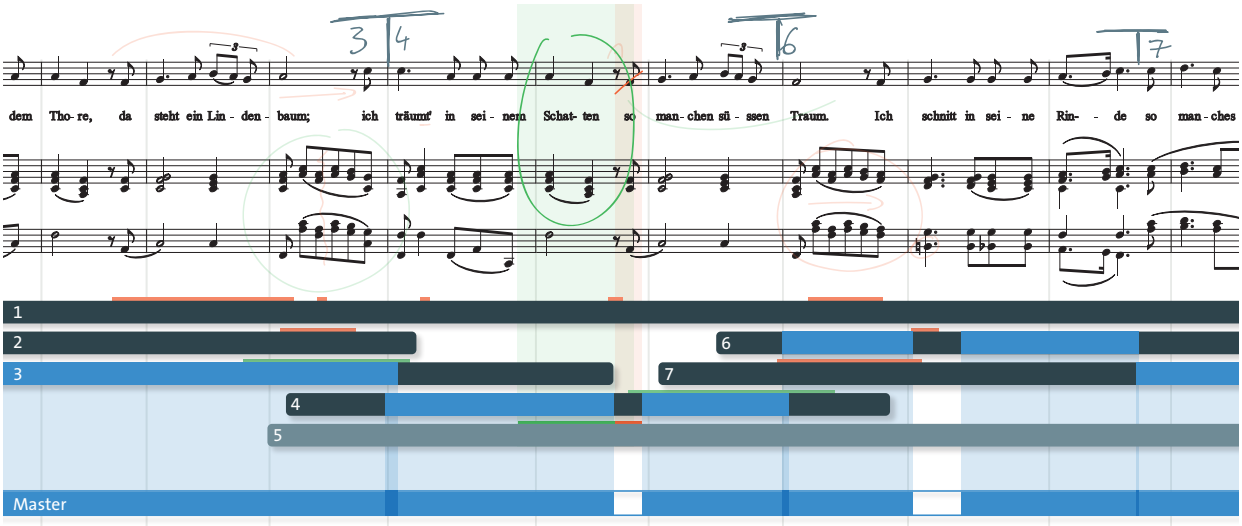


Figure 3. Mock-up of a Pen and Touch-based User Interface for Take Selection and Editing

A transformation of the analog writing on the music sheet into the digital domain can be achieved via digital pen and paper technology such as Anoto. An overview of respective technologies can be found in [20]. The advantage of digital paper lies in its possibility to link the annotations with the recorded audio data and process them accordingly. Limitations become apparent when musical sections have to be recorded repeatedly, each take introducing new annotations until the paper sheet is overfull and hardly readable. Furthermore, the step from the recording to the editing phase introduces a media disruption. In the latter, the focus lies on navigating, selecting and editing takes which cannot be done with the sheet music. Here, a continuous staff layout, aligned with the audio data, is desirable.

Even though printed sheet music has some practical limitations, it features the aforementioned clear advantages. These motivate a fundamental design decision that will underlay our further concept development: *The interactive score should become the central interface widget during the recording and editing phase.* Up to now, the DAW (see Figure 2) marks the central productive interface and the printed score serves as a secondary medium to hold further information. To preserve the advantages of (digital) pen and paper we regard pen and touch displays as the most promising technology. The score layout can easily be switched from a traditional page-wise style during the recording sessions to a continuous staff layout in the editing phase. Annotations can likewise be shifted as they are linked to score positions (e.g., measures and single notes). Figure 3 demonstrates the continuous score layout, aligned with the recorded audio material and supplemented by handwritten annotations.

We conceive pen and touch interaction with interactive scores for music production according to the following scheme. Touch input is mainly used for navigation (turning score pages, panning and zooming) and to toggle buttons and input mode switches. Productive input—primarily the creation of annotations and the editing of takes—is done with the pen as these require a higher precision. At this, we follow the same precedent as Yee [19], Hinckley et al [10] and Frisch [7].

Annotations are layered for each take, i.e., each recording starts with a non-annotated score. However, previous annotations can be switched on, if required. This overcomes the problem of overfull music sheets during the recording session. Hence, the record producer can make annotations at their exact place in the score without having to deal with previous annotations. Annotations can be structured, moved, hidden, or deleted. This allows, for example, to show only those annotations that have been written throughout the last three takes, helping to keep an overview. Mostly, annotations are also rated as positive, neutral or negative which helps the record producer to select the best takes in the editing phase. Such ratings may be indicated by symbols such as "+", "−" and "∼". All annotations have to be made very quickly during the music performance and each additional mark costs time. Instead of such additional symbols, the side-switch of a digital pen and its eraser can be used as mode switches and the annotations may be color coded accordingly. These may come in handy during the editing phase to quickly find the right takes (see Section 3.4).

Moreover, annotations can even serve as control gestures. Record producers typically note the start and stop position of a take in the score by "⌐" and "¬" and a serial number. Instead of controlling the recording functionality at a different place, it can be triggered immediately when the input symbol is recognized as a control gesture. The take's serial number and naming can be generated. The symbols and their positions in the score further help to align the recorded audio material with the score.

3.2 Protocol Automation

Centering the recording and editing workflow around digital scores is advantageous also during the pre-production

Figure 4. A pen and touch display used for score annotation.

phase. Digital scores can be generated from commonly used notation formats such as MusicXML and MEI (Section 4.1 addresses music formats with regard to technological requirements). In the latter case, i.e. MEI, elements of the editors' critical report can be included and help to clarify the musical idea.

The information that are provided by MusicXML, MEI and other symbolic music representation formats, that may underlay the digital score, often include information about the instrumentation, the number of voices and voice groupings. From these information a basic recording project can be automatically initialized, i.e. the creation of audio tracks and their naming. Even a rough panning setup can be generated from the typical orchestra setting and from the miking plan that is made during the preparatory meeting.

During the recording sessions, the record producer might maintain a take list, see Table 1. This is done in addition to the interactions in the DAW and the annotations in the score. The information in the take list is actually redundant with the score annotations and can be generated from them which relieves the record producer. Take list comments can be generated from the qualitative connotations of the score annotations (positive, negative, neutral) and from textual annotations and symbols that may be recognized by the system.

3.3 Communication

Interactive scores can help communicating with the musicians and serve as communication channel. In the recording session the music notation is displayed in a traditional page-wise score layout that matches with the layout that the musicians have. This eases the communication. Referring to a position in the music is mostly done in the format (page, staff line, measure). If the musicians use digital music stands, the digital score can even become a means for visual communication. Here, the record producer may make annotations in their score that are synchronously displayed on the musicians' music stands. This audiovisual mode can help making the communication more effective, less ambiguous and faster.

3.4 Take Selection & Editing

The editing phase and the recording phase utilize different facets of the interface. During the recording phase, the record producer needs to be able to orient himself quickly in the digital score that must be consistent with the musicians' score [1] to facilitate communication. In the editing phase the score and all its additional information (annotations, take list etc.) should facilitate a quick selection of suitable takes which are then spliced via cross-fades.

Since the music editing process changed from analog to digital, the average splice count and frequency increased drastically [18]. We conducted a preliminary survey with 15 record producers to determine the approximate durations of specific tasks in the editing phase. While recording a 10 minute piece takes approx. 2:26h, the pure navigation and splicing process takes 1.62 times as much. The actual selection of the takes in terms of aesthetics was not considered. Navigation between suitable takes marks the most time-consuming part of the editing phase, 54.9% of the time.

Cues that help to identify promising takes are spread over the score and the take list. Takes are arranged consecutively and not aligned in accordance with their actual musical content. However, it is possible to tackle these flaws and approach a solution similar to Figure 3, i.e., a continuous score, each take aligned with it and color coded for qualitative indication.

From the control gestures ("⌐" and "¬", see Section 3.1) we know each take's corresponding score position. This helps aligning them with each other and with the score. Annotations made in the recording phase are linked to positions and even regions in the score. They can also be linked to the actual audio recordings via an audio-to-score alignment. Thereby, problematic passages can be indicated directly in the takes since annotations have a qualitative connotation. Selecting a take reveals its annotations in the score. Based on these qualitative connotations, takes can be recommended and automatically combined into a "raw edit version". This does not make the detailed work of the editor obsolete but accelerates the time-consuming search for good candidates.

4. TECHNOLOGICAL STATE OF THE ART

Many of the previously outlined aspects and issues have already been addressed by current research. This section provides an overview of the relevant developments and links together fundamental techniques that can be used to implement the aforementioned features.

4.1 Digital Score Format

The most basic requirement for score interactivity is the availability of scores in a digital format. Unfortunately, most publishers do still publish their scores solely in a printed form. In order to produce digital counterparts, two different ways can be employed: *Optical Music Recognition (OMR)* [15], which aims at digitizing printed scores

[1] at least with the conductor's score.

via image recognition algorithms, and direct encoding of notes in an appropriate data format. While OMR techniques have been used for linking scores with audio [6,11], they present the same layout related issues discussed in Section 3.1. Therefore, further considerations concentrate mainly on symbolic music encoding.

In order to adequately represent symbolic music data, the *Music Encoding Initiative* (MEI) [16] was founded and elaborated a data format (also called MEI), that is able to hold both musical and layout data of a score.

Although encoding music directly in such formats is a rather time-consuming task, it offers the best flexibility in terms of post processing capabilities and manual layout restructuring. The latter oftentimes provides a tremendous advantage in readability over automatically generated score layouts.

Ongoing digitization efforts of major music publishers will most likely help to overcome the lack of digital scores in the near future.

4.2 Digital Score Rendering

Various approaches have been used in rendering digital scores paired with interactivity, mainly for musical practice. MOODS [2] and muse [8] are two early examples.

MOODS is a collaborative music editor and viewer, which lets musicians, the conductor and the archivist view and annotate the score based on their user priviliges. A similar approach would be useful to support the communication between the record producer and the musicians during music production (see Section 3.3). The muse lets the user view and annotate the music score. It turns pages automatically using audio to score alignment [8]. Page-turning in the MOODS system is based on a horizontal separator that splits the page into two parts: the part below the separator is currently being played, the part above is a preview of the next page, which is good for musicians who oftentimes read ahead.

An alternative representation is an infinite scroll of music score without page breaks as shown in Figure 3. In the score editing software *Sibelius* this approach is called "Panorama" and its "Magic Margins" provide information about clef, key and measure number on the left side. [2]

With the advent of *Verovio* [14], a more modern approach to score rendering is available. Providing an easy interface to render MusicXML as well as MEI files with custom layout properties, it is gaining usage amongst web-based score applications.

4.3 Linking Scores and Audio

In order to visualize the recorded audio takes time-synchronously to the score (see Figure 3), both representations have to be aligned algorithmically; Each position in the individual takes should be linked to the equivalent position in the score and vice versa.

Symbolic music representations offer the possibility to be transformed into audio files, reducing the *score-to-audio-alignment* task to the (commonly considered) simpler problem of *audio-to-audio-alignment*. This way, the generated audio can be annotated automatically with cue points from the score. Tools such as *music21* [9] and *meico* [3] are able to transform MEI, MusicXML etc. into MIDI, which in return can be used to output audio data.

In the next step, these audio data are to be aligned with the recorded takes. Various approaches have been applied and are thoroughly discussed by Thomas et al. [17] and Müller [13]. Annotations in the score during the recording phase as described in Section 3.1 can be used to bypass false alignments in situations, where identical musical passages occur multiple times throughout a composition.

Scenarios with partial usage of alignment techniques in recording situations can be found in [4, 12], though practical implementations are still lacking. This renders comprehensive evaluations of new score-based editing methods impossible.

4.4 Interaction

Changing from printed to digital scores allows for a wide range of interaction possibilities as presented in Section 3.4. Rendering engines like *Verovio* output the score as *Scalable Vector Graphics* (SVGs), which can be viewed in every modern web browser. SVG however developed from a pure visualization format into a major interactivity framework. The content can be changed programmatically with low expenditure of time and allows for pixel-precise reaction to pen and touch input. Using a web browser as front-end enables the usage of JavaScript as the underlying programming language.

Ready-to-use gesture frameworks, e.g. the $N Multistroke Recognizer [1], can easily be incorporated and adapted to make use of established Pen & Touch interaction modalities [3] and the aforementioned gestures to start and stop the recording etc. due to its JavaScript nature.

Hardware-wise, large pen-enabled touch screen displays as shown in Figure 4 are already widely used by media designers and can be adapted to the discussed scenario without further modifications.

5. CONCLUSIONS

Although many technical aspects of interactive scores in recording scenarios have already been addressed by research, the main issue remains in bringing their results together in a usable manner. This usability is generally driven by in-depth insights into the workflow of record producers. Classical music recording is in its roots a rather conservative task that remains skeptical about new developments and, therefore, requires comprehensive analysis in advance. Unfortunately, this field of work seems to be not very accessible in terms of open exchange of working strategies.

[2] http://www.sibelius.com/products/sibeliusedu/5/panorama.html (last accessed March 2016)

[3] http://www.zemfi.de/resources/meico-mei-converter/ (last accessed March 2016)

Once the (classical) recording process is well documented, new interface and usability concepts as exemplary outlined in Section 3 can be developed and evaluated. To have an impact on the actual work processes, the conception and development should be as close as possible to potential users.

The software implementation of such an interface can gain advantage of present MIR and usability engineering research and brings together several topics in a new way. Thus, it provides an interesting and unique use case for future research on music information retrieval methods combined with user interaction analysis.

6. REFERENCES

[1] L. Anthony and J. O. Wobbrock. $N-Protractor: A Fast and Accurate Multistroke Recognizer. In *Proc. of Graphics Interface*, pages 117–120. Canadian Information Processing Soc., 2012.

[2] Pierfrancesco Bellini, Paolo Nesi, and Marius B Spinu. Cooperative visual manipulation of music notation. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 9(3):194–237, 2002.

[3] P. Brandl, C. Forlines, D. Wigdor, M. Haller, and Ch. Shen. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proc. of the Working Conf. on Advanced Visual Interfaces*, pages 154–161. ACM, 2008.

[4] R. B. Dannenberg and N. Hu. Polyphonic Audio Matching for Score Following and Intelligent Audio Editors. In *Proc. of the Int. Computer Music Conf.*, San Francisco, 2003.

[5] J. Eargle. *Handbook of Recording Engineering*. Springer US, 2006.

[6] Chr. Fremerey, M. Müller, F. Kurth, and M. Clausen. Automatic Mapping of Scanned Sheet Music to Audio Recordings. In *Proc. of the Int. Soc. for Music Information Retrieval Conf.*, pages 413–418, 2008.

[7] M. Frisch. *Interaction and Visualization Techniques for Node-Link Diagram Editing and Exploration*. Verlag Dr. Hut, Munich, Germany, 2012.

[8] Christopher Graefe, Derek Wahila, Justin Maguire, and Orya Dasna. Designing the muse: A digital music stand for the symphony musician. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 436–ff. ACM, 1996.

[9] A. Hankinson, P. Roland, and I. Fujinaga. The Music Encoding Initiative as a Document-Encoding Framework. In *Proc. of the Int. Soc. for Music Information Retrieval Conf.*, pages 293–298, 2011.

[10] K. Hinckley, K. Yatani, M. Pahud, N. Coddington, J. Rodenhouse, A. Wilson, H. Benko, and B. Buxton. Pen + Touch = New Tools. In *Proc. of the 23nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 27–36, New York, NY, USA, 2010. ACM.

[11] Ö. İzmirli and G. Sharma. Bridging printed music and audio through alignment using a mid-level score representation. In *Proc. of the Int. Soc. for Music Information Retrieval Conf.*, pages 61–66, 2012.

[12] N. Montecchio and A. Cont. Accelerating the mixing phase in studio recording productions by automatic audio alignment. In *Proc. of the Int. Soc. for Music Information Retrieval Conf.*, 2011.

[13] M. Müller. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer International Publishing, 2015.

[14] L. Pugin, R. Zitellini, and P. Roland. Verovio: A Library For Engraving MEI Music Notation Into SVG. In *Proc. of the Int. Soc. for Music Information Retrieval Conf.*, Taipei, Taiwan, 2014.

[15] Ana Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. S. Marcal, C. Guedes, and J. S. Cardoso. Optical Music Recognition: State-of-the-Art and Open Issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, 2012.

[16] Perry Roland. The music encoding initiative (mei). In *Proceedings of the First International Conference on Musical Applications Using XML*, pages 55–59, 2002.

[17] V. Thomas, Chr. Fremerey, M. Müller, and M. Clausen. Linking Sheet Music and Audio – Challenges and New Approaches. *Dagstuhl Follow-Ups*, 3, 2012.

[18] S. Weinzierl and C. Franke. 'Lotte, ein Schwindel!' – History and practice of editing recordings of Beethoven's symphony No. 9. In *22. Tonmeistertagung – VDT International Convention*, 2003.

[19] K. Yee. Two-handed Interaction on a Tablet Display. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '04, pages 1493–1496, Vienna, Austria, 2004. ACM.

[20] R. B. Yeh. *Designing Interactions That Combine Pen, Paper, and Computer*. PhD thesis, Stanford, CA, USA, 2008.

JAZZ ENSEMBLE EXPRESSIVE PERFORMANCE MODELING

Bantula, Helena **Giraldo, Sergio** **Ramirez, Rafael**
Music Technology Group, Universitat Pompeu Fabra, Barcelona (Spain)
helenabantula@gmail.com, {sergio.giraldo, rafael.ramirez}@upf.edu

ABSTRACT

Computational expressive music performance studies the analysis and characterisation of the deviations that a musician introduces when performing a musical piece. It has been studied in a classical context where timing and dynamic deviations are modeled using machine learning techniques. In jazz music, work has been done previously on the study of ornament prediction in guitar performance, as well as in saxophone expressive modeling. However, little work has been done on expressive ensemble performance. In this work, we analysed the musical expressivity of jazz guitar and piano from two different perspectives: solo and ensemble performance. The aim of this paper is to study the influence of piano accompaniment into the performance of a guitar melody and vice versa. Based on a set of recordings made by professional musicians, we extracted descriptors from the score, we transcribed the guitar and the piano performances and calculated performance actions for both instruments. We applied machine learning techniques to train models for each performance action, taking into account both solo and ensemble descriptors. Finally, we compared the accuracy of the induced models. The accuracy of most models increased when ensemble information was considered, which can be explained by the interaction between musicians.

1. INTRODUCTION

Music is a very important part in the life of milions of people, whether they are musicians, they enjoy attending live music concerts or simply like listening to musical recordings at home. The engaging part of music is the human component added to the performance: instead of a "dead" score, musicians shape the music by changing parameters such as intensity, velocity, volume and articulation. The study of music expressive performance from a computational point of view consists of characterising the deviations that a musician introduces in a score, often in order to render human-like performances from inexpressive music scores.

There are numerous works which study expressive performance in classical music, and most of these studies have been done on piano performances (for an overview, see Goebl [24]). Other works analyse expressivity in a jazz context. For instance, Giraldo and Ramírez [8] study and model the ornamentation introduced to a jazz melody by using machine learning techniques. In an ensemble context, the musicians' performance is influenced by what is being played by the other musicians. Although most works are focused on soloist performances, some works take into account ensemble performances in classical music ([26], [12], [15]). However, to our knowledge little work addresses ensemble expressive performance in a jazz context. In this work, we present a method to study the interaction between jazz musicians from a computational perspective. Our data set consisted of 7 jazz pieces recorded by a jazz quartet (guitar, piano, bass and drums), in which each instrument was recorded on a separate track. In this study we considered the interaction between guitar melodies and the accompaniment of piano. We extracted individual (soloist) score descriptors as well as ensemble descriptors. We calculated performance actions for both guitar (embellishments) and piano (chord density, range and weight). We applied machine learning techniques to predict these performance actions using Artificial Neural Networks, Support Vector Machine and Decision Trees. We generated *individual models* for each instrument and measured the level of interaction between musicians by introducing ensemble descriptors into each individual model to create *mixed models*, and compared the *individual models* with the *mixed models*. Finally, we evaluated the performance of the algorithms by computing statistical significance tests (Paired T-Test).

The rest of the paper is organised as follows. In Section 2, we present related work in expressive music performance. In section 3, we describe the materials we have used. In Section 4, the proposed method is described and the evaluation process is explained. In Section 5, the results of the evaluation are presented. Finally, in Section 6 we put forward conclusions and future improvements.

2. RELATED WORK

Many works study expressive performance actions in music, defined as variations in timing (duration and onsets), energy, articulation, and vibrato from different perspectives, including psychology ([6], [19]), neurology ([13]), musicology ([22]) and at a computational level. Previous work has been done in a **classical context** by Friberg [5], who develops a set of rules using analysis by synthesis to

© . Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** . "Jazz Ensemble Expressive Performance Modeling", 17th International Society for Music Information Retrieval Conference, 2016.

generate the deviations to be applied to a score, obtaining human-like performances. Widmer [25] analyses recordings of 13 complete Mozart piano sonatas and uses a machine learning approach to create predictive rules for note-level timing, dynamics and articulation. In a **jazz context**, previous work has focused on the saxophone: Lopez de Mántaras et al. [1] use case-based reasoning to develop a system capable of modeling expressive performances (onset, duration and energy) of jazz saxophone. Ramírez and Hazan [20] apply classification and regression methods to a set of acoustic features extracted from saxophone recordings and a set of descriptors which characterised the context of the performed notes. In jazz guitar, Giraldo and Ramírez ([9], [10]) use machine learning techniques to model and synthesise embellishments by training models to classify score notes as embellished or not, according to the characteristics of the notes' context.

2.1 Ensemble Performance

In ensemble performance, the expressivity of a soloist might be influenced by what the other musicians are playing. Most of the literature refers to **classical context**, studying timing asynchrony among performers. Repp [21] studies the synchronisation of the task of tapping by taking into account phase and frequency correction mechanisms. Wing et al. [26] develop a model for studying synchronisation in string quartets in different contexts (democratic or dictatorial). Goebl and Palmer [12] investigate the effect of the auditory and visual feedback so to study the synchronisation among musicians. More recently, Marchini [15] studies the interaction between musicians by generating independent machine learning models of expressive performance for each musician and taking into account the influence of the other musicians.

3. MATERIALS

We recorded 7 jazz standards performed by a jazz quartet both in wav and MIDI format, using the digital audio workstation Logic Pro X [14]. The scores were written in Music-XML format using Muse Score [18] to extract descriptors. We developed code by using the computing environment Matlab [17], concretely, the MidiToolBox Library developed by Toiviainen and Eerola [4] to process the data in MIDI format. We used the fundamental frequency estimator YIN [3] to create an automatic guitar melody transcriber. We performed beat tracking of the recordings using the beat tracker developed by Zapata [27]. Finally, we used the Weka Data Mining Software [16] for machine learning modeling.

4. METHODOLOGY

The methodology is divided into three stages, which are depicted in Figure 1. Firstly, we acquired the data from recordings and its respective scores (Section 4.1). Secondly, the data was analysed to extract the chords played by the pianist, which were aligned with the score afterwards so as to obtain piano performance actions. For guitar, we transcribed the audio into MIDI, and aligned the played notes with the score to obtain guitar performance actions. From the score, descriptors for notes and chords were extracted. We manually transcribed the audio of the piano and guitar into a new score in order to also extract descriptors from the *performed* score. The audio mix was used for beat tracking, and to compute a mean tempo. Thirdly, machine learning techniques were applied using the different data sets created from the extracted data to predict the calculated performance actions (Section 4.3).

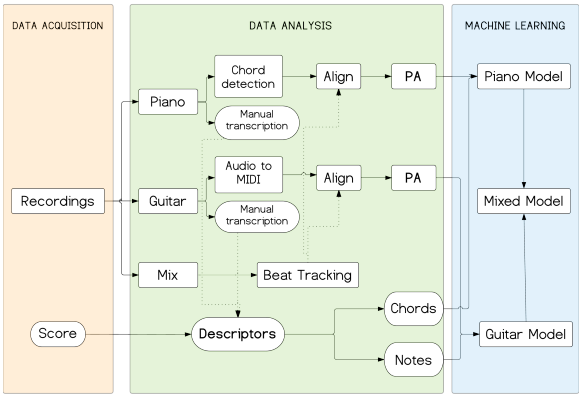


Figure 1. Overall framework: the data related to the score is shown inside ellipses while the data related to the recordings is placed inside rectangles.

4.1 Data acquisition and pre-processing

We recorded a jazz ensemble consisting of keyboard, electric guitar, electric bass and drums. We only used the guitar data in wav format, the piano data in MIDI format and an audio mix of the band in wav format for further tempo computation. Improvisers usually play the main melody at the beginning and end of a performance with improvisations in the central part and so both the recordings and the scores were segmented in order to contain only the melody part (no introductions or solos).

4.2 Data analysis

The aim of this part was to obtain a machine readable representation from the input data (recordings and scores) in the form of **descriptors** (data extracted from the score which characterised both notes and chords by taking into account their properties and the properties of their contexts) and **performance actions** (deviations from the score introduced by the musician to add expressivity, extracted from the recordings). In this stage, there were 4 types of input data: piano recordings, guitar recordings, scores and audio mix recordings. The following Sections explain the processing of this data.

4.2.1 Piano Data

We detected chords in the piano data by grouping together individual notes. The process consisted of identifying

groups of notes played at the same time and so we created heuristic rules based on the work done by Traube and Bernays [23], who identify groups of notes which have near-synchronous onsets by analysing onset differences between two consecutive notes. Our approach consisted of three rules: the first one searched for and grouped notes which were played at the same time. The second one, was in charge of merging chords with an inter onset difference $< 100ms$. Finally, the third rule took into consideration pedal notes (notes that remain while two or more chords are played consequently).

Alignment was performed to link the detected chords with the score chords. It was done at a beat level: since the position of the beats was computed by the beat tracker (see Section 4.2.4), we converted the onsets/offsets of each performed chord from seconds to beats. Based on beat information, we aligned a chord written in the score with the chord (or chords) that had been played.

Based on the alignment of the played chords to the score, the performance actions for every chord in the score were calculated according to what had been played. We computed three performance actions: **density** (Equation 1), defined as *low* or *high* depending on the number of chords used to perform a score chord (i.e. chords played with a duration of half-note or more were labelled as low while a duration of less than a half note corresponded to a "high" label); **weight** (Equation 2), defined as *low* or *high* according to the total number of notes which were utilised to perform a score chord; and **range** (Equation 3), defined as *low* or *high* if the distance in semitones from the highest to the lowest performed note per chord in the score was larger than 18 (an octave and a half) .

$$den(chord_S) = \begin{cases} low & \text{if } \frac{\sum chords_P}{dur(chord_S)} < 1/2 \\ high & \text{if } \frac{\sum chords_P}{dur(chord_S)} \geq 1/2 \end{cases} \quad (1)$$

Where:

$chord_S$: is the corresponding chord on the score

$\sum chords_P$: is the amount of performed chords for a chord on the score

$dur(chord_S)$: is the duration of the corresponding chord on the score

$$wei(chord_S) = \begin{cases} low & \text{if } \frac{\sum notes_P}{\sum chords_P} < 4 \\ high & \text{if } \frac{\sum notes_P}{\sum chords_P} \geq 4 \end{cases} \quad (2)$$

Where:

$chord_S$: is the corresponding chord on the score

$\sum notes_P$: is the total number of performed notes for a chord on the score

$\sum chords_P$: is the amount of performed chords for a chord on the score

$$ran(chord_S) = \begin{cases} low & \text{if } max(pitch_{PN}) - min(pitch_{PN}) < 18 \\ high & \text{if } max(pitch_{PN}) - min(pitch_{PN}) \geq 18 \end{cases} \quad (3)$$

Where:

$chord_S$: is the corresponding chord on the score

$pitch_{PN}$: is the vector of pitch of the performed notes (PN) for a chord on the score

4.2.2 Guitar Data

We automatically converted the guitar recording in wav format into a MIDI format in order to obtain a note representation based on pitch, onset (in seconds) and offset (in seconds) by following the framework presented in Bantula et al. work [2].

Alignment was then performed at two levels. Firstly, the onsets and offsets of the MIDI notes were converted from seconds to beats using the beats' information computed by the beat tracker (see Section 4.2.4). Secondly, we performed manual alignment between the performed notes and the score notes by using a graphical interface that allowed to link the performed notes and the score notes in two pianoroll representations [11]. Embellishments were computed by following the same approach by Giraldo and Ramírez [11]: a note was considered to be embellished if two or more notes were played in its place. Then, each score note was labelled as embellished or not (y/n) according to the previous alignment.

4.2.3 Score Data

In this stage, we extracted horizontal and vertical descriptors from the score to characterise both chords and notes.

- **Chord Descriptors (Figure 2)** For chords, the horizontal context concerned harmonic information and the vertical context considered melodic, ensemble information. In Table 1, the intrinsic descriptors of the reference chords are listed. In Table 3, the harmonic horizontal descriptors, computed according to the neighbours of the reference chord are shown. Table 2 includes the vertical descriptors computed by averaging or weighting the single note descriptors of the notes below the region defined by the reference chord.

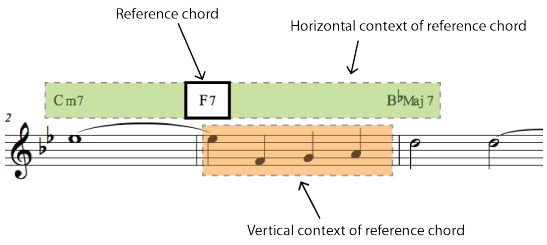


Figure 2. Excerpt of *Autumn Leaves*: horizontal and vertical contexts for the reference chord F7

descriptor	units	computation	range
id	num	$root \rightarrow number$	[0,11]
type	label	$type$	{M, m, +,7, dim, half_dim}
tens	label	tension based on musical criteria	{++, +, -, - -}
chord_dur	beats	$chord\_dur$	[1,∞)
on_b	beats	on_b	[1,∞)

Table 1. Individual descriptors for a reference chord (no context).

descriptor	units	computation	range
onset_b	beats	$min(onset_{notes})$	[1,∞)
dur_b	beats	$max(onset_{notes})$ $+max(dur_{notes})$ $-min(onset_{notes})$	[1,∞)
meanPitch (mP)	MIDI note	$mean(pitch_{notes})$	[36,96]
onset_s	seconds	$60 * \frac{onset\_b}{tempo}$	[1,∞)
dur_s	seconds	$60 * \frac{dur\_b}{tempo}$	[1,∞)
chroma	half tones	$mod_{12}(mP)$	[0,11]
measure	num	$measure$	[1,∞)
pre_dur_b	beats	pre_dur_b	[1,∞)
pre_dur_s	seconds	$60 * \frac{pre\_dur\_b}{tempo}$	[1,∞)
nxt_dur_b	beats	nxt_dur_b	[1,∞)
nxt_dur_s	seconds	$60 * \frac{nxt\_dur\_b}{tempo}$	[1,∞)
prev_int	half tones	$prev_{mP} - mP$	[1,∞)
next_int	half tones	$mP - next_{mP}$	[1,∞)
note2key	half tones	$chroma - key$	[0,11]
note2chord	half tones	$chroma - id$	[0,11]
isChordN*	label	-	{y,n}
mtr*	label	$mean(met_{pos}(notes))$	{strong, weak}
intHop*	num	$mean(intervals)$	[0,96]
melody*	num	$\frac{\#notes}{chord\_dur}$	-

Table 2. Chord melodic descriptors (vertical)

descriptor	units	computation	range
tempo	bpm	$tempo$	[1,300]
keyMode	label	$keyMode$	{major, minor}
numKey	num	key position in the Fifths Circle	[0,11]
keyDistance	half tones	$id - numKey$	[0,11]
metP*	label	metrical position	{strongest, strong, weak, weakest}
function	label	harmonic analysis from $keyDistance$	{tonic, subdom, dom, no_func}
next_root_int	half tones	$id - next_{id}$	[0,11]
prev_root_int	half tones	$prev_{id} - id$	[0,11]

Table 3. Chord harmonic descriptors (horizontal)

- **Note descriptors (Figure 3)** For note descriptors, the horizontal context included melodic information while the vertical context included harmonic, ensemble information. Following the approach made by Giraldo [7], we computed horizontal note descriptors using the information of the reference notes' neighbours whereas we computed vertical note descriptors by using the chords' information. Since every note belonged to a chord, the features of the note were merged with the descriptors of the corresponding chord by concatenating both lists and eliminating repeated items.

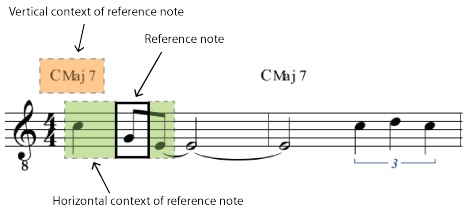


Figure 3. Excerpt of *All Of Me*: Horizontal and Vertical contexts for a reference note

4.2.4 Audio mix Data

For every recording, we performed a semi-automatic alignment between the performance and the score.The tempo varied during the performance because no metronome was used. Hence, beat positions were not equidistant and beat-tracking was performed to create a beat grid which allowed to link the performed information to the score information. We used the algorithm developed by Zapata et al. [27] to track the beats, followed by manual correction. Afterwards, the mean tempo of each song was computed using Equation 4, where $beats$ was the vector of beats computed in the previous step.

$$tempo = round\left(\frac{60}{mean(diff(beats))}\right) \quad (4)$$

4.3 Machine learning

4.3.1 Datasets

As it can be seen in Figure 1, the inputs of the Machine Learning stage were the performance actions for both piano and guitar as well as the score descriptors (for chords and notes). Hence, we constructed three types of datasets, shown in Figure 4.

- **Simple Datasets (D1)**: Horizontal score context. It only contained individual descriptors of the chords or notes.
- **Score Mixed Datasets (D2)**: D1 plus vertical **score** context, which contained merged descriptors of chords and notes.
- **Performance Mixed Datasets (D3)**: D1 plus vertical **performance** context (extracted from the manual

transcriptions of the performances), which contained merged features of chords and notes, taking into account the real interaction between musicians.

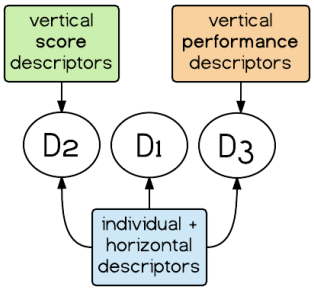


Figure 4. Three different datasets depending on the included descriptors

Therefore, for piano we trained models to learn the function shown in Equation 5 while for guitar, the function to learn is presented in Equation 6.

$$f(Chord) \rightarrow (Den, Wei, Ran) \quad (5)$$

Where:

$Chord$: is a chord only characterised by harmonic descriptors plus melodic descriptors (depending on the dataset)

Den, Wei, Ran: are the predicted density, weight and range labels (high/low), respectively.

$$f(Note) \rightarrow (Emb) \quad (6)$$

Where:

$Note$: is a note characterised by the set of melodic descriptors plus harmonic descriptors (depending on the dataset)

Emb: corresponds to the predicted embellishment label (yes/no).

4.3.2 Feature Selection

The aim of this part was to identify specific score descriptors that best described the previously defined performance actions, so as to train models with the most representative ones. Therefore, for every dataset we evaluated the descriptors by their information gain. Tables 4, 5, 6 and 7 show the best ranked descriptors for density, weight, range and embellishments, respectively.

D1	D2	D3
metP	mtr	metP
chord_dur	metP	chord_dur
function	chord_dur	intHop
type	isChordN	isChordN
tens	type	function
metP	tens	type
		tens

Table 4. Selected features for density

D1	D2	D3
tens	tens	tens
function	function	function
chord_dur	type	type
metP	metP	metP
	isChordN	keyMode
	keyMode	isChordN
	mtr	tens

Table 5. Selected features for weight

D1	D2	D3
numKey	numKey	numKey
function	dur_s	pre_dur_s
type	duration_b	prev_int
tens	pre_dur_b	function
keyMode	nxt_dur_b	type
metP	isChordN	mtr
	function	isChordN
	mtr	tens
	type	keyMode
	tens	metP
	keyMode	
	metP	

Table 6. Selected features for range

D1	D2	D3
phrase	phrase	phrase
dur_b	dur_b	dur_b
dur_s	dur_s	dur_s
pre_dur_b	pre_dur_b	pre_dur_b
pre_dur_s	pre_dur_s	pre_dur_s
onset	onset	onset
	tens	tens
	type	type
	function	function
	isChordN	isChordN
	keyMode	keyMode
		metP

Table 7. Selected features for embellishments

4.3.3 Algorithms

The aim of this stage was to compare the results of the widely used algorithms *Decision Trees*, *Support Vector Machine (SVM)* (with a linear kernel) and *Neural Networks (NN)* (with one hidden layer). We used the implementation of these algorithms in the Weka Data Mining Software [16], utilising the default parameters.

5. RESULTS

Since every performance action contained 3 datasets, we generated a model for each of them. Thus, the results we present include a comparison between the datasets as well as the algorithms.

5.1 Piano data: density, weight and range

We evaluated the accuracy (percentage of correct classifications) using 10-cross fold validation with 10 iterations. We performed statistical testing by using the t-test with a significance value of 0.05 to compare the methods with the baseline (Zero Rule Classifier) and decide if one produced measurably better results than the other.

Table 8 shows the results for **density**. It can be seen that the accuracy increased when ensemble information was considered (datasets D2 and D3). The significant improvements were achieved by the algorithms NN and SVM, being 65.13 the highest accuracy reached with the dataset D2 which consisted in both harmonic and melodic score descriptors. For **weight** (Table 9), none of the results was statistically significant and the performance of the three models can be interpreted as random. The highest results were achieved when only piano information was considered (D1), showing no interaction between this performance action and the guitar melody. Table 10 presents the results for **range**. In that case, the three algorithms reached their maximum accuracy when information of the ensemble performance (D3) was considered, which can be explained as a presence of correlation between the range of the chords performed and the melody the piano player was hearing. Moreover, the results for the algorithms Decision Trees and SVM were statistically significant.

Dataset	Baseline	NN	SVM	Decision Tree
D1	51.82	61.19 ○	62.13 ○	53.75
D2	51.82	61.72	65.13 ○	55.34
D3	51.82	55.75	61.75	57.65

○, ● statistically significant improvement or degradation

Table 8. Accuracy for the models of density in comparison to the baseline using NN, SVM and Decision Trees

Dataset	Baseline	NN	SVM	Decision Tree
D1	53.73	63.52	52.96	54.48
D2	53.73	50.62	49.64	51.85
D3	53.73	57.70	50.90	51.36

○, ● statistically significant improvement or degradation

Table 9. Accuracy for the models of weight in comparison to the baseline using NN, SVM and Decision Trees

Dataset	Baseline	NN	SVM	Decision Tree
D1	56.73	54.51	62.06	63.72
D2	56.73	57.11	60.90	60.93
D3	56.73	58.83	67.85 ○	67.98 ○

○, ● statistically significant improvement or degradation

Table 10. Accuracy for the models of range in comparison to the baseline using NN, SVM and Decision Trees

5.2 Guitar data: embellishments

In that case, there was a skewed classes distribution, which led us to evaluate the sensitivity (true positive rate) rather than the accuracy of the model. Table 11 presents the results obtained. It can be seen that, despite the low percentage of sensitivity, the results for the three algorithms increased when considering ensemble information (D2, D3).

Dataset	NN	SVM	Decision Tree
D1	26	20	12
D2	30	38	26
D3	30	32	24

Table 11. Sensitivity percentage for embellishments

6. CONCLUSIONS

In this work we have developed a system which studies the interaction between musicians by using techniques related to computational analysis of expressive music performance and machine learning. We have created a database consisting of recordings of 7 jazz standards played by a quartet (piano, guitar, bass and drums) and their corresponding scores. For processing both the recordings and the scores, we have developed code libraries consisting of specific functions for every stage of the process: select chords, extract vertical and horizontal descriptors for both notes and chords, align and compare the recordings with the score and extract performance actions. Finally, we have generated models for different datasets consisting of information from individual performances and ensemble performances. Based on the accuracy and sensitivity of the models, we have obtained numerical results which have allowed us to estimate the level of interaction between musicians. The data analysis indicated that, in general terms, the performance actions of the accompaniment are influenced by the soloist and vice versa, since both written and performed descriptors contributed to a better performance of the models.

In a future work, it would be interesting to extract other performance actions such as energy or duration for both chords and notes and to study the extent to which the measures are sensitive to the incorporation of other instruments. Moreover, since we have at our disposal a database which contains the recordings of bass and drums, it would be interesting to incorporate both instruments into the analysis. We have observed that the majority of the models got better results with ensemble information but the accuracies of the models could still improve by collecting more data (making new recordings) or extracting more descriptors. Finally, the parameters of the used algorithms could be further investigated so as to improve the results.

7. ACKNOWLEDGEMENTS

This work has been partly sponsored by the Spanish TIN project TIMUL (TIN2013-48152-C2-2-R) and the H2020-ICT-688268 TELMI project.

8. REFERENCES

[1] Josep Lluís Arcos, Ramon Lopez De Mantaras, and Xavier Serra. Saxex: A case-based reasoning system for generating expressive musical performances*. *Journal of New Music Research*, 27(3):194–210, 1998.

[2] Helena Bantulà, Sergio Giraldo, and Rafael Ramírez. A Rule-based System to Transcribe Guitar Melodies. In *Inernational Symposium on Computer Music Multidisciplinary Research (CMMR)*, pages 1–8, 2015.

[3] Alain de Cheveigne and Hideki Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917, 2002.

[4] Tuomas Eerola and Petri Toiviainen. MIDI toolbox: MATLAB tools for music research. *University of Jyväskylä, Jyväskylä, Finland*, 2004.

[5] Anders Friberg. *A quantitative rule system for musical performance*. PhD thesis, PhD Thesis, KTH, Sweden, 1995.

[6] Alf Gabrielsson. The performance of music. *The psychology of music*, 2:501–602, 1999.

[7] Sergio Giraldo. Modelling embellishment, duration and energy expressive transformations in jazz guitar, 2012.

[8] Sergio Giraldo and Rafael Ramírez. Computational generation and synthesis of jazz guitar ornaments using machine learning modeling. In *International Workshop on Machine Learning and Music (MML)*, 2015.

[9] Sergio Giraldo and Rafael Ramírez. Computational Modeling and Synthesis of Timing, Dynamics and Ornamentation in Jazz Guitar Music. In *Inernational Symposium on Computer Music Multidisciplinary Research (CMMR)*, 2015.

[10] Sergio Giraldo and Rafael Ramírez. Computational modeling of ornamentation in jazz guitar music. In *International Symposium on Performance Science*, 2015.

[11] Sergio Giraldo and Rafael Ramírez. Score Sequence Matching for Automatic Ornament Detection in Jazz Music. In *Inernational Conference on New Music Concepts (ICNMC)*, 2015.

[12] Werner Goebl and Caroline Palmer. Synchronization of timing and motion among performing musicians. 2009.

[13] Talar Hopyan, Maureen Dennis, Rosanna Weksberg, and Cheryl Cytrynbaum. Music skills and the expressive interpretation of music in children with williams-beuren syndrome: pitch, rhythm, melodic imagery, phrasing, and musical affect. *Child Neuropsychology*, 7(1):42–53, 2001.

[14] Apple Inc. Logic pro x. http://www.apple.com/logic-pro/.

[15] Marco Marchini. *Analysis of Ensemble Expressive Performance in String Quartets: A Statistical and Machine Learning Approach*. PhD thesis, PhD Thesis, UPF, Barcelona, 2014.

[16] Geoffrey Holmes Bernhard Pfahringer Peter Reutemann Ian H. Witten Mark Hall, Eibe Frank. *"The WEKA Data Mining Software: An Update"*, volume 11. 2009.

[17] MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[18] MuseScore. *version 1.3*. 2013.

[19] Caroline Palmer. Music performance. *Annual review of psychology*, 48(1):115–138, 1997.

[20] Rafael Ramirez and Amaury Hazan. Modeling expressive music performance in jazz. In *FLAIRS Conference*, pages 86–91, 2005.

[21] Bruno H Repp. Sensorimotor synchronization: a review of the tapping literature. *Psychonomic bulletin & review*, 12(6):969–992, 2005.

[22] John Rink. *Musical performance: a guide to understanding*. Cambridge University Press, 2002.

[23] Caroline Traube and Michel Bernays. Piano Touch Analysis: a Matlab Toolbox for Extracting Performance Descriptors from High Resolution Keyboard and Pedalling Data. *Journées d'Informatique Musicale (JIM)*, 2012.

[24] G. De Poli A. Friberg R. Bresin W. Goebl, S. Dixon and G. Widmer. *Sound to sense-sense to sound: a state of the art in sound and music computing*, chapter Sense in expressive music performance: Data acquisition, computational studies, and models, pages 195–242. Berln, 2008.

[25] Gerhard Widmer and Asmir Tobudic. Playing mozart by analogy: Learning multi-level timing and dynamics strategies. *Journal of New Music Research*, 32(3):259–268, 2003.

[26] Alan M Wing, Satoshi Endo, Adrian Bradbury, and Dirk Vorberg. Optimal feedback correction in string quartet synchronization. *Journal of The Royal Society Interface*, 11(93):20131125, 2014.

[27] José R. Zapata, André Holzapfel, Matthew E.P. Davies, Joao Lobato Oliveira, and Fabien Gouyon. Assigning a Confidence Threshold on Automatic Beat Annotation in Large Datasets. In *ISMIR 2012, Proceedings of the 13th International Society for Music Information Retrieval Conference.*, number ISMIR, pages 157–162, 2012.

MINING MUSICAL TRAITS OF SOCIAL FUNCTIONS IN NATIVE AMERICAN MUSIC

Daniel Shanahan[1] **Kerstin Neubarth**[2] **Darrell Conklin**[3,4]
[1] Louisiana State University, Baton Rouge, LA, USA
[2] Canterbury Christ Church University, United Kingdom
[3] University of the Basque Country UPV/EHU, San Sebastian, Spain
[4] IKERBASQUE, Basque Foundation for Science, Bilbao, Spain

ABSTRACT

Native American music is perhaps one of the most documented repertoires of indigenous folk music, being the subject of empirical ethnomusicological analyses for significant portions of the early 20th century. However, it has been largely neglected in more recent computational research, partly due to a lack of encoded data. In this paper we use the symbolic encoding of Frances Densmore's collection of over 2000 songs, digitized between 1998 and 2014, to examine the relationship between internal musical features and social function. More specifically, this paper applies contrast data mining to discover global feature patterns that describe generalized social functions. Extracted patterns are discussed with reference to early ethnomusicological work and recent approaches to music, emotion, and ethology. A more general aim of this paper is to provide a methodology in which contrast data mining can be used to further examine the interactions between musical features and external factors such as social function, geography, language, and emotion.

1. INTRODUCTION

Studying "musical universals" in the context of contemporary theories of music evolution, Savage *et al.* [23] argue that many of the most common features across musical cultures serve as a way of facilitating social cohesion and group bonding (see also [2, 18]). The focus of their analysis, however, is on comparing geographical regions without systematically differentiating between social contexts and functions of music making. Across these regions, the authors look for links and elements of "sameness". The application and methodology presented here can be viewed as complementary to the earlier study [23]. Firstly, we focus on the relationship between internal musical features (such as pitch range, melodic or rhythmic variability) and the specific social function ascribed to songs rather than feature distributions across geographic regions. Secondly, using computational techniques we study features that can *contrast* between different social functions within a culture, rather than those that are potentially *universal* in music.

The folk songs of Native American groups provide a convenient starting point for the analysis of social function and musical features: a large number of pieces has been recorded by (relatively few) individuals who often annotated the music with an explicit social function. Nettl commented in 1954 that "more musical material [was] available from this large area [...] than from any other of similar size" [20, p. 45]. The collection created by Frances Densmore [25] covers repertoires from five out of the six musical areas postulated by Nettl. Densmore collected songs by Native American groups (see Table 1), and recorded the social usage of songs, ranging from the general (e.g. war songs) to the specific (e.g. songs of the corn dance). Building on Densmore's work, Herzog [10] discussed four categories of social function in music of the North American Plains, specifically love songs, songs of hiding games, ghost dance songs, and songs in animal stories. Employing quantitative analysis, Gundlach also compared songs used in different situations, e.g. war songs or healing songs; groups of songs were taken as proxies for studying mood, specifically asking if "objective characteristics of a piece of music form the basis for the mood which it may arouse" [7, pp. 134-135]. Interestingly, Gundlach found a diversity in the treatment of some musical features to convey emotion across indigenous groups, such as larger intervals mainly associated with "sad" love songs among the Chippewa and Ojibway but with "happy" love songs among the Teton-Sioux [7, p. 139].

This paper builds upon Gundlach's work, exploring quantitative analysis to identify musical traits of songs associated with different social functions. More specifically, we adopt contrast data mining [1, 5, 21], a type of descriptive supervised data mining. In the context of music information retrieval, supervised data analysis has been largely dominated by predictive classification, i.e. building models that discriminate labeled groups in data and predict the group label of unseen data instances. Classifiers are generally treated as a black box, and results tend to focus on predictive accuracy. By comparison, contrast data mining aims to discover distinctive patterns which offer an understandable symbolic description of a group.

© Daniel Shanahan, Kerstin Neubarth, Darrell Conklin. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Daniel Shanahan, Kerstin Neubarth, Darrell Conklin. "Mining Musical Traits of Social Functions in Native American Music", 17th International Society for Music Information Retrieval Conference, 2016.

Discovered patterns are discussed both in light of ethnomusicological writings such as those by Densmore, Herzog and Gundlach, and in the context of research into music and emotion. The music information retrieval community has engaged with models and classifiers of emotion in music from many different perspectives and utilizing a wide range of approaches. For example, Han *et al.* [8] implemented support vector regression to determine musical emotion, and found their model to correlate quite strongly with a two-dimensional model of emotion. Schmidt and Kim used conditional random fields to model a dynamic emotional response to music [24]. For a thorough review of emotional models in music information retrieval, see Kim *et al.* [14], and for an evaluation and taxonomy of the many emotional approaches to music cognition, see Eerola and Vuoskoski [6]. Unlike these studies, the current study does not attempt to model emotion or provide a method of emotion classification, but considers findings from emotion and ethological research in discussing the mining results.

2. THE DENSMORE CORPUS

Frances Densmore's transcriptions of Native American folksongs provide an invaluable dataset with which we might examine issues pertaining to geography, language, and culture. As Nettl points out, many of the earlier recordings were conducted in the very early days of field recording, and contain performances from elderly individuals who had little contact and influence from the Western musical tradition [20]. The fact that this collection was transcribed by a single individual, covers such a large geographic area, and focuses on cultures with disparate social and linguistic norms, makes it immensely useful for studies of large-scale relationships between music and language, geography, and social function.

Interest in digitally encoding Frances Densmore's collection of Native American songs began in the late 1990s, when Paul von Hippel encoded excerpts of the first book of Chippewa songs in 1998 into Humdrum's **kern format. David Huron encoded the Pawnee and Mandan books in 2000, and Craig Sapp encoded the lengthy Teton Sioux book in 2002. In 2014, Eva and Daniel Shanahan encoded the remaining books into **kern format [25]. The digitized collection contains 2,083 folksongs from 16 books (Table 1), collected between 1907 and 1958. [1]

The Densmore volumes provide a rich source of information because they not only give transcriptions of all the collected songs, but also additional information – including the associated social function – and musical analyses. Densmore's annotations were integrated as metadata into the digital collection. As exact phrasings and annotation criteria vary across the chronological span of Densmore's writing, the metadata vocabulary was prepared by cleaning and generalizing social function terms: firstly, inconsistent phrasings were harmonized, e.g. "hand game songs" (Northern Ute book) and "songs of the hand game" (Cheyenne and Arapaho book). Secondly, functions were merged to create generalized functions, e.g. different game songs such as hand game songs and moccasin game songs were collated into one group "game songs" (see Fig. 1).

[1] The corpus is available at `musiccog.lsu.edu/densmore`

Book	Year published
Chippewa I	1910
Chippewa II	1913
Teton Sioux	1918
Northern Ute	1922
Mandan and Hidatsa	1923
Papago	1929
Pawnee	1929
Menominee	1932
Yuman and Yaqui	1932
Cheyenne and Arapaho	1936
Nootka and Quileute	1939
Indians of British Columbia	1943
Choctaw	1943
Seminole	1956
Acoma, Isleta, Cochiti, and Zuñi Pueblos	1957
Maidu	1958

Table 1. Collections included in the Densmore corpus.

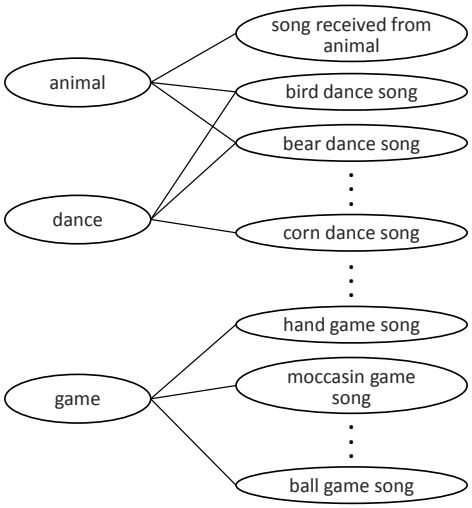


Figure 1. Excerpt of the social functions ontology.

Songs which Densmore listed as "uncategorized" or "miscellaneous" were not considered. The resulting ontology reduces the 223 distinct terms used by Densmore to 31 generalized functions. Note that songs can be assigned more than one function, e.g. bird dance songs are annotated as both "animal" and "dance" (see Fig. 1).

3. CONTRAST DATA MINING

Contrast data mining [1, 5] refers to a range of methods which identify and describe differences between groups in a dataset, and has been applied with success to several folk song corpora [21]. In the current study with the Densmore corpus, groups are defined by songs associated with different social functions. Following several other earlier works on contrast data mining in folk music analysis, in

Attribute	Definition	High (H)	
AverageMelodicInterval	average melodic interval in semitones	$\geq$	1.676
AverageNoteDuration	average duration of notes in seconds	$\geq$	0.347
DirectionofMotion	fraction of melodic intervals that are rising rather than falling	$\geq$	0.388
Duration	total duration of piece in seconds	$\geq$	22.294
DurationofMelodicArcs	average number of notes that separate melodic peaks and troughs	$\geq$	1.704
PitchVariety	number of pitches used at least once	$\geq$	6.583
PrimaryRegister	average MIDI pitch	$\geq$	55.578
Range	difference between highest and lowest MIDI pitches	$\geq$	13.084
RepeatedNotes	fraction of notes that are repeated melodically	$\geq$	0.462
SizeofMelodicArcs	average melodic interval separating the top note of melodic peaks and bottom note of melodic troughs	$\geq$	4.899
StepwiseMotion	fraction of melodic intervals corresponding to a minor or major second	$\geq$	0.250
VariabilityofNoteDuration	standard deviation of note durations in seconds	$\geq$	0.224
DcontRedundancy	duration contour relative redundancy	$\geq$	0.749
DurRedundancy	note duration relative redundancy	$\geq$	0.667
IntRedundancy	melodic interval relative redundancy	$\geq$	0.606
MIRedundancy	metric level relative redundancy	$\geq$	0.681
PcontRedundancy	melodic pitch contour relative redundancy	$\geq$	0.751
PitchRedundancy	pitch relative redundancy	$\geq$	0.603

Table 2. A selection of attributes used in this study. Top: jSymbolic attributes [17]. Bottom: information-theoretic attributes. The rightmost column indicates the value range for the discretisation bin High.

this study songs are described by *global features* which are attribute-value pairs each describing a song by a single value. Global features have been used productively in computational folk music analysis in the areas of classification (e.g. [11, 17, 27]) and descriptive mining (e.g. [16, 26]).

It is important to highlight the distinction between attribute *selection* and contrast data mining. Whereas the former is the process of selecting informative *attributes*, usually for the purposes of classifier construction, contrast data mining is used to discover *particular* attribute-value pairs (features) that have significantly different supports in different groups.

3.1 Global feature representation

All songs in the corpus were converted to a MIDI format, ignoring percussion tracks and extracting one single melodic spine for each song. Since only a fraction of the songs in the corpus were annotated with tempo in the **kern files, all songs were standardized to a tempo of $\text{♩} = 60$. This was followed by computing 18 global attributes: twelve attributes from the jSymbolic set [17] and six newly implemented information-theoretic attributes. After discarding attributes not applicable to the current study such as those related to instrumentation, dynamics, or polyphonic texture, the twelve jSymbolic attributes were selected manually, informed by Densmore's own writings, additional ethnomusicological studies of Native American music [7, 10, 20] and research into music and emotions [6, 13, 22]. The six information-theoretic attributes measure the relative redundancy *within a piece* of a particular event attribute (pitch, duration, interval, pitch contour, duration contour, and metric level). The features are defined as $1 - H/H_{\max}$ where H is the entropy of the event attribute in the piece and the maximum entropy $H_{\max}$ is the logarithm of the number of distinct values of the attribute in the piece. The value of relative redundancy therefore ranges from 0 (low redundancy, i.e. high variability) to 1 (high redundancy, i.e. low variability) of the particular attribute. Numeric features were discretized into categorical values, with a split point at the mean: the value Low covers attribute values below the average across the complete dataset, the value High covers attribute values at the average or above (cf. [26]). Table 2 gives definitions for the attributes which contribute to the contrast patterns reported in Section 4.

3.2 Contrast data mining method

Global features are assessed as candidate contrast patterns by evaluating the difference in pattern support between different groups (e.g. [1, 5]). A feature (attribute-value pair) is *supported* by a song if the value of the attribute is true for the song. Then the support $n\,(X \wedge G)$ of a feature X in a group G is the number of songs in group G which support feature X. A feature is a *contrast pattern* for a certain group if its support in the group, $n\,(X \wedge G)$, is significantly higher or lower than in the remaining groups taken together, $n\,(X \wedge \neg G)$. This is known as a *one-vs.-all strategy* for contrast mining [5, 21] as it contrasts one group against the combined set of other groups rather than contrasting groups in pairs. The significance of a pattern, that is, how surprising is the under- or over-representation of X in G, can be quantified using the hypergeometric distribution (equivalent to *Fisher's exact test*). This uses a 2×2 contingency table (see Table 3) which gives the

	G	$\neg G$	
X	$n(X \wedge G)$	$n(X \wedge \neg G)$	$n(X)$
$\neg X$	$n(\neg X \wedge G)$	$n(\neg X \wedge \neg G)$	$n(\neg X)$
	$n(G)$	$n(\neg G)$	N

Table 3. Contingency table showing the occurrence of a pattern X and its complement $\neg X$ in a target group G and in the background $\neg G$. The highlighted area is the support of the putative contrast pattern $X \wedge G$. For the Densmore corpus $N = 2083$.

probability of sampling $n(X)$ pieces, and finding exactly $n(X \wedge G)$ successes (instances of group G). Thus the left or right tails of the hypergeometric distribution give the two desired p-values: the probability of observing at most or at least $n(X \wedge G)$ instances in a single random sample of $n(X)$ instances. A low p-value, less than some specified significance level α, indicates a statistically significant contrast pattern which is assumed to be interesting for further exploration [3].

Following the extraction of features as described in Section 3.1, each song in the corpus is represented by a set of global features together with a set of group labels (functions) of the song. Note that, as mentioned above, more than one function can be assigned to a song. From this input dataset candidate patterns are generated as the cross-product for all occurring pairs of features X and groups G. For each $X \wedge G$ pair its support and a p-value for each tail are computed and the results processed to form a matrix of function/feature patterns.

4. RESULTS AND DISCUSSION

A total of 17 social function groups (uncategorized and miscellaneous songs and groups supported by less than ten songs were not considered) were mined for contrast pairs with 18 attributes. The 17 groups together cover most of the corpus: 1891 of the 2083 songs. Regarding the attributes for global features, though each has two possible values High (H) and Low (L), if one is significantly over-represented the other must be significantly under-represented, therefore in this study only the High value was considered during mining. Table 4 presents the results of the contrast data mining. Each cell in the matrix shows the distribution of a particular feature in a particular group. White indicates presence of the feature in the group (with area $n(X \wedge G)$) and black absence (with area $n(\neg X \wedge G)$). Thus the total area covered by a cell in a row indexed by group G is $n(G)$. The rows and columns in the table are ordered by the geometric mean of the p-value to all other functions or features in that particular row or column.

Statistical significance of each contrast pattern was evaluated using the hypergeometric distribution as described above, with significance level $\alpha = 0.05$ adjusted using a Bonferroni multiple testing correction factor of $306 = 17 \times 18$, representing the number of contrast patterns tested for significance. Using the adjusted significance level of $0.05/306$ =1.6e-4, green areas in Table 4 indicate significant over-representation, and red areas significant under-representation of a feature in a group. A total of 56 significant patterns were found (colored patterns of Table 4).

As a statistical control, a permutation method was used to estimate the false discovery rate, assuming that most contrast patterns found in randomized data would be artifactual. Social function labels were randomly redistributed over songs, while maintaining the overall function counts, then the number of significant (left or right tail p-value $\leq$ 1.6e-4) contrast patterns using the 306 possible pairs was counted. Repeated 1000 times, this produced a mean of just 1.14 significant contrast patterns per iteration, suggesting that there are few false discoveries to be expected in the colored patterns of Table 4.

Bearing in mind that the exact data samples and feature definitions differ, the results seem to confirm – and generalize to a larger dataset – several observations presented in earlier studies. The significance of PrimaryRegister : H for love songs recalls Gundlach's finding that "love songs tend to be high" [7, p. 138]. Herzog describes the "melodic make-up" of love songs as "spacious" [10, p. 28]: in our analysis we find that love songs generally have larger average melodic intervals and a wider range than other songs. The over-representation of AverageNoteDuration : H in love songs may reflect characterisations of love songs as slow [7, 10]. For hiding game songs of the Plains, Herzog notices that they are comparatively short with a very often limited range [10, p. 29]; game songs in the current corpus – including 90 out of the 143 game songs explicitly associated with hiding games such as moccasin, hand and hiding-stick or hiding-bones games – show a significant under-representation of Duration : H and Range : H. The narrow range that Gundlach observed in healing songs [7, pp. 138,140] is also reflected in the results in Table 4, but in the current analysis is not statistically significant. Gundlach compared healing songs specifically against war and love songs: considering only those two groups as the background does indeed lead to a lower p-value (left-tail) for Range : H in healing songs (6.8e-9 instead of 2.6e-3). Together with other traits which are common in healing songs but not distinctive from other song types — e.g. a high proportion of repeated notes and low variability in pitch, intervals or duration — a comparatively narrow range may contribute to a soothing character of many healing songs, intended "to remove discomfort" [4, p. 565].

The information-theoretic features are particularly characteristic of songs labelled as "nature", which show an over-representation of redundancy values above the average for all six considered event attributes. The group contains 13 Yuman lightning songs, which trace the journey of White Cloud who controls lightning, thunder and storms. More generally, Yuman songs tend to be of comparatively small range, the melodic movement on the whole mainly descending, the rhythm dominated by few duration values and isometric organisation more common than in other repertoires [9,20]; structurally, Yuman music is often based on repeated motifs [9]. The example of the Yuman light-

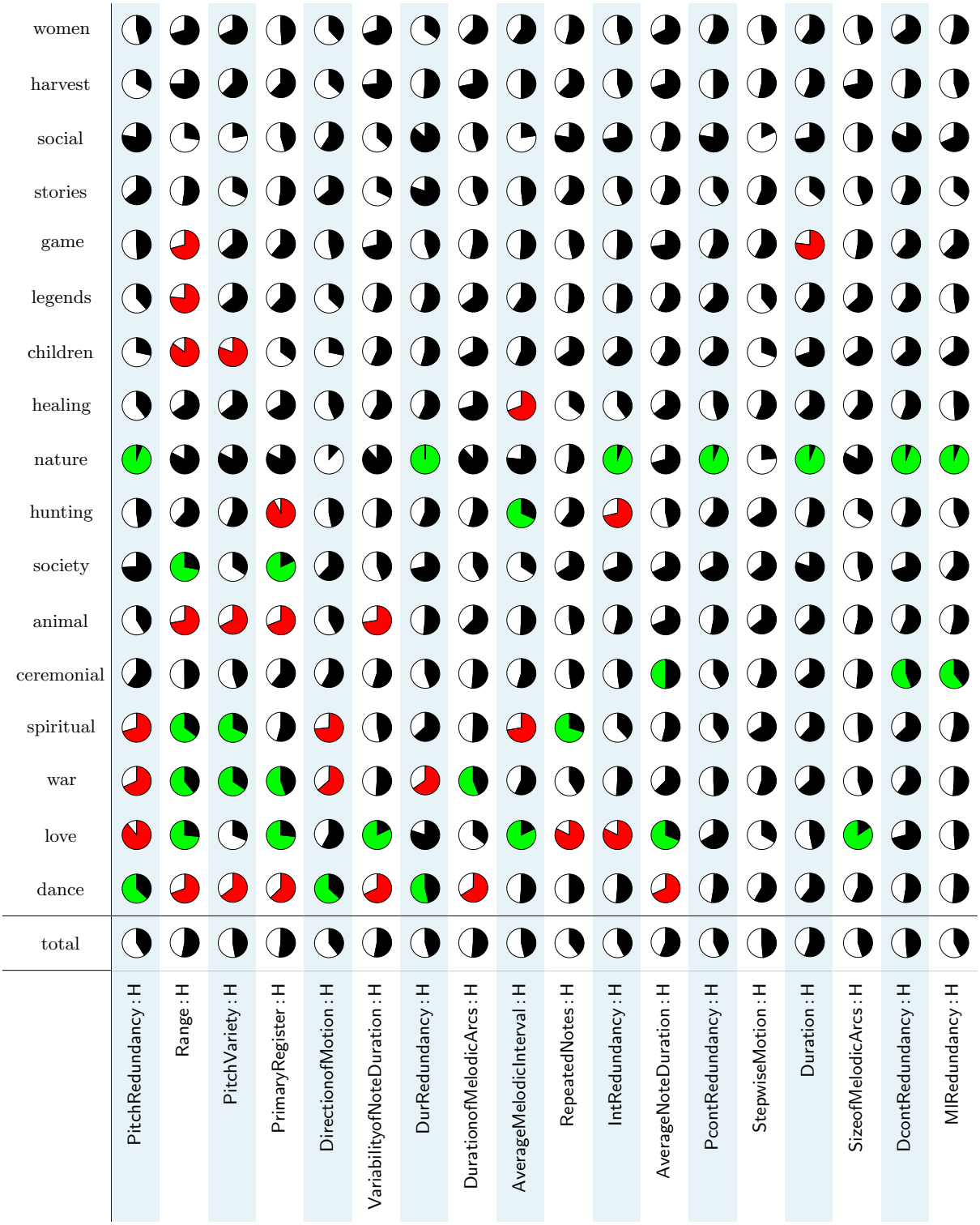


Table 4. Pie charts for contrast patterns showing the distribution of social function groups (rows) against features (columns). White indicates presence and black absence of the corresponding feature. Green/red (light/dark gray in grayscale) indicate significant over/under-representation of a feature in a group.

ning songs opens avenues for future analysis, such as considering also sequential pattern mining [3], and encourages applying data mining to questions left unanswered in earlier work, such as exploring the stylistic unity of songs forming a series related to a myth or ritual [9, p. 184].

It can be productive to discuss the mining results in the context of recent work that takes an "ethological" approach to music and emotion. This approach argues that pitch height, tempo, dynamics, and variability convey levels of both arousal and valence, and many of these relationships are innate, cross-cultural, and cross-species [12, 13, 19]. Similarly to Gundlach's earlier work [7], we find that war songs and love songs exhibit several salient musical traits. In the Densmore collection, war songs are distinguished from other song types by significant over-representation of a wider than average range, higher than average register, and higher variability in both pitch and duration (over-representation of PitchVariety : H and under-representation of PitchRedundancy : H and DurRedundancy : H). Interestingly, dance songs also show significant contrasts in these features, but consistently in the opposite direction compared to war songs. War songs and dance songs might both be thought of as "high arousal", but on opposite ends of the valence spectrum on Russell's Circumplex model [22]. This hypothesis invites further inspection of war and dance songs in the corpus. Significant features shared between dance and animal songs (Range : H, PitchVariety : H, PrimaryRegister : H and VariabilityofNoteDuration : H being under-represented) reflect the fact that many of the supporting songs – e.g. bird, bear or deer dance songs – are annotated with both "dance" and "animal" (see also Fig. 1).

In love songs, the over-representation of higher pitch registers, observed both by Gundlach and in the current study, seems in line with Huron's acoustic ethological model [13], according to which higher pitches (alongside quiet dynamics) connote affiliation. For a Pawnee love song Densmore relates her informant's explanation that in this song a married couple for the first time openly expressed affection for each other. Both Densmore and Gundlach characterize many love songs as "sad", associated with departure, loss, longing or disappointment, which might be reflected in the relatively slow movement of many love songs (see above). Remarkably, though, at first inspection other contrast patterns describing love songs (e.g. under-representation of IntRedundancy : H or over-representation of PrimaryRegister : H) seem at odds with findings on e.g. sad speech which contains markers of low arousal such as weak intervallic variability and lower pitch [15]. However, when comparing observations across studies, their specific feature definitions and analysis methods need to be taken into account. In the current study, significant contrast features are discovered relative to the feature distributions in the dataset, both in terms of feature values and thus the mean value in the corpus (used in discretizing global features into values Low and High), and occurrence across groups (used in evaluating significant over- or under-representation during contrast mining).

5. CONCLUSIONS

This paper has presented the use of descriptive contrast pattern mining to identify features which distinguish between Native American songs associated with different social functions. Descriptive mining is often used for explorative analysis, as opposed to statistical hypothesis testing or predictive classification. Illustrating contrast pattern mining in an application to the Densmore collection, results suggest musical traits which describe contrasts between musics in different social contexts. Different from studies focusing on putative musical universals [23], which test generalized features with disjunctive values (e.g. two- or three-beat subdivisions), and from attribute selection studies [27], which do not specify distinctive values, global-feature contrast patterns make explicit an attribute and value pair which is distinctive for a certain song type. In this case study, mining results confirm findings of earlier ethnomusicological research based on smaller samples, but also generate questions for further investigation.

The Densmore corpus of Native American music provides a rich resource for studying relations between internal musical features and contextual aspects of songs, including not only their social function but also e.g. languages and language families [25], geographical or musical areas [20]. Thus, contrast mining of the Densmore collection could be extended to other groupings. Regarding social functions, the ontology used here possibly could be linked to anthropological taxonomies on functions of musical behaviour (e.g. [2, 18]), whose categories on their own are too broad for the purposes of contrast pattern mining but could open additional interpretations if integrated into hierarchical, multi-level, mining. Regarding pattern representations, the method of contrast data mining is very general and in theory any logical predicate can be used to describe groups of songs. For future work we intend to explore the use of sequential melodic patterns to describe social functions in the Densmore corpus, and also to apply the methods to other large folk song collections.

6. ACKNOWLEDGMENTS

This research is partially supported by the project Lrn2Cre8 which is funded by the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET grant number 610859. The authors would like to thank Olivia Barrow, Eva Shanahan, Paul von Hippel, Craig Sapp, and David Huron for encoding data and assistance with the project.

7. REFERENCES

[1] Stephen Bay and Michael Pazzani. Detecting group differences: Mining contrast sets. *Data Mining and Knowledge Discovery*, 5(3):213–246, 2001.

[2] Martin Clayton. The social and personal functions of music in cross-cultural perspective. In Susan Hallam, Ian Cross, and Michael Thaut, editors, *The Oxford*

Handbook of Music Psychology, pages 35–44. Oxford University Press, 2008.

[3] Darrell Conklin. Antipattern discovery in folk tunes. *Journal of New Music Research*, 42(2):161–169, 2013.

[4] Frances Densmore. The use of music in the treatment of the sick by American Indians. *The Musical Quarterly*, 13(4):555–565, 1927.

[5] Guozhu Dong and Jinyan Li. Efficient mining of emerging patterns: discovering trends and differences. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-99)*, pages 43–52, San Diego, CA, USA, 1999.

[6] Tuomas Eerola and Jonna K. Vuoskoski. A review of music and emotion studies: approaches, emotion models, and stimuli. *Music Perception: An Interdisciplinary Journal*, 30(3):307–340, 2013.

[7] Ralph H. Gundlach. A quantitative analysis of Indian music. *The American Journal of Psychology*, 44(1):133–145, 1932.

[8] Byeong-Jun Han, Seungmin Rho, Roger B. Dannenberg, and Eenjun Hwang. SMERS: Music emotion recognition using support vector regression. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 651–656, Kobe, Japan, 2009.

[9] George Herzog. The Yuman musical style. *The Journal of American Folklore*, 41(160):183–231, 1928.

[10] George Herzog. Special song types in North American Indian music. *Zeitschrift für vergleichende Musikwissenschaft*, 3:23–33, 1935.

[11] Ruben Hillewaere, Bernard Manderick, and Darrell Conklin. Global feature versus event models for folk song classification. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 729–733, Kobe, Japan, 2009.

[12] Leanne Hinton, Johanna Nichols, and John Ohala, editors. *Sound Symbolism*. Cambridge University Press, 1995.

[13] David Huron. Understanding music-related emotion: Lessons from ethology. In *Proceedings of the 12th International Conference on Music Perception and Cognition (ICMPC 2012)*, pages 23–28, Thessaloniki, Greece, 2012.

[14] Youngmoo E. Kim, Erik M. Schmidt, Raymond Migneco, Brandon G. Morton, Patrick Richardson, Jeffrey Scott, Jacquelin A. Speck, and Douglas Turnbull. Music emotion recognition: A state of the art review. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 255–266, Utrecht, The Netherlands, 2010.

[15] Emile Kraepelin. *Psychiatrie. Ein Lehrbuch für Studierende und Ärzte, ed. 2*. Edinburgh: E.& S. Livingstone, 1921.

[16] Mario L.G. Martins and Carlos N. Silla Jr. Irish traditional ethnomusicology analysis using decision trees and high level symbolic features. In *Proceedings of the Sound and Music Computing Conference (SMC 2015)*, pages 455–462, Maynooth, Ireland, 2015.

[17] Cory McKay. *Automatic music classification with jMIR*. PhD thesis, McGill University, Canada, 2010.

[18] Alan P. Merriam and Valerie Merriam. *The Anthropology of Music*. Northwestern University Press, 1964.

[19] Eugene S. Morton. On the occurrence and significance of motivation-structural rules in some bird and mammal sounds. *American Naturalist*, 111(981):855–869, 1977.

[20] Bruno Nettl. North American Indian musical styles. *The Journal of American Folklore*, 67(263,265,266), pages 44–56,297–307,351–368, 1954.

[21] Kerstin Neubarth and Darrell Conklin. Contrast pattern mining in folk music analysis. In David Meredith, editor, *Computational Music Analysis*, pages 393–424. Springer, 2016.

[22] James A. Russell. Core affect and the psychological construction of emotion. *Psychological Review*, 110(1):145–172, 2003.

[23] Patrick E. Savage, Steven Brown, Emi Sakai, and Thomas E. Currie. Statistical universals reveal the structures and functions of human music. *Proceedings of the National Academy of Sciences*, 112(29):8987–8992, 2015.

[24] Erik M. Schmidt and Youngmoo E. Kim. Modeling musical emotion dynamics with conditional random fields. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, pages 777–782, Miami, FL, USA, 2011.

[25] Daniel Shanahan and Eva Shanahan. The Densmore collection of Native American songs: A new corpus for studies of effects of geography and social function in music. In *Proceedings for the 13th International Conference for Music Perception and Cognition (ICMPC 2014)*, pages 206–209, Seoul, Korea, 2014.

[26] Jonatan Taminau, Ruben Hillewaere, Steijn Meganck, Darrell Conklin, Ann Nowé, and Bernard Manderick. Descriptive subgroup mining of folk music. In *2nd International Workshop on Music and Machine Learning at ECML/PKDD 2009 (MML 2009)*, Bled, Slovenia, 2009.

[27] P. van Kranenburg, A. Volk, and F. Wiering. A comparison between global and local features for computational classification of folk song melodies. *Journal of New Music Research*, 42(1):1–18, 2013.

MINING ONLINE MUSIC LISTENING TRAJECTORIES

Flavio Figueiredo[1] **Bruno Ribeiro**[2] **Christos Faloutsos**[3]
Nazareno Andrade[4] **Jussara M. Almeida**[5]
[1] IBM Research - Brazil [2] Purdue University [3] Carnegie Mellon University
[4] Universidade Federal de Campina Grande [5] Universidade Federal de Minas Gerais

ABSTRACT

Understanding the listening habits of users is a valuable undertaking for musicology researchers, artists, consumers and online businesses alike. With the rise of Online Music Streaming Services (OMSSs), large amounts of user behavioral data can be exploited for this task. In this paper, we present SWIFT-FLOWS, an approach that models user listening habits in regards to how user attention transitions between artists. SWIFT-FLOWS combines recent advances in trajectory mining, coupled with modulated Markov models as a means to capture both how users switch attention from one artist to another, as well as how users fixate their attention in a single artist over short or large periods of time. We employ SWIFT-FLOWS on OMSSs datasets showing that it provides: (1) semantically meaningful representation of habits; (2) accurately models the attention span of users.

1. INTRODUCTION

Is it possible to create expressive yet succinct representations of individuals' music listening habits? Are there common patterns on how music is listened to across different genres and different artists that have highly different popularity? For a long time such questions have attracted the attention of researchers from different fields. In the fields of psychology and musicology [10, 20, 21], researchers exploit musical preferences to study social and individual identity [20], mood regulation [23], as well as the underlying factors of preferences [21]. Computer scientists are also tackling such questions as they become central to develop music recommender systems [3, 4, 7].

With the rise of Online Music Streaming Services (OMSSs) over the last decade, large datasets of user [1] behavior can be used to shed light on questions like the ones above. More specifically, digital traces of the listening habits of individuals are readily available to researchers.

In this paper, we focus on the online listening habits of users as trajectories [7] (or trails [24]). Given that a user, u, listens to music by switching attention between different artists, a trajectory captures the sequence of artists or songs visited by a user when listening to music. The main contribution of this paper is to present the SWIFT-FLOWS [2] model, a general technique designed to study user trajectories in OMSSs. We tackle several challenges that stem from the complexity of user behavior, such as:

(a) *Asynchronous users with mixed but similar behavior*: Users that consume music from a set of artists will not start their playlists at the same time or listen to songs in the same order.
(b) *Repeated consumption*: Users tend to listen to artists in bursts, more than what one would expect at random in a shuffled playlist.
(c) *Biased Observations & Small Subpopulations*: User behavior datasets are naturally sparse and biased towards more popular artists. Nevertheless, we still want to be able to analyze underrepresented subpopulations of users and artists.

SWIFT-FLOWS effectiveness is evaluated in large datasets, with results showing that SWIFT-FLOWS: (1) captures semantically meaningful representation of artist transitions; (2) accurately models the attention span of users.

2. RELATED WORK

Understanding the listening habits of individuals has attracted interest from different research fields. Among other problems, musicologists and social psychologists have looked into the latent factors that explain musical preferences [20, 21], factors that affect listener experience (e.g., Music itself, Situational Factors and the Listener him/herself) [10], as well as the relationships between musical imagination and human creativity [10].

Regarding the material methods listeners exploit to listen to music, Nowak [16] discussed the social-material relations of music consumption. The authors conclude that even the same user still relies on multiple forms of listening to music (e.g., legal and illegal downloading, streaming services, CDs, etc). These various forms of consumption were also discussed by Bellogin *et al.* [1]. Here, the au-

[1] Since our case study is on Online Music Streaming Services (OMSSs), we use the terms users and listeners interchangeably.

[2] Switch and Fixation Trajectory Flows

© Figueiredo, Ribeiro, Faloutsos, Andrade, Almeida. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Figueiredo, Ribeiro, Faloutsos, Andrade, Almeida. "Mining Online Music Listening Trajectories", 17th International Society for Music Information Retrieval Conference, 2016.

thors showed the disagreement between different web and social music services (in terms of artist popularity).

Several studies, such as the ones Marques *et al.* [12], Park *et al.* [18], and Moore *et al.* [15], characterized the exploratory behavior of users in OMSSs. Marques *et al.* [12] described the habits of users when searching for novel songs to listen to. Park *et al.* [18] defined a new measure to compute the diverseness of musical tastes. Moore *et al.* [15] looked into the tastes of users over time through the use of Latent Markov Embedding (LME) [3, 4].

In contrast with the aforementioned studies, our work with SWIFT-FLOWS is focused on extracting the latent trajectories that explain user attention when listening to music online. Nevertheless, SWIFT-FLOWS can be used to tackle problems as the ones described. For instance, we are able to aggregate the preferences of users from different demographics as shown in Section 4. For musicologists and psychologists, these results indicate how SWIFT-FLOWS can be used as a tool to better understand the hidden factors that define our consumption habits. Regarding OMSSs, previous work [12, 17, 18] usually relied on defining specific point estimates that are used to understand and capture listening behavior. Such measures are susceptible to effects (a-c) described in Section 1.

To capture both the inter-artist transitions, those were a listener changes attention from one artist to another, as well the long and short tails of listener fixation in a single artist, SWIFT-FLOWS advances the state-of-the-art [7] by defining a combined approach that can capture both behaviors. Inter-artist transitions, or switches in attention, are captured by exploring the ideas in [7]. Intra-artist transitions, or fixation, is captured by modulated Markov models [22]. In this sense, SWIFT-FLOWS provides a interpretable results than [7] or [22] in isolation. We describe the details of the model next.

3. THE SWIFT-FLOWS MODEL

We now describe SWIFT-FLOWS. Let $\mathcal{D}$ be a dataset consisting of $(\text{user}, \text{artist}, \text{timestamp})$ tuples observed over a time window (i.e., the temporal range of our datasets). Each tuple registers that a user listened to songs from an artist at a moment in time. Let $u \in \mathcal{U}$ define the set of users and $a \in \mathcal{A}$ define the set of artists. By ordering $\mathcal{D}$ according to the timestamps, each user can be represented as a trajectory: $T_u =< a_{u,1}, a_{u,2}, ..., a_{u,|\mathcal{T}_u|} >$. This trajectory represents the history of the user listening to music transitioning between songs of a same artist – in intra-artist ($a_{u,i} = a_{u,i+1}$) transitions – and songs from different artists – in inter-artist ($a_{u,i} \neq a_{u,i+1}$) transitions.

Both inter and intra artist transitions are important when studying trajectories. Inter-artist transitions capture a switch in users attention from one artist to another, whereas intra-artist transitions captures a fixation on a same artist. SWIFT-FLOWS isolates both effects and exploits stochastic complementation [14] to propose two complementary Markov models, as illustrated in Figure 1, that together are able to capture both the intra-artist and inter-artist transition behavior. Isolation of intra from inter artist transitions is necessary to model both the long and and short attention tails of repeated consumption [6, 22].

The *intra-artist* (Figure 1-b), or fixation, model consists of a modulated Markov model that is able to capture how users revisit artists. Intra/inter transition separation is possible by treating user attention as a reducible system, where we model the strong memory of intra-artist transitions – some users continuously listen to the same artist for hours – as only interfering with the inter-artist dynamics through limited user attention. This creates an effective separation between the intra-artist model and the inter-artist model. A play takes us to an inter-artist transition from artist s to artist d, $s \neq d$, which then again transitions to the intra-artist model of artist d. The *inter-artist* (Figure 1-c) attention, or switch, transitions are captured by a graphical model, using a Bayesian approach to estimate inter-artist transitions. This approach avoids problems associated with point estimates [7, 19] and is robust to infrequent transitions of small sub-populations of interest.

Data Representation: Let users (artists) to be numbered between one and $|\mathcal{U}|$ ($|\mathcal{A}|$). Let n_{dsu}, the number of times user $u \in \mathcal{U}$ transitioned from $s \in \mathcal{A}$ to $d \in \mathcal{A}$:

$$n_{dsu} = \sum_{i=2}^{|T_u|} \mathbf{1}(a_{u,i-1} = s \wedge a_{u,i} = d), \tag{1}$$

where, $\mathbf{1}$ is an indicator function that will evaluate to 1 when $a_{u,i-1} = s$ and ($\wedge$) $a_{u,i} = d$, 0 otherwise.

With these counts, we can define a tensor $\boldsymbol{\mathcal{X}}$ (as shown in Figure 1-a) $\boldsymbol{\mathcal{X}} = \left[\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_{|\mathcal{U}|}\right]$, where $\mathbf{X}_u$ is:

$$\mathbf{X}_u = \begin{bmatrix} n_{11u} & \cdots & n_{1|\mathcal{A}|u} \\ \vdots & \ddots & \vdots \\ n_{|\mathcal{A}|1u} & \cdots & n_{|\mathcal{A}||\mathcal{A}|u} \end{bmatrix} \tag{2}$$

This data representation is distinct from other tensor decompositions that mine $\mathcal{D}$ in its original "user", "object" and "time" coordinates as the three tensor modes [13, 25]. These techniques are meant to capture synchronous user behavior. As shown in previous work [7], the representation of $\boldsymbol{\mathcal{X}}$ is more suitable to capture the asynchronous but similar behavior patterns that emerge when we have a mixed population of users, spread across different time zones and with different activity patterns as in OMSSs.

We now describe both the inter-artist an intra-artist models. In the following, we use the "·" notation to imply a sum over a given dimension (e.g., $n_{ds\cdot} = \sum_{u \in \mathcal{U}} n_{dsu}$).

3.1 Switch Model

To model inter-artist transitions, we define $\boldsymbol{\mathcal{X}}^{-} = \boldsymbol{\mathcal{X}} - diagonals(\boldsymbol{\mathcal{X}})$ by removing the cases where $s = d$ from $\boldsymbol{\mathcal{X}}$, since this behavior is captured by the Fixation model (next subsection). Our goal with the Switch model is to estimate trajectories as an interpretable probability space. That is, our goal is to decompose $\boldsymbol{\mathcal{X}}$ in a probability matrix $\mathbf{P}$, where each cell in this matrix captures the probability of a user switching attention for s to d (or $p(d|s)$).

A naïve way to define $\mathbf{P}$ is simply to define $p(d|s) \propto n_{ds\cdot}$. That is, to use maximum likelihood estimates [11].

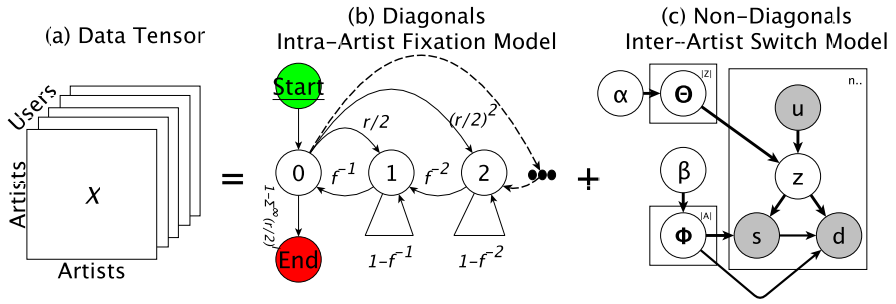


Figure 1. The SWIFT-FLOWS model: Data representation by tensor $\mathcal{X}$ (left), the repeated consumption model (center) and the inter-artist graphical model (right).

However, this approach has three undesirable properties [19]: (1) there are not enough samples to accurately estimate the transition probabilities for most artists; (2) the transition probability matrix $\mathbf{P}$ is sparse, stating that it is impossible to transition from an artist s to d when no user has done so in the past; and, (3) it does not take into account user preferences. For example, if we observe that the transition Sepultura→Beyonce is very common for a single or small group of users, this does not imply that it is frequent for all of the listeners in the OMSSs.

In order to deal with such issues, we employ the Bayesian model depicted in Figure 1-c. With this model, our goal is capture the latent spaces of inter-artist transition patterns shared by a group of users. We call this the Switch model. The latent space $\mathcal{Z}$ defines a set of *transitions* between pairs of artists s and d. We refer to each latent factor z as an *attention transition gene*, and the collection of genes as a *genome*. These terms are inspired by the "Music Genome Project", a proprietary approach developed by Pandora that aims to describe in detail the musical characteristics of individual songs [3].

Estimating the Model: Let $k = |\mathcal{Z}|$ ($k << |\mathcal{A}|$) be an input variable determining the number of genes (or latent factors) to be estimated. Later, we shall describe our approach to define k. The two other inputs are the hyper-parameters α and β. The outputs of the Switch model are two matrices, $\mathbf{\Theta}$ and $\mathbf{\Phi}$, as well as a vector $\mathbf{z}$. $\mathbf{\Theta}$ has $|\mathcal{U}|$ rows and $|\mathcal{Z}|$ columns, where each cell contains the probability that a user has a preference towards a given gene:

$$p(z|u) = \mathbf{\Theta}(u,z) = \theta_{z|u}(z) = \frac{n_{zu} + \alpha}{n_{\cdot u} + |\mathcal{Z}|\alpha} \quad (3)$$

where n_{zu} is estimated by the model. Matrix $\mathbf{\Phi}$ has $|\mathcal{Z}|$ rows and $|\mathcal{A}|$ columns. It captures the probability that when a user is interest in gene z it will transition to a, i.e.:

$$p(a|z) = \mathbf{\Phi}(z,a) = \phi_{a|z}(a) = \frac{n_{az} + \beta}{n_{\cdot z} + |\mathcal{A}|\beta} \quad (4)$$

where, once again, n_{az} is estimated from the data by the model. Finally, vector $\mathbf{z}$ contains the probabilities of each gene $z \in \mathcal{Z}$, referred to as $p(z)$, that is: $p(z) \propto n_z$. Finally, the decomposed transition matrix $\mathbf{P}$ is defined by:

$$\mathbf{P}(s,d) = \sum_{z \in |\mathcal{Z}|} p(z|s)p(d|z) \quad (5)$$

where $p(d,s|z) = p(s|z)p(d|z)$, and $p(z|s) \propto p(s|z)p(z)$.

Gibbs Sampling: We use a collapsed Gibbs sampler [8] to estimate matrices $\mathbf{\Theta}$ and $\mathbf{\Phi}$ by estimating n_{zu} and n_{az}, as well as vector $\mathbf{z}$. We sample from the posterior defined by the product $\theta_{z|u}\phi_{s|z}\phi_{d|z}$ [7]. We fix hyper-parameters $\alpha = \frac{50}{|\mathcal{Z}|}$, and $\beta_s = \beta_d = 0.001$, as is usually done with similar models [7, 13]. We execute the sampler for 800 iterations with 300 being discarded as burn-in.

Estimating k: We apply the minimum description length (MDL) principle [9], which is largely used for problems of model selection, to determine the number of genes $k = |\mathcal{Z}|$. With MDL, we fine tune SWIFT-FLOWS in order to extract a succinct, yet still accurate, representation of the listening habits of users. MDL captures how good a model $\mathcal{M}$ ($\mathbf{P}$ in our case) represents the data by taking into account the trade-off between the "goodness" (or likelihood) and the complexity (or generality) of the model.

To apply MDL we first define the likelihood of the data given the model $\mathcal{M}$. Given $n_{ds} = n_{\cdot ds}$ the number of transitions from s to d by all users, the log likelihood of matrix $\mathbf{P}$ is given by $\sum_{s,d|s\neq d} n_{ds} log(p(d|s))$ [4]. The MDL cost of model $\mathcal{M}$ is given by the sum:

$$Cost(\mathbf{P}, \mathcal{M}) = Cost(\mathbf{P} \mid \mathcal{M}) + Cost(\mathcal{M}). \quad (6)$$

$Cost(\mathbf{P} \mid \mathcal{M})$, defined as the negative log-likelihood, captures the goodness-of-fit of the data given the model: higher-values imply on accurate but yet succinct (less factors) recoveries of $\mathbf{P}$. $Cost(\mathcal{M})$ captures the complexity:

$$Cost(\mathcal{M}) = log^*(|\mathcal{A}|) + log^*(|\mathcal{Z}|) + \sum_{s,d,z} [log^*(\lceil p(d|z)n_{\cdot\cdot} \rceil) + log^*(\lceil p(s|z)n_{\cdot\cdot} \rceil) + log^*(\lceil p(z)n_{\cdot\cdot} \rceil)]$$

where $log*$ is the universal coding cost (number of bits) for integers [9]. $Cost(\mathcal{M})$ represents the encoding each matrix in the model in integer representation with precision $n_{\cdot\cdot}$ (the total number of transitions) [5].

[3] http://www.pandora.com/about/mgp

[4] The likelihood is the product of $p(d|s)$ for all n_{ds} transitions [5].

[5] Since we deal with counts, the smallest probability value is $(1/n_{\cdot\cdot})$.

3.2 Fixation Model

Users' bursty repeated consumption of artists requires modeling this behavior with a stochastic process that has memory. Markov modulated processes are a class of models that are particularly versatile for this task [22]. Our goal here is not only to model user behavior but also, through the use of intuitive parameters, understand how users repeatedly consume artists. Most importantly, we want to reproduce user attention giving rise to both exponential and power law distributions observed in our datasets.

Our fixation model, which captures the intra-artist transitions, is a Markov modulated process where we use an infinite number of states, an approach widely used to model systems with bursty behavior [22]. Figure 1 (b) illustrates our model (only the initial states). The "start" circle represents the initial transition from the Inter-artist model. From state zero we are interested in how long it takes to exit from the "exit" transition. Thus, circles "start" and "exit" in Figure 1 (b) are not states but rather entrance (exit) transitions from (to) the inter-artist model. The states of the model capture the affinity of the user for the artist, that is how much the user is willing to repeatedly listen the artist's songs. There is a fixed residency time Δt on each state. Thus, higher states represent that the user has a higher affinity and thus dedicates more play-time to the artist.

The model has parameters $0<r<1$ and $1\leq f<4r$. The limit of $4r$ is required as described in [22] [6]. Parameter r models the user "rush", capturing how users are led to hear more from an artist (e.g. entire album) after hearing some music by this artist. Parameter f models user "fixation", representing how long it takes for users to get over an initial impulse to listen to an artist, which is also a function of the artist's song inventory size. A large value of f implies that users quickly get over their initial impulse or happens because the artist has just a few songs.

We can fit the Fixation model to the complementary cumulative distribution function (CCDF) of the time users dedicate to an artist using the Levenberg-Marquardt algorithm. The CCDF will define the probability of the residency time in the chain. The infinite number of states can be captured by using a sufficient number of states (100 in our datasets). We evaluate the algorithm on the mean squared error of the real data and the residency times generated by the model. As we shall show empirically in our results, this model is capable of generating both power-law and exponential residency times as also discussed in [22].

One interesting property of the Fixation model is that it is able to estimate the expected amount time users will fixate on a given artist. To achieve this, we can compute the expected number of steps that it takes to go from the Start state to the End state [11] [7]. If we define this value per artist as e_a, we can couple the Switch model with the Fixation model by estimating the expected fixation steps per latent spaces, e_z, as:

$$e_z = \sum_{a \in |\mathcal{A}|} p(a|z) e_a \tag{7}$$

That is, e_a is the expected number of steps a user will remain in artist a with regards to his/her interest in gene z.

In the next section we describe SWIFT-FLOWS at work.

4. SWIFT-FLOWS AT WORK

We apply SWIFT-FLOWS on datasets crawled from Last.FM. Last.FM aggregates various forms of digital music consumption, ranging from desktop/mobile media players to streaming services (theirs and others) [8]. Last.FM is also an online social network (OSN), allowing the creation of user groups as well as providing demographical data. The datasets we explore are:

Last.FM-2009 Collected in using a snowball sampling [2]. After the snowball sampling, 992 uniformly random users were selected. The dataset contains, for each user, the complete listening history (all plays) from February 2005 to May 2009, the self-declared nationality, age (at the time), and registration date [2]. This dataset accounts for 18.5 million user, artist, and timestamp triples, as well as 107,397 unique artists.

Last.FM-2014 Crawled in 2014 by identifying users that participate in discussion groups on Last.FM. Contains the listening history (from February 2005 to August 2014) of a subset of the users that discuss pop-artists on Last.FM discussion groups. The total number of users in this dataset is 15,329. Also, this dataset contains 836,625 unique artists and roughly 218 million user, artist, and timestamp triples. As is the case with Last.FM-2009, this dataset provides the age and nationality of all the users.

Because of these various means of consumption [16], Last.FM presents itself as an interesting platform for studying *online behavior*. The service aggregates user accesses from desktop media players (that incorporates legal and illegal downloads), free, and also paid streaming services. Nevertheless, it is important to point out that the observed attention trajectories will be impacted by how the data was gathered (e.g., Last.FM-2014 has a bias towards pop artists), as well as the internal mechanisms of the OMSSs (e.g., such as recommendation services and user interfaces). As we shall discuss in our results, regardless of the data biases, SWIFT-FLOWS is able to represent the attention trajectories of under-represented user populations.

We run the inter-artist attention Switch model of SWIFT-FLOWS on $\boldsymbol{\mathcal{X}}^{-}$, and the Fixation model of SWIFT-FLOWS on the intra-artist transitions. In both cases, only artists which had at least five plays by five users are considered. In total, the Last.FM-2014 dataset has over 3M plays of such artists, while Last.FM-2009 has roughly

[6] The authors write the model in terms of $a = 2/r$ and $b = f/a$.

[7] https://en.wikipedia.org/wiki/Absorbing_Markov_chai

[8] Aggregation is done using plugins available on other OMSSs and media players.

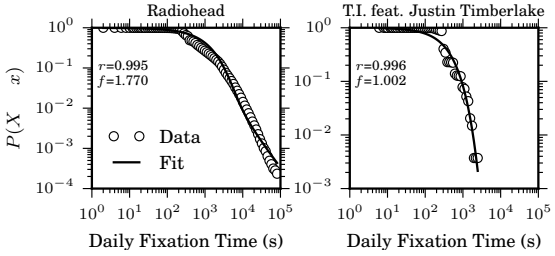


Figure 2. Validation of Fixation Model.

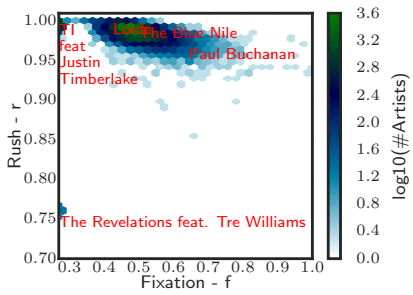


Figure 3. Rush vs Fixation

176k plays. We note that even after filtering, there still remains a significant number of rare transitions as 44% of the inter-artist transitions happen less than ten times.

4.1 The Fixation Model at Work

We first discuss the Fixation model. We begin by showing how it fits the time users spend listening to different artists on any given day (referred to as daily fixation time). Figure 2 shows the fitted and empirical complementary cumulative distribution functions (CCDF) of the daily fixation time for two particular example artists, namely *Radiohead*, and *T.I. feat. Justin Timberlake* (a collaboration between two artists). This example was extracted from the Last.FM-2014 dataset.

The distribution for *Radiohead* clearly has long tails, and is similar to the distributions for most artists. In contrast, the distribution for the *T.I. feat. Justin Timberlake* collaboration has a much shorter tail, approaching an exponential distribution. Unlike for the other artists, there is only one song by this artist collaboration in our dataset, which might explain why users tend to spend less time listening to them. Yet, our Fixation model provides close fittings for both distributions, capturing both long and short tails. Interestingly, we can also use the model parameters r and f to distinguish between these artists: compared to *Radiohead*, the *T.I. feat. Justin Timberlake* collaboration has a slightly higher rush parameter ($r = 0.996$) but a much lower fixation parameter ($f = 1.002$). Despite the higher initial surge of attention, users lose interest more quickly in them. If it were not for our separation of the intra-artist from the inter-artist transitions, it would be impossible to capture these different distributions with SWIFT-FLOWS. That is, these superior fits are only possible through the use the modulated Markov models as done by our intra-artist model. This allows the model to capture both long and short tails of user attention [22].

We proceeded to fit our model to the daily fixation times of 36,344 and 2,570 artists in the Last.FM-2014 and Last.FM-2009 datasets respectively (artists with more than 5 plays by at least 5 users). In Figure 3 we show a scatter plot of the fixation versus rush scores for the Last.FM-2014 dataset. We found that, the vast majority of the artists have very high values of rush r (above 0.95) and values of fixation f (1.5 to 2.5). There were also two other small groups of artists with very low (near 1) fixation.

Looking into these groups, we found many collaborations between artists, such as the aforementioned TI feat Justin Timberlake. Another example of a collaboration is The Revelations feat Tre Williams, it has both low fixation and low rush scores, thus attracts little attention in our datasets. We also found that Blue Nile is an interesting example. Blue Nile is a Scottish alternative/pop band whose lead singer is Paul Buchanan. From the plot, we can see that the band has higher rush and lower fixation than the solo songs by Paul Buchanan. This is likely because the solo career of Paul Buchanan mostly attracts more interested fans. Another example is Lorde, a relatively new pop singer at time (2014). The artist obtains high rush and somewhat lower fixation. This may be explained listeners discovering her music.

Our fitting errors are very small in most cases. The average Mean Squared Errors (MSE) of each fitted distribution for artists in Last.FM-2014 is only of 0.02, whereas in the Last.FM-2009 dataset it was of 0.03. The standard deviations were of 0.02 and 0.04 for the Last.FM-2014 and Last.FM-2009 datasets, respectively.

4.2 Extracting Listening Trajectories

We now discuss the Switch model. The first step to execute the model us to decide the number of genes (latent factors) k using the MDL-based criteria described previously. To measure the MDL score, we searched for k in the range $k \in [2, 400]$ [9]. With MDL, we aim at finding a succinct (smaller) yet accurate latent representation of our datasets. In our search, we found that in both datasets as k increases the MDL cost first decreases and then rapidly increases, reaching global minimum at k=40. This value was achieved in both sets of data. For this reason, our experiments use a genome with 40 genes.

Table 1 describes four different genes (latent factors) extracted by SWIFT-FLOWS from the Last.FM-2014 dataset. For each gene, the table shows the top 7 source s and destination d artists in a single column ranked by $(p(a \mid z))$. To further examine the genes, the table also summarizes the nationality and age reported in the LastFM profile of the top 50 users which have attention transitions within each gene. Finally, we cross-referenced the top artists in each gene with the AllMusic guide [10] for an authoritative source on artist metadata. The labels given to each gene stem from our own interpretation.

[9] We searched $k \in \{2, 4, 8, 10, 20, 30, 40, 50, 100, 200, 300, 400\}$.
[10] `http://www.allmusic.com/`

Table 1. Genes from the Last.FM-2014 dataset: top source and destination artists, and demographics of top-50 users. Artists and users are sorted by probabilities $p(a|z)$ and $p(z|u)$, respectively. Countries are: BR = Brazil, US = USA, NL = Netherlands, DE = Germany, PL = Poland, FI = Finland. Age statistics presented here are the 1^{st}, 2^{nd} and 3^{rd} quartiles. We also show the expected fixation e_z per gene.

	Gene=18 ("BR/US pop")	Gene=20 ("metal")	Gene=23 ("electronic")	Gene=39 ("pop'")
Source/Dest Artists	Britney Spears Wanessa Christina Aguilera t.A.T.u. Katy Perry Pitty Lady Gaga	Nightwish Within Temptation Epica Korn Disturbed Marilyn Manson Rammstein	Daft Punk David Guetta Deadmau5 Skrillex The Prodigy Tiesto Pendulum	Britney Spears Madonna Christina Aguilera Rihanna Lady Gaga Katy Perry Kesha
Users Nationality	BR=98% NL=2%	DE = 18% PL = 16% US = 12% FI = 8%	US = 18% BR = 10% PL = 10% UK = 10%	BR=78% US=10% PL=5%
Age Quartiles	1^{st} = 19 2^{nd} = 21 3^{rd} = 24	1^{st} = 21 2^{nd} = 24 3^{rd} = 29	1^{st} = 20 2^{nd} = 22 3^{rd} = 25	1^{st} = 19 2^{nd} = 22 3^{rd} = 25
e_z	e_z = 793.55	e_z = 642.15	e_z = 636.10	e_z = 886.10

Overall, the genes found through SWIFT-FLOWS point to a semantically sound segmentation of transition spaces that combines characteristics of the artists and users of the OMSS. Illustratively, gene $z = 18$ is predominately formed by female pop/rock singers as both sources and destinations. This is not the only gene with similar pop singers; gene $z = 39$ is another gene with a similar composition in this respect. Yet, the presence of Brazilian pop artists (e.g., *Wanessa*, *Claudia Leitte*, and *Pitty*) in gene $z = 18$ explains why the vast majority (98%) of the top users in this gene are Brazilians (BR). Gene $z = 20$ in turn is mostly focused on different sub-genres of metal (e.g., goth-metal and rap-metal). A large fraction of the top-50 users of the "heavy metal" gene are from Germany and Poland. Finally, gene $z = 23$ represents users of different nationalities (American being the most frequent one) who like to listen to electronic dance music, often transitioning between different artists of that genre. It is also noteworthy that in a dataset mostly comprised of pop artists fans (Last.FM-2014), SWIFT-FLOWS is able to account for the trajectories of heavy metal and electronic music fans.

To understand the expected fixation of users per gene, we make use of Equation 7 (e_z). Initially translate e_z values to seconds. That is, we performed a linear regression using the values of e_a (see Eq. 7), expected number of steps per artist, with the average fixation time per day (described in the previous subsection). With this regression, we found that each step in the chain accounts for, approximately, 1.11 seconds. From the table, we can see that genes 20 and 23 have lower expected fixation times. That is, gene $z = 20$ expects 642 steps (roughly 12 minutes) in the Fixation model, whereas gene $z = 23$ expects 632 steps (11.6 minutes). The highest value in the table is from gene $z = 18$ (14 minutes).

Notice that both models combined provide a general overview of attention. That is, we are able to understand how users will transition between artists, as well as the expected number of steps users will listen to a given artist. This represents one of the major strengths of SWIFT-FLOWS when compared to previous efforts [7, 22].

5. CONCLUSIONS

In this paper, we presented the SWIFT-FLOWS model. One of the main advantages of SWIFT-FLOWS is that it allows researchers to explore user listening habits based on complementary behaviors: the fixation on a single artist over short or long bursts, as well as the change in attention from one artist to the next. We applied SWIFT-FLOWS to uncover semantically meaningful maps of attention flows in large OMSSs datasets. Moreover, SWIFT-FLOWS provides excellent fits to the attention time dedicated to artists. SWIFT-FLOWS, therefore, is an useful tool for further research aiming to understand listening behavior.

Acknowledgments: Research supported by the project FAPEMIG-PRONEX-MASWeb, process number APQ-01400-14, by grant 460154/2014-1 from CNPq, by the EU-BR BigSea project (MCTI/RNP 3rd Coordinated Call), and by the EU-BrazilCC (grants FP7 614048 and CNPq 490115/2013-6) project, as well as by individual grants from CNPq/CAPES/Fapemig. Our work was also supported by NSF grant CNS-1065133, U.S. Army Research Laboratory under Cooperative Agreement W911NF-09-2-0053, and by the Defense Threat Reduction Agency contract No. HDTRA1-10-1-0120. The views and conclusions contained here are our own (the authors).

6. REFERENCES

[1] Alejandro Bellogín, Arjen P. de Vries, and Jiyin He. Artist Popularity: Do Web and Social Music Services Agree? In *Proc. ICWSM*, 2013.

[2] Oscar Celma. *Music Recommendation and Discovery in the Long Tail*. Springer, 1 edition, 2010.

[3] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *Proc. KDD*, 2012.

[4] Shuo Chen, Jiexun Xu, and Thorsten Joachims. Multi-space probabilistic sequence modeling. In *Proc. KDD*, 2013.

[5] Imre Csiszár and Paul C. Shields. The consistency of the bic markov order estimator. *The Annals of Statistics*, 28(6):1601–1619, 12 2000.

[6] Flavio Figueiredo, Jussara M Almeida, Yasuko Matsubara, Bruno Ribeiro, and Christos Faloutsos. Revisit Behavior in Social Media: The Phoenix-R Model and Discoveries. In *PKDD/ECML*, 2014.

[7] Flavio Figueiredo, Bruno Ribeiro, Jussara M Almeida, and Christos Faloutsos. TribeFlow: Mining & Predicting User Trajectories. In *Proc. WWW*, 2016.

[8] Tom Griffiths. Gibbs sampling in the generative model of latent dirichlet allocation. Technical report, 2002.

[9] Mark H Hansen and Bin Yu. Model Selection and the Principle of Minimum Description Length, 2001.

[10] David J Hargreaves. Musical imagination: perception and production, beauty and creativity. *Psychology of music*, 40(5):539–557, 2012.

[11] David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times*. AMS, 2008.

[12] Andryw Marques, Nazareno Andrade, and Leandro Balby Marinho. Exploring the Relation Between Novelty Aspects and Preferences in Music Listening. In *Proc. ISMIR*, 2013.

[13] Yasuko Matsubara, Yasushi Sakurai, Christos Faloutsos, Tomoharu Iwata, and Masatoshi Yoshikawa. Fast mining and forecasting of complex time-stamped events. In *Proc. KDD*, 2012.

[14] Carl D. Meyer. Stochastic Complementation, Uncoupling Markov Chains, and the Theory of Nearly Reducible Systems. *SIAM Review*, 31(2):240–272, 1989.

[15] Joshua L. Moore, Shuo Chen, Thorsten Joachims, and Douglas Turnbull. Taste Over Time: The Temporal Dynamics of User Preferences. In *Proc. ISMIR*, 2013.

[16] Raphaël Nowak. Investigating the interactions between individuals and music technologies within contemporary modes of music consumption. *First Monday*, 19(10):Online, 2014.

[17] Chan Ho Park and Minsuk Kahng. Temporal Dynamics in Music Listening Behavior: A Case Study of Online Music Service. In *Proc. ACIS*, 2010.

[18] Minsu Park, Ingmar Weber, Mor Naaman, and Sarah Vieweg. Understanding Musical Diversity via Online Social Media. In *Proc. ICWSM*, 2015.

[19] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized Markov chains for next-basket recommendation. *Proc. WWW*, 2010.

[20] P. J. Rentfrow and S. D. Gosling. The Do Re Mi's of Everyday Life: The Structure and Personality Correlates of Music Preferences. *Journal of personality and social psychology*, 84(6), 2003.

[21] Peter J. Rentfrow, Lewis R. Goldberg, and Daniel J. Levitin. The structure of musical preferences: a five-factor model. *Journal of personality and social psychology*, 100(6):1139–1157, June 2011.

[22] Stephan Robert and Jean-Yves Le Boudec. On a Markov modulated chain exhibiting self-similarities over finite timescale. *Performance Evaluation*, 27-28:159–173, 1996.

[23] Suvi Saarikallio and Jaakko Erkkilä. The role of music in adolescents' mood regulation. *Psychology of music*, 35(1):88–109, 2007.

[24] Philipp Singer, Denis Helic, Andreas Hotho, and Markus Strohmaier. HypTrails: a bayesian approach for comparing hypotheses about human trails on the web. In *Proc. WWW*, 2015.

[25] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G. Carbonel. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proc. SDM*, 2010.

MUSICAL TYPICALITY: HOW MANY SIMILAR SONGS EXIST?

Tomoyasu Nakano[1] **Daichi Mochihashi**[2] **Kazuyoshi Yoshii**[3] **Masataka Goto**[1]
[1] National Institute of Advanced Industrial Science and Technology (AIST), Japan
[2] The Institute of Statistical Mathematics, Japan
[3] Kyoto University, Japan
[1] {t.nakano, m.goto}[at]aist.go.jp [2] daichi[at]ism.ac.jp
[3] yoshii[at]kuis.kyoto-u.ac.jp

ABSTRACT

We propose a method for estimating the musical "typicality" of a song from an information theoretic perspective. While musical similarity compares just two songs, musical typicality quantifies how many of the songs in a set are similar. It can be used not only to express the uniqueness of a song but also to recommend one that is representative of a set. Building on the type theory in information theory (Cover & Thomas 2006), we use a Bayesian generative model of musical features and compute the typicality of a song as the sum of the probabilities of the songs that share the type of the given song. To evaluate estimated results, we focused on vocal timbre which can be evaluated quantitatively by using the singer's gender. Estimated typicality is evaluated against the Pearson correlation coefficient between the computed typicality and the ratio of the number of male singers to female singers of a song-set. Our result shows that the proposed measure works more effectively to estimate musical typicality than the previous model based simply on generative probabilities.

1 INTRODUCTION

The amount of digital content that can be accessed has been increasing and will continue to do so in the future. This is desirable with regard to the diversity of the content, but is making it harder for listeners to select from this content. Furthermore, since the amount of similar content is also increasing, creators will be more concerned with the originality of their creations. All kinds of works are influenced by some existing content, and it is difficult to avoid an unconscious creation of content partly similar in some way to other content.

This paper focuses on *musical typicality* which reflects the number of songs having high similarity with the target song as shown in Figure 1. This definition of musical typicality is based on *central tendency*, which in cognitive psychology is one of the determinants of typicality [2]. Musical typicality can be used to recommend a unique or representative song for a set of songs such as those in a particular genre or personal collection, those on a specific playlist, or those released in a given year or a decade. And it can help listeners to understand the relationship between a song and such a song set. However, human ability with regard to typicality is limited. Judging similarity between two songs is a relatively simple task but is time-consuming, so judging the similarities of a million songs is impossible. Consequently, despite the coming of an era in which people other than professional creators can enjoy creating and sharing works, the monotonic increase in content means that there is a growing risk that one's work will be denounced as being similar to someone else's. This could make it difficult for people to freely create and share content. The musical typicality proposed in this paper can help create an environment in which specialists and general users alike can know the answers to the questions "How often does this occur?" and "How many similar songs are there?".

Much previous work has focused on *musical similarity* because it is a central concept of MIR and is also important for purposes other than retrieval. For example, the use

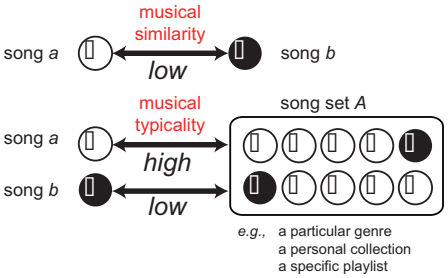


Figure 1. Musical similarity and typicality.

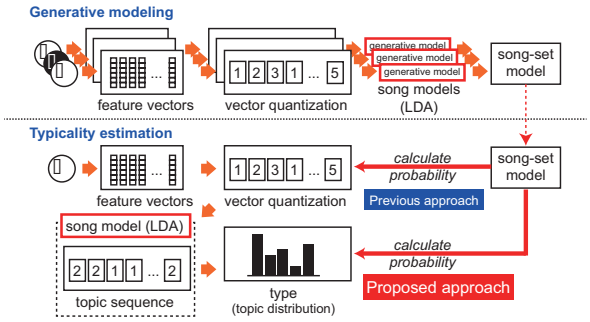


Figure 2. Estimation of music typicality represented by a discrete sequence based on the type theory. Both the previous and the proposed approach are illustrated.

© Tomoyasu Nakano, Daichi Mochihashi, Kazuyoshi Yoshii, Masataka Goto. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Tomoyasu Nakano, Daichi Mochihashi, Kazuyoshi Yoshii, Masataka Goto. "Musical Typicality: How Many Similar Songs Exist?", 17th International Society for Music Information Retrieval Conference, 2016.

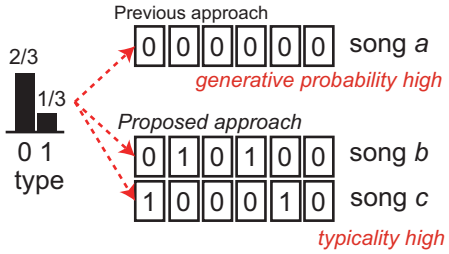


Figure 3. Examples of a song having high generative probability and songs having high typicality.

of similarity to automatically classify musical pieces (into genres, music styles, etc.) has been studied [10, 13], as has its use for music auto-tagging [17]. Each of these applications, however, is different from musical typicality: musical similarity is usually defined by comparing two songs, *music classification* is defined by classifying a given song into one out of a set of categories (category models, centroids, etc.), and *music auto-tagging* is defined by comparing a given song to a set of tags (tag models, the closest neighbors, etc.).

Nakano *et al.* proposed a method for estimating musical typicality by using a generative model trained from the song set (Figure 2) [16] and showed its application to visualizing relationships between songs in a playlist [14]. Their method estimates acoustic features of the target song at each frame and represents the typicality of the target song represented as an average probability of each frame of the song calculated using the song-set model. However, we posit that the generative probability is not truely appropriate to represent typicality.

The method we propose here, in contrast, introduces the *type* from information theory for improving estimated musical typicality by a bag-of-features approach [16]. In this context, the type is same meaning with the unigram distribution. We first model musical features of songs by using a vector quantization method and latent Dirichlet allocation (LDA) [4]. We then estimate a song-set model from the song models. Finally, we compute the typicality of the target song by calculating the probability of a type of the musical sequence (quantized acoustic features) calculated using the song-set model (Figure 2).

2 METHOD

The key concept of the method in this paper is the *type* of a sequence on which we consider the typicality of a given music. Previous work have mentioned/used simple generative probabilities to compute musical similarity [1] or typicality [16] of a music and for singer identification [8]. However, simple generative probability will not conform to our notion of typicality. Imagine the simplest example in Figure 3: here, each song consists of alphabets of $\{0, 1\}$ and the stationary information source has a probability distribution on alphabets $Q(0) = 2/3,\ Q(1) = 1/3$.

Clearly, while the song "a" has the highest probability of generation, we can see that the sequences like "b" and "c" will occur more typically. This means that we should think about the *sum* of the probabilities of songs that are similar to the song to measure the typicality.

2.1 Type and the Typicality

Let us formalize our ideas from the viewpoint of information theory [5–7]. Let $\mathbf{x} = \{x_1, x_2, \cdots, x_n\}$ be a sequence of length n whose alphabet x comes from a set $\mathcal{X}$. We assume that $\mathbf{x}$ comes from a stationary memoryless information source, i.e. we can drop the order of symbols in $\mathbf{x}$ and regard $\mathbf{x}$ as a bag of words. Next, we introduce some definitions:

Definition 1 (type). Let $N(x|\mathbf{x})$ be the number of times that $x \in \mathcal{X}$ appeared in sequence $\mathbf{x}$. The *type* $P_{\mathbf{x}}$ of the sequence $\mathbf{x}$ is an empirical probability distribution of symbols in $\mathbf{x}$:

$$P_{\mathbf{x}} = \left\{ \frac{1}{n} N(x|\mathbf{x}) \,\middle|\, x \in \mathcal{X} \right\}. \tag{1}$$

We denote the space of all $P_{\mathbf{x}}$ as $\mathcal{P}_n$.

Definition 2. Let $P \in \mathcal{P}_n$. A set of sequences of length n that share the same type P is called a *type class* T^n of P:

$$T^n(P) = \{\mathbf{x} \in \mathcal{X}^n \mid P_{\mathbf{x}} = P\}. \tag{2}$$

Now let us denote the probability of a sequence $\mathbf{x}$ from an memoryless information whose symbol probabilities are $Q(x)$:

$$p(\mathbf{x}) = Q^n(\mathbf{x}) = \prod_{i=1}^{n} Q(x_i). \tag{3}$$

Given these definitions, the following simple theorems follow:

Theorem 1. The probability of a sequence $\mathbf{x}$ having type P from a stationary memoryless information source Q is expressed as follows:

$$Q^n(\mathbf{x}) = \exp\big[-n(H(P) + D(P||Q))\big] \tag{4}$$

Here, $H(P)$ and $D(P||Q)$ are an entropy of P and Kullback-Leibler divergence of P from Q, respectively.

$$H(P) = -\sum_{x \in \mathcal{X}} P(x) \log P(x) \tag{5}$$

$$D(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)} \tag{6}$$

Proof.

$$\begin{aligned} Q^n(\mathbf{x}) &= \prod_{i=1}^{n} Q(x_i) = \prod_x Q(x)^{N(x|\mathbf{x})} = \prod_x Q(x)^{nP(x)} \\ &= \prod_x \exp\big[nP(x) \log Q(x)\big] \qquad (7) \\ &= \exp\Big[-n\Big(-\sum_x P(x) \log Q(x)\Big)\Big] \qquad (8) \\ &= \exp\Big[-n\Big(H(P) + D(P||Q)\Big)\Big]. \quad \square \qquad (9) \end{aligned}$$

Theorem 2 (lower and upper bounds)**.** For any type $P \in \mathcal{P}_n$,

$$\frac{1}{(n+1)^{|\mathcal{X}|-1}} \exp\{nH(P)\} \leq |T^n(P)| \leq \exp\{nH(P)\}. \tag{10}$$

Using the theorems above, the following important theorem can be proved.

Theorem 3. For any type $P \in \mathcal{P}_n$ and any probability distribution Q,

$$Q^n(T^n(P)) \doteq \exp\{-nD(P||Q)\}, \tag{11}$$

where $a_n \doteq b_n$ if $\lim_{n\to\infty}(1/n)\log(a_n/b_n) = 0$.

Proof. Using (4) and (10),

$$\begin{aligned} Q^n(T^n(P)) &= \textstyle\sum_{\mathbf{x}\in T^n(P)} Q^n(\mathbf{x}) \\ &= |T^n(P)| \exp(-n(H(P) + D(P||Q))) \\ &\doteq \exp(nH(P)) \cdot \exp(-n(H(P) + D(P||Q))) \\ &= \exp\{-nD(P||Q)\}\,. \quad \square \end{aligned} \tag{12}$$

This theorem says that the *sum* of the probabilities of sequences that share the same type P is given by an exponential of Kullback-Leibler divergence from the information source Q. While the equation (11) is usually used in information theory to formalize that such a probability exponentially decays with the length n, here we do not care for n but for the form of the function. Thus, we normalize (11) for a unit observation like the well-known *perplexity*, yielding the definition of typicality as follows:

Definition 3 (Typicality).

$$\text{Typicality}(P|Q) = \exp\{-D(P||Q)\} \tag{13}$$

where P is the type of a musical sequence and Q is a generative model of its musical features.

2.2 Generative modeling and Type

To evaluate the typicality estimation method, we compute the type of each song by modeling them in a way based on our previous work [16]. From polyphonic musical audio signals including a singing voice and sounds of various musical instruments, we first extract vocal timbre. We then model the timbre of songs by using a vector-quantization method and latent Dirichlet allocation (LDA) [4]. Finally, a song-set model Q is estimated by integrating all song models (Figure 2).

In addition, we use the expectation of Dirichlet topic distribution as a type P because the hyperparameter of the posterior Dirichlet distribution can be interpreted as the number of observations of the corresponding topic. In the other words, the P indicates mixing weights of the multiple topics.

2.2.1 Extracting acoustic features: Vocal timbre

We use the mel-frequency cepstral coefficients of the LPC spectrum of the vocal (LPMCCs) and the ΔF_0 of the vocal to represent vocal timbre because they are effective for identifying singers [8, 15]. In particular, the LPMCCs represent the characteristics of the singing voice well, since singer identification accuracy is greater when using LPMCCs than when using the standard mel-frequency cepstral coefficients (MFCCs) [8].

We first use Goto's PreFEst [11] to estimate the F_0 of the predominant melody from an audio signal and then the F_0 is used to estimate the ΔF_0 and the LPMCCs of the vocal. To estimate the LPMCCs, the vocal sound is resynthesized by using a sinusoidal model based on the estimated vocal F_0 and the harmonic structure estimated from the audio signal. At each frame the ΔF_0 and the LPMCCs are combined as a feature vector.

Then *reliable frames* (frames little influenced by accompaniment sound) are selected by using a vocal GMM and a non-vocal GMM (see [8] for details). Feature vectors of only the reliable frames are used in the following processes (model training and probability estimation).

2.2.2 Quantization

Vector quantization is applied using the k-means algorithm to convert acoustic feature vectors of each element to a symbolic time series representation. In that algorithm the vectors are normalized by subtracting the mean and dividing by the standard deviation, and then the normalized vectors are quantized by prototype vectors (centroids) trained previously. Hereafter, we call the quantized symbolic time series *acoustic words*.

2.2.3 Probabilistic generative model: song model

The observed data we consider for LDA are D independent songs $\boldsymbol{X} = \{\boldsymbol{X}_1, ..., \boldsymbol{X}_D\}$. A song $\boldsymbol{X}_d$ is N_d acoustic words $\boldsymbol{X}_d = \{\boldsymbol{x}_{d,1}, ..., \boldsymbol{x}_{d,N_d}\}$. The size of the acoustic words vocabulary equals to the number of clusters of the k-means algorithm, V. We consider a K-dimensional multinomial of latent topic proportions θ_d for each $\boldsymbol{X}_d$, and write $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_D)$.

Introducing latent topic assignments $\boldsymbol{Z}_d = \{\boldsymbol{z}_{d,1}, ..., \boldsymbol{z}_{d,N_d}\}$ for $\boldsymbol{X}_d$ and collectively write $\boldsymbol{Z} = \{\boldsymbol{Z}_1, ..., \boldsymbol{Z}_D\}$, the full joint distribution of our LDA model is given by

$$p(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\theta}, \boldsymbol{\phi}) = p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\phi})p(\boldsymbol{Z}|\boldsymbol{\theta})p(\boldsymbol{\theta})p(\boldsymbol{\phi}) \tag{14}$$

where $\boldsymbol{\phi}$ indicates the emission distribution of each topic. The first two terms are likelihood functions, and the other two are prior distributions. The likelihood functions are defined as

$$p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\phi}) = \prod_{d=1}^{D}\prod_{n=1}^{N_d}\prod_{v=1}^{V}\left(\prod_{k=1}^{K}\phi_{k,v}^{z_{d,n,k}}\right)^{x_{d,n,v}} \tag{15}$$

and

$$p(\boldsymbol{Z}|\boldsymbol{\theta}) = \prod_{d=1}^{D}\prod_{n=1}^{N_d}\prod_{v=1}^{V}\theta_{d,k}^{z_{d,n,k}}. \tag{16}$$

We endow $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ conjugate Dirichlet priors:

$$p(\boldsymbol{\theta}) = \prod_{d=1}^{D}\text{Dir}(\theta_d|\alpha_0) \propto \prod_{d=1}^{D}\prod_{k=1}^{K}\theta_{d,k}^{\alpha_0-1} \tag{17}$$

$$p(\boldsymbol{\phi}) = \prod_{k=1}^{K}\text{Dir}(\boldsymbol{\phi}_k|\beta_0) \propto \prod_{k=1}^{K}\prod_{v=1}^{V}\phi_{k,v}^{\beta_0-1}\,. \tag{18}$$

where p($\boldsymbol{\theta}$) and p($\boldsymbol{\phi}$) are products of Dirichlet distributions and α_0, β_0 are their prior hyperparameters.

Finally, we obtain a type of each song $\boldsymbol{X}_d$ as an expectation of the Dirichlet posterior distribution of θ_d.

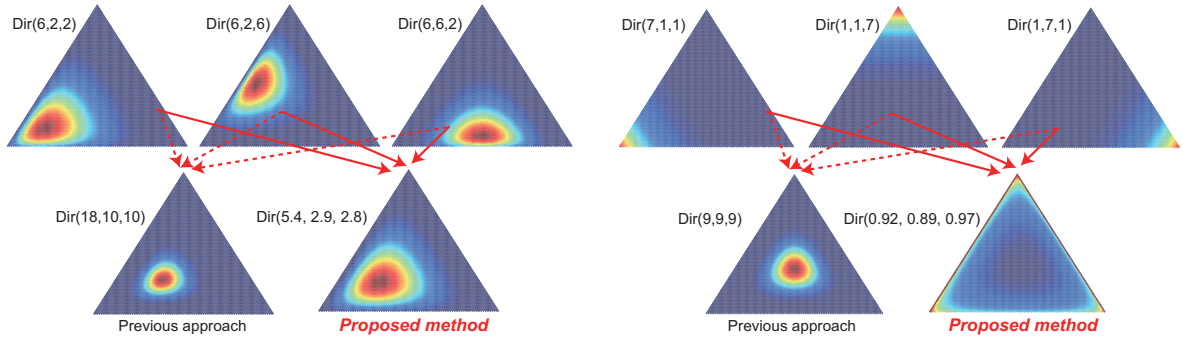


Figure 4. Song-set model estimation of previous approach and a model estimated by our proposed method.

2.3 Typicality over a set of Songs

Given the type of each song, we wish to compute the typicality of a song as compared to a set of other songs. In Section 2.1, we defined the typicality of a sequence of type P from an information source having distribution Q. Therefore, we need some way to estimate Q from the set of songs (i.e. types) beforehand. Actually, we do not have to estimate a single Q but compute an expectation around it:

$$\text{Typicality}(P|\Theta) = \mathbb{E}[\exp(-D(P||\theta))]_{\theta\sim\text{Dir}(\alpha)} \quad (19)$$

where $\Theta = \{\theta_1, \cdots, \theta_n\}$ is a set of types of other songs and $\text{Dir}(\alpha)$ is a prior Dirichlet distribution from which each $\theta_i \in \Theta$ is deemed to be generated.

In the previous work [16], we estimated the hyperparameter α by just summing the topic distributions $\theta_1, \cdots, \theta_n$. As shown in Figure 4, however, this could lead to an undesirable result and we employ a Bayesian formula to estimate α. This derivation is based on the following Dirichlet and Gamma distributions:

$$\text{Dir}(\theta|\alpha) = \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k - 1} \quad (20)$$

$$\text{Ga}(\alpha|a,b) = \frac{b^a}{\Gamma(a)} \alpha^{a-1} e^{-b\alpha} \quad (21)$$

Therefore,

$$p(\alpha|\Theta) \propto p(\Theta|\alpha)p(\alpha) \quad (22)$$

$$\propto \prod_k \alpha_k^{a-1} e^{-b\alpha_k} \cdot \prod_j \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k}, \quad (23)$$

which leads to

$$p(\alpha_k|\alpha_{k-1}, \Theta) \propto \alpha_k^{a-1} e^{-b\alpha_k} \cdot \prod_j \frac{\Gamma\left(\sum_k \alpha_k\right)}{\Gamma(\alpha_k)} \theta_k^{\alpha_k}. \quad (24)$$

Because we cannot expand $\Gamma(\sum_k \alpha_k)/\Gamma(\alpha_k)$, we make a following approximation with n being a nearest integer to $\sum_{j\neq k} \alpha_k$ [18]:

$$\frac{\Gamma(\sum_k \alpha_k)}{\Gamma(\alpha_k)} = \frac{\Gamma(\alpha_k + \sum_{j\neq k} \alpha_j)}{\Gamma(\alpha_k)} \simeq \frac{\Gamma(\alpha_k + n)}{\Gamma(\alpha_k)} \quad (25)$$

$$= \alpha_k(\alpha_k + 1) \cdots (\alpha_k + n - 1) \quad (26)$$

$$= \prod_{i=0}^{n-1} \alpha_k(\alpha_k + i) \quad (27)$$

$$= \prod_{i=0}^{n-1} \sum_{y\in\{0,1\}} (\alpha_k)^{y_i} (i)^{1-y_i}. \quad (28)$$

Therefore, introducing auxiliary variables

$$y_i \sim \text{Bernoulli}\left(\frac{\alpha_k}{\alpha_k + i}\right), \quad (29)$$

we can make a following Gamma proposal for α_k:

$$p(\alpha_k|\alpha_{k-1}, \Theta) \quad (30)$$

$$\simeq \alpha_k^{a-1} e^{-b\alpha_k} \cdot \prod_j e^{\alpha_k \log \theta_{jk}} \cdot \prod_j \prod_{i=0}^{n-1} \alpha_k^{y_{ji}} \quad (31)$$

$$= \alpha_k^{a + \sum_j \sum_{i=0}^{n-1} y_{ji} - 1} \cdot e^{-\alpha_k (b - \sum_j \log \theta_{jk})} \quad (32)$$

$$= \text{Ga}(a + \sum_j \sum_{i=0}^{n-1} y_{ji}, b - \sum_j \log \theta_{jk}). \quad (33)$$

Because this is just a proposal, we further correct the bias using a Metropolis-Hastings algorithm with the exact likelihood formula (24).

2.4 Computing the Expectation

Once we obtain α from Θ, we can compute the expectation (19) analytically. Denoting $P = (p_1, \cdots, p_K)$ and writing $\mathbb{E}[]$ as $\langle\rangle$,

$$\text{Typicality}(P|\Theta) = \left\langle \exp(-D(P||\theta)) \right\rangle_{\theta\sim\text{Dir}(\alpha)} \quad (34)$$

$$= \left\langle \exp \sum_{k=1}^{K} p_k \log \frac{\theta_k}{p_k} \right\rangle_{\theta\sim\text{Dir}(\alpha)}$$

$$= \frac{1}{\exp(\sum_k p_k \log p_k)} \left\langle \exp \sum_k p_k \log \theta_k \right\rangle_{\theta\sim\text{Dir}(\alpha)}$$

$$= \exp(H(P)) \left\langle \prod_{k=1}^{K} \theta_k^{p_k} \right\rangle_{\theta\sim\text{Dir}(\alpha)}. \quad (35)$$

Here, the second term is

$$\left\langle \prod_{k=1}^{K} \theta_k^{p_k} \right\rangle_{\theta\sim\text{Dir}(\alpha)} = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \int \prod_k \theta_k^{\alpha_k - 1} \cdot \prod_k \theta_k^{p_k} d\theta$$

$$= \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \int \prod_k \theta_k^{\alpha_k + p_k - 1} d\theta$$

$$= \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \frac{\prod_k \Gamma(\alpha_k + p_k)}{\Gamma(\sum_k \alpha_k + p_k)}$$

$$= \frac{1}{\sum_k \alpha_k} \prod_k \frac{\Gamma(\alpha_k + p_k)}{\Gamma(\alpha_k)} . \quad (36)$$

Therefore, from (35) we finally obtain

$$\text{Typicality}(P|\Theta) = \frac{\exp(H(P))}{\sum_k \alpha_k} \prod_k \frac{\Gamma(\alpha_k + p_k)}{\Gamma(\alpha_k)} . \quad (37)$$

3 EXPERIMENTS

The proposed methods were tested in an experiment evaluating the estimated typicality. To evaluate estimated results, we focused on vocal timbre which can be evaluated quantitatively by using the singer's gender.

3.1 Dataset

The song set used for the LDA-model-training and typicality estimation comprised 3,278 Japanese popular songs[1] that appeared on a popular music chart in Japan (`http://www.oricon.co.jp/`) and were placed in the top twenty on weekly charts appearing between 2000 and 2008. Here we refer to this song set as the JPOP MDB.

The song set used for GMM training and k-means training to extract the acoustic features consisted of 100 popular songs from the RWC Music Database (RWC-MDB-P-2001) [9]. These 80 songs in Japanese and 20 in English reflect styles of the Japanese popular songs (J-Pop) and Western popular songs in or before 2001. Here we refer this song set as the RWC MDB.

3.2 Experimental Settings

Conditions and parameters of the methods described in the METHODS section are described here in detail. Some conditions and each parameter value were based on previous work [15, 16].

3.2.1 Typicality estimation

The number of iterations of the Bayesian song-set model estimation described in Subsection 2.3 was 1000.

3.2.2 Extracting acoustic features

For vocal timbre features, we targeted monaural 16-kHz digital recordings and extracted ΔF_0 and 12th-order LPMCCs every 10 ms. To estimate the features, the vocal sound was re-synthesized by using a sinusoidal model. The ΔF_0 was calculated every five frames (50 ms).

The feature vectors were extracted from each song, using as reliable vocal frames the top 15% of the feature frames. Using the 100 songs of the RWC MDB, a vocal GMM and a non-vocal GMM were trained by variational Bayesian inference [3].

3.2.3 Quantization

To quantize the vocal features, we set the number of clusters of the k-means algorithm to 100 and used the 100 songs of the RWC MDB to train the centroids.

[1] Note that some are Western popular songs and English in them .

3.2.4 Training the generative models

Training song models and song-set models of the vocal timbre by LDA, we used all of the 3,278 original recordings of the JPOP MDB.

The number of topics, K, was set to 100, and the model parameters of LDA were trained using the collapsed Gibbs sampler [12]. The hyperparameters of the Dirichlet distributions for topics and words were initially set to 1 and 0.1, respectively.

3.3 Four typicality measures

We evaluated the following four typicality computing conditions.

T1: computing the Euclidean distance
T2: computing the generative probability [16]
T3: computing the KL-divergence, equation (13)
T4: computing the KL-divergence, equation (37)

As a baseline method, under the T1 condition, one simple method used to estimate the typicality of vocal timbre calculated the Euclidean distance between mean feature vectors of a song and a song-set. Each mean vector was calculated from each song, using the reliable vocal frames, and was normalized by subtracting the mean and dividing by the standard deviation of all mean vectors.

Under the T2 condition, one typicality between a song and a set of songs is obtained by calculating a generative probability [16] of song P calculated using a song-set model of song Q. This typicality $p_g(P|Q)$ is defined as follows:

$$\log p_g(P|Q) = \frac{1}{N_P} \sum_{n=1}^{N_P} \log p(\mathbf{x}_{P,n} | \, \mathbb{E}[\boldsymbol{\theta}_Q], \mathbb{E}[\boldsymbol{\phi}]), \quad (38)$$

$$p(\mathbf{x}_{P,n} | \, \mathbb{E}[\boldsymbol{\theta}_Q], \mathbb{E}[\boldsymbol{\phi}]) = \sum_{k=1}^{K} \left(\mathbb{E}[\theta_{Q,k}] \cdot \mathbb{E}[\phi_{k,v}] \right), \quad (39)$$

where $\mathbb{E}[\cdot]$ is the expectation of a Dirichlet distribution, N_P is the number of frames, and v is the corresponding index (the word id) of the K-dimensional 1-of-K observation vector $\boldsymbol{x}_{b,n}$.

The other two typicalities, under the T3 and T4 conditions, are calculated $\text{Typicality}(P, Q)$ by using equations (13) and (37), respectively.

3.4 Experiment: musical typicality estimation

We evaluated the four typicality computing conditions (T1-T4) in combination with the following three song-set modeling conditions.

M1: computing a mean vector
M2: summing the Dirichlet hyperparameters [16]
M3: Bayesian estimation of the hyperparameters described in Subsection 2.3

We computed typicalities under five evaluation conditions T1+M1, T2+M2, T3+M2, T3+M3, and T4+M3.

Our typicality evaluation experiment used five hundred songs by a hundred singers (50 male and 50 female), each singer sung five songs. The songs are taken from the JPOP MDB and each of the songs included only one vocal. To

Evaluated conditions	First selection			Second selection			Third selection			Fourth selection			Fifth selection		
	ρ_{m}	ρ_{f}	ρ_{mf}	ρ_{m}	ρ_{f}	ρ_{mf}	ρ_{m}	ρ_{f}	ρ_{mf}	ρ_{m}	ρ_{f}	ρ_{mf}	ρ_{m}	ρ_{f}	ρ_{mf}
T1+M1	.855	.866	.855	.775	.821	.798	.935	.835	.882	.870	.876	.872	.914	.842	.876
T2+M2	.924	.930	.860	.905	.921	.866	.953	.918	.879	.925	.938	.875	.945	.910	.864
T3+M2	.921	.927	.861	.912	.921	.871	.951	.919	.880	.924	.935	.876	.944	.907	.865
T3+M3	.940	.961	.931	.910	.961	.926	.962	.955	.944	.936	.967	.934	.952	.950	.933
T4+M3	.936	.973	.942	.844	.973	.896	.968	.962	.952	.930	.976	.936	.970	.949	.939

Table 1. Pearson correlation coefficients of a hundred songs under the five evaluated conditions ("T4+M3" is the proposed method) and the underline means the highest value. The songs are randomly selected five times from five hundred songs.

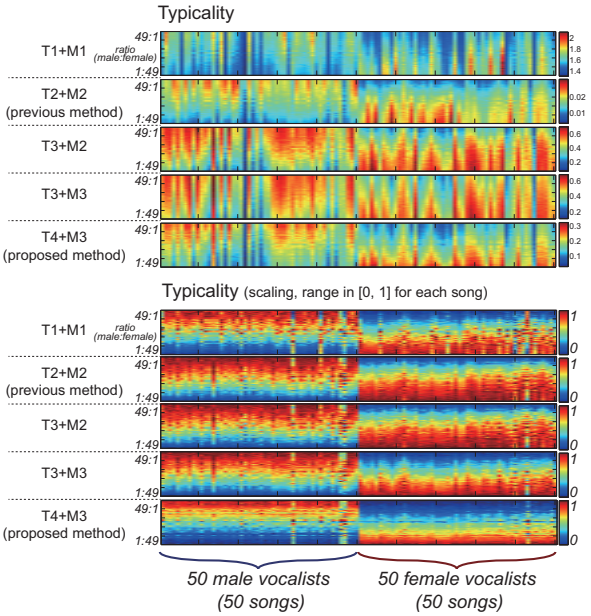


Figure 5. Estimated typicalities (first selection) and those scaled values for each of the five evaluation conditions.

estimate musical typicality, a hundred songs by different singers are randomly selected five times. Then, to integrate or estimate song-set models, fifty songs are randomly selected from the songs with different ratios of the number of male singers to female singers $(1:49, 2:48, ..., 49:1)$. When a model with a high proportion of female songs is used, the typicality of songs sung by females is higher than the typicality of songs sung by males (and vice versa).

Estimated typicality was evaluated against the Pearson product-moment correlation coefficient between the computed typicality the ratio of the number of male singers to female singers with respect to song-set modeling. Before computing the coefficients, the typicality for each song was scaled to have values from 0 to 1 for evaluating smooth transition. Let ρ_{m}, ρ_{f}, and ρ_{mf} denote the coefficients under a set of songs consist of 50 songs by male singers, 50 songs by female singers, and all 100 songs, respectively.

The estimated typicalities and those scaled values are shown in Figure 5 for each of the five evaluation conditions. The Pearson's correlation coefficients are listed in Table 1. The results show that the proposed method achieved the highest value of the correlation coefficient (T4+M3). This means that the proposed method works better than the baseline method based on the Euclidean distance of mean vectors (T1+M1) and the previous method based on computing the generative probabilities (T2+M2). The results also show that estimated musical typicality by using the proposed method can reflect the ratio between the number of songs belonging to a class (*e.g.*, male singer) and the number of songs belonging to another class (*e.g.*, female singer).

4 CONCLUSION

We proposed a method for estimating musical typicality based on the type theory. Although this method is used for quantized acoustic features for vocal timbre in this paper, it can be used for other discrete sequence representations of music, such as quantized other acoustic features (*e.g.*, MFCCs to represent musical timbre/genre), lyrics and musical score. It can also be used with probabilistic representation instead of estimating musical similarities of all possible song-pairs by using a model trained from each song, for integrating or collaboration with other probabilistic approach as a unified framework.

Our definition of musical typicality was based on the central tendency [2] which is only the definition to be computed from the audio data; this is the reason to adopt it. In future work we expect to deal with two other definitions in cognitive psychology are *frequency of instantiation* and *ideals*. The frequency of instantiation is a perspective on social recognition, that is, things with a lot of exposure on media or in advertisements are typical, and ideals focuses on an ideal condition of the category, that is, things that are close to an ideal condition are typical.

Musical typicality can be used not only for music-listening support interface such as retrieving an uniqueness song or visualizing typicalities, but also to do this by developing a music-creation support interface enabling high/low typicality elements (*e.g.*, timbre and lyrics) to be used to increase originality or visualize typicality in order to avoid unwarranted accusations of plagiarism. We also want to promote a proactive approach to encountering and appreciating content by developing music-appreciation support technology that enables people to encounter new content in ways based on its typicality to other content.

5 ACKNOWLEDGMENT

This paper utilized the RWC Music Database (Popular Music). This work was supported in part by CREST, JST.

6 REFERENCES

[1] Jean-Julien Aucouturier and Francois Pachet. Music similarity measures: What's the use? In *Proc. ISMIR 2002*, pages 157–163, 2002.

[2] Lawrence W. Barsalou. Ideals, central tendency, and frequency of instantiation as determinants of graded structure in categories. *journal of Experimental Psychology: Learning, Memory, and Cognition*, 11(4):629–654, 1985.

[3] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer-Verlag New York, Inc., 2006.

[4] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[5] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. New-York: Wiley, 2006.

[6] Imre Csiszár. The method of types. *IEEE Trans. on Information Theory*, 44(6):2505–2523, 1998.

[7] Imre Csiszár and János K orner. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Academic Press, 1981.

[8] Hiromasa Fujihara, Masataka Goto, Tetsuro Kitahara, and Hiroshi G. Okuno. A modeling of singing voice robust to accompaniment sounds and its application to singer identification and vocal-timbre-similarity based music information retrieval. *IEEE Trans. on ASLP*, 18(3):638–648, 2010.

[9] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC music database: Popular, classical, and jazz music databases. In *Proc. ISMIR 2002*, pages 287–288, 2002.

[10] Masataka Goto and Keiji Hirata. Recent studies on music information processing. *Acoustical Science and Technology (edited by the Acoustical Society of Japan)*, 25(6):419–425, 2004.

[11] Masatala Goto. A real-time music scene description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, 43(4):311–329, 2004.

[12] Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. In *Proc. Natl. Acad. Sci. USA (PNAS)*, volume 1, pages 5228–5235, 2004.

[13] Peter Knees and Markus Schedl. A survey of music similarity and recommendation from music context data. *ACM Trans. on Multimedia Computing, Communications and Applications*, 10(1):1–21, 2013.

[14] Tomoyasu Nakano, Jun Kato, Masahiro Hamasaki, and Masataka Goto. PlaylistPlayer: An interface using multiple criteria to change the playback order of a music playlist. In *Proc. ACM IUI 2016*, 2016.

[15] Tomoyasu Nakano, Kazuyoshi Yoshii, and Masataka Goto. Vocal timbre analysis using latent Dirichlet allocation and cross-gender vocal timbre similarity. In *Proc. ICASSP 2014*, pages 5239–5343, 2014.

[16] Tomoyasu Nakano, Kazuyoshi Yoshii, and Masataka Goto. Musical similarity and commonness estimation based on probabilistic generative models. In *Proc. IEEE ISM 2015*, 2015.

[17] Markus Schedl, Emilia Gómez, and Julián Urbano. Music information retrieval: Recent developments and applications. *Foundations and Trends in Information Retrieval*, 8(2–3):127–261, 2014.

[18] Yee Whye Teh. A bayesian interpretation of interpolated kneser-ney. *Technical Report TRA2/06*, 2006.

MUSICDB: A PLATFORM FOR LONGITUDINAL MUSIC ANALYTICS

Jeremy Hyrkas
University of Washington
hyrkas@cs.washington.edu

Bill Howe
University of Washington
billhowe@cs.washington.edu

ABSTRACT

With public data sources such as Million Song dataset, researchers can now study longitudinal questions about the patterns of popular music, but the scale and complexity of the data complicate analysis. We propose MusicDB, a new approach for longitudinal music analytics that adapts techniques from relational databases to the music setting. By representing song timeseries data relationally, we aim to dramatically decrease the programming effort required for complex analytics while significantly improving scalability. We show how our platform can improve performance by reducing the amount of data accessed for many common analytics tasks, and how such tasks can be implemented quickly in relational languages — variants of SQL. We further show that expressing music analytics tasks over relational representations allows the system to automatically parallelize and optimize the resulting programs to improve performance. We evaluate our system by expressing complex analytics tasks including calculating song density and beat-aligning features and showing significant performance improvements over previous results. Finally, we evaluate expressiveness by reproducing the results from a recent analysis of longitudinal music trends using the Million Song dataset.

1. INTRODUCTION

Over the past decade, a concerted investment in building systems to help extract knowledge from large, noisy, and heterogeneous datasets — big data — has had a transformative effect on nearly every field of science and industry. Progress has also been fueled by the availability of public datasets (e.g., the Netflix challenge [1], early releases of Twitter data [14], Google's syntatic n-grams data [7], etc.), which have focused and accelerated research in both domain science and systems. The field of Music Information Retrieval (MIR) appears to been less affected, as complications from copyright-encumbered properties have limited the introduction of big datasets to the community. Now, however, such datasets are finally making their way into the field.

In other fields we have observed that as the data size and scope of problems increased, issues of scale became the bottleneck: single-site solutions written in R or python give way to distributed shared-nothing systems that can readily handle large datasets. These systems relieve the user of worrying about issues such as memory management, concurrency and distributed computing by focusing on limited data models and APIs. General purpose systems such as Hadoop [21] and Spark [22] provide MapReduce-style dataflow computations [4], but can be hard to program and optimize because of their generality and relatively low-level interfaces. Increasingly, programmers are experimenting with the models and languages of relational databases [11,12,17] for non-relational analytics over timeseries, graphs, images, and more due to their higher-level programming abstractions, simpler data models, and automatic optimization.

In this paper, we propose a platform for *longitudinal music analytics* built on a relational big data system. Because music data (which may include multi-dimensional arrays and timeseries of extracted features) is ostensibly not relational upon collection, we describe a "relationalization" of the data to afford distributed, parallel processing. Then, we present four algorithms found in music analytics tasks in the MIR literature and show that they can be expressed in declarative relational languages similar to SQL, affording scalability, portability, and automatic optimization, and freeing the programmer from systematic concerns related to memory management, concurrency, and distributed processing. The four algorithms are song density, feature beat-alignment, pitch keyword distribution by year, and timbre keyword distribution by year. The first two algorithms are common music analysis tasks, and the latter two are highly computational algorithms presented in a high profile MIR study over the Million Song Data set [20]. We reproduce prior results with only a few lines of code and a significant performance improvement over prior experiments on similar large-scale systems.

We propose our model as an approach for platforms to raise the level of abstraction for longitudinal music analytics and reduce the barrier to entry for researchers in musicology and sociology.

1.1 Million Song Dataset

The key dataset used in our experiments is the The Million Song Dataset (MSD) [3]. The dataset includes metadata and extracted features from one million pop music songs from a period of decades, including information such as genre tags, chroma measurements, timbre and loudness


© Jeremy Hyrkas, Bill Howe.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Jeremy Hyrkas, Bill Howe. "MusicDB: A Platform for Longitudinal Music Analytics", 17th International Society for Music Information Retrieval Conference, 2016.

measurements, detected beats, artist and song metadata such as year, duration and location, and many other attributes. This dataset has been highly influential in the MIR community, leading to the Million Song Dataset Challenge [16], as well as being a key data set used to study music history [20] and MIR tasks such as cover song detection [2, 10]. The MSD is available as available as a large collection of HDF5 [5] files; at over 250GB even when heavily compressed, it is the largest public dataset in MIR and is the first true "Big Data" dataset in the field.

The MSD is available in a number of formats, including a relational database. However, the database representation include the metadata only, and cannot be used for the content-based longitudinal analytics tasks we aim to support with MusicDB.

2. RELATED WORK

Serra et al. provide a longitudinal analysis of music trends in popular music by utilizing the MSD [20] . The authors study the chroma, timbre, and loudness components of the MSD and find that the frequency of chroma keywords (which can be thought of as single notes, chords, etc) fit a power law distribution which is mostly invariant across time. However, the authors also find that transitions from one keyword to the next have become more uniform over time, suggesting less complexity in newer music. They also describe shifting trends in timbre and loudness, including numerical evidence that recorded music is increasingly louder on average. In Section 4 we describe these tasks in detail and present new algorithms for them using MusicDB. In Section 5 we evaluate our approach experimentally.

Raffek and Ellis analyzed MIDI files and matched them to corresponding songs in the MSD [18] . Their algorithm is fast but is not distributed; they estimate that running their approach on roughly 140k MIDI files against the MSD would take multiple weeks even when parallelized on their multi-threaded processor with 12 threads.

Bertin-Mahieux and Ellis described a method for finding cover songs in the MSD [2]. The method begins with an aggressive filtering step, which requires computing jumpcodes for the entire MSD and storing them in a SQLite database. Once the number of potential matches for a new song is filtered, a more accurate matching process is run to find cover songs. Using three cores, they computed the jumpcodes for the entire MSD in roughly three days, although matching cover songs once the jumpcodes are computed takes roughly a second per new song. The authors also mention that the jumpcodes had to be stored in many different SQLite tables, as they were unable to index roughly 1.5M codes in a single table. MusicDB provides a platform that can process the data directly, in parallel, without specialized engineering.

Humphrey, Nieto, and Bello also provide a method to detect cover songs [10]. Their method starts by transforming beat-aligned chroma of a song into a high-dimensional, sparse representation by projecting its 2D Fourier Magnitude Coefficients. They then use PCA to reduce dimensionality and use the results to find cover songs using a distance function. The authors claim that using ten threads on a machine with "plenty of RAM", various methods can take between 3-8 hours to complete this computation on the MSD.

Hauger et al. describe the million musical tweets dataset (MMTD) [9] collected from tweets with information about a user's location and what they were listening to at a certain time. This dataset, as well as others, can be used to augment the MSD for new MIR tasks. As it exists, the MSD is available as a directory hierarchy with hundreds of gigabytes of HDF5 files stored on AWS. Incorporating new data in analysis tasks over this dataset requires additional effort in the analysis pipeline, leaving either the authors of the data or the users of the data to write new code to handle the new data source and manually join it with the MSD. MusicDB provides a scalable substrate for such integration tasks.

3. DATA MODEL

A key step in efficiently analyzing the MSD is to represent its information in an appropriate data model. The representation of the MSD available on the website (on million HDF5 files) support efficient lookup by ID, but any more complex processing requires custom programs to be written, and parallelization, concurrency, distribution, and memory management are all the direct responsibility of the programmer. Moreover, tasks that require only a portion of metadata from each song must still access and load all song data from disk.

Instead, we can organize the music data as sets of records. In practice, this "relationalization" of timeseries and multidimensional data can significantly increase the size of the dataset. In our work, the end size of our relationalized data is roughly 500GB, about twice as large as the original dataset. However, all applications we have observed do not require the entire MSD, and the subset of relationalized data necessary for computation is much smaller than the entire MSD in HDF5 format. Further, representing the data as a set of records affords automatic partitioning and parallel processing, as we will see.

The steps to relationalization are as follows:

- Metadata that appears only once per song is inserted into one table (songs), with song ID as the key. This includes fields such as song duration, artist name, song name, etc.

- Nested fields are represented in separate tables, retaining a foreign key to the songs table. To represent the order within the nested field, an additiona column is added. For example, each song segment is represented as a record *(song_id, segment_number, value*, where *segment_number* explicitly encodes the implicit order in the original array. This additional field is one source of the space overhead we find in practice.

- We use additional tables to support specific components of the MSD, including a separate table for each of the following: beat-aligned chroma features, beat-aligned timbre features, and beat-aligned chroma features that have been transposed such that most songs are in C major or C minor.

table	key	arity	non-key fields
songs	song_id	33	duration, key, tempo, etc
segments	song_id, seg_num	31	timbre, loudness, and pitch measurements
mbtags	song_id, tag	3	tag count
bars	song_id, bar_num	4	bar start and confidence
beats	song_id, beat_num	4	beat start and confidence
sections	song_id, section_id	4	section start and confidence
terms	song_id, term	4	term frequency and weight
tatums	song_id, tatum_num	4	tatum start and confidence
similar artists	song_id, artist_id, similar_artist_id	3	N/A

Table 1. Core tables after relationalization of the MSD. Additional tables may be created, such as beat aligned features.

4. ALGORITHMS

We describe four algorithms for scalable analysis of the relationalized MSD dataset. The first two algorithms are common MIR tasks, and the latter two come from an influential study using the MSD [20].

4.1 Song Density

In 2011, Lamere described a Hadoop-based approach for calculating song density from the MSD [15]. A song's density is defined as the number of detected segments divided by the duration of the song in seconds. To calculate this metric for every song in the MSD, Lamere provides a MapReduce [4] algorithm that scans each song, extracts the segments, and computes the density. The map function was written in Java specifically for this purpose.

This task can be expressed directly with no custom general purpose code in SQL. In MusicDB, song density for a single song can be expressed as a simple count query over the segments table, followed by a join with the songs table and division by song duration. This method generalizes to the following query (in an imperative dialect of SQL used by the Myria system [8]) that computes the density for all songs.

Query 1. Lines 1-2 scan the relevant relations. Lines 4-7 count the number of segments per song and lines 8-14 calculate the density by dividing the number of segments by the duration in seconds. Line 15 stores the result.

```
1  segments = SCAN(SegmentsTable);
2  songs = SCAN(SongsTable);
3  -- implicit GROUP BY song_id
4  seg_count = SELECT
5      song_id,
6      COUNT(segment_number) AS c
7      FROM segments;
8  density = SELECT
9      songs.song_id,
10     (seg_count.c /
11         songs.duration) AS density
12     FROM songs, seg_count
13     WHERE songs.song_id =
14         seg_count.song_id;
15 store(song_density);
```

4.2 Beat-aligning features

While chroma and timbre data in the MSD are provided on a per-segment basis, it is often more useful to align these features to beats, which are easier to interpret musically. Beat-aligning these features is an extremely common and useful processing step that is used in cover song detection [2], longitudinal music studies [20], and many other MIR tasks, and is therefore a useful task to consider for MusicDB.

In 2011, Serra identifies dynamic time warping as one of the best methods for beat-alignment [19]. Dynamic time warping involves creating an SxB matrix, where S is the number of segments of a song and B is the number of beats. If a segment s overlaps with a beat b, the b, s entry of the matrix is set to the fraction of the segment contained in the beat (i.e. 1 if the segment falls entirely in the beat, .5 if the beat contains exactly half of the segment, etc). All other entries are set to 0, and then the rows are normalized such that each row of values sums to 1. Segment-based features such as chroma or timbre can then be beat-aligned by transposing the beat matrix and performing matrix multiplication on the features (a BxS matrix multiplied by a SxF matrix will result in a BxF matrix, where F is the number of features). Some additional regularization of rows is performed for chroma features.

In a relational system, the time warp operation can be computed using just two operations: a join and an aggregation. We first join the segments and beats table on the start and end time of each segment and bar, such that overlapping segments and beats are joined:

Query 2. Portion of a query that joins overlapping segments and beats of a song so that beat aligned features can be computed.

```
1  JOIN segments, beats WHERE
2      -- segment overlaps start of beat
3      (seg_start <= beat_start
4          AND seg_end <= beat_start)
```

```
    OR
    -- segment overlaps end of beat
    (seg_start < beat_end
        AND seg_end >= beat_end)
    OR
    -- segment fully inside beat
    (seg_start > beat_start
        AND seg_end < beat_end)
    OR
    -- beat fully inside segment
    (seg_start < beat_start
        AND seg_end > beat_end)
;
```

We can then perform two aggregations on the result, and re-join the aggregate queries to perform an operation identical to multiplying the features by a time warp matrix. Refer to Figure 1, which visualizes this query. In each aggregation, we will calculate the fraction of a segment that falls within a beat, which is defined as the length of the segment that falls within the beat divided by the length of the segment (or 1.0 if the segment spans the beat).

On the right side of the diagram, we compute the first aggregation. We use the fraction of the segment that falls within a beat to scale each feature of that segment (i.e. chroma or timbre features), and then take the sum of the scaled features per beat.

On the left side of the diagram, we perform a second aggregate which simply computes the sum of the segment fractions for each beat. We call this sum the divisor.

The two aggregates are then re-joined on beat id and the weighted sum from the first aggregate is normalized by dividing each value by the sum from the second aggregate. This divisor serves the same function as making sure rows of the time warp matrix sum to 1. The result of the joined aggregates is a table with the schema (song_id, beat_id, feature columns), which is a relational representation of the BxF feature matrix described above.

This algorithm, while correct, is not necessarily optimal. Specifically, performing two aggregates over the same joined relation and then re-joining is an expensive operation. Relational engines that support *window functions* offer an alternative approach. A window function makes a single pass over a dataset, applying an aggregation over each window as defined by a grouping value or a fixed size. The engine on which MusicDB is based (a variant of the Myria system [8]) provides a generalization of window functions, but we do not employ that mechanism here to ensure reuse across platforms.

4.3 Pitch Keyword Distribution by Year

In 2012, Serra et al studied the progression of *chroma keywords* over time [20]. The authors form these keywords by transposing every song to an equivalent main tonality by correlating to tonal profiles provided by [13]. After that, the beat-aligned chroma values are discretized to binary values (1 if the value is greater than 0.5, 0 otherwise) and then concatenated. Intuitively, this discretization represents whether or not a certain pitch is present or not. These

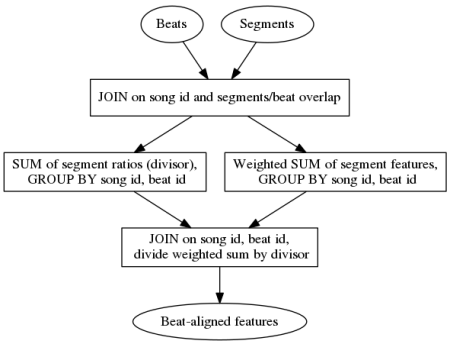


Figure 1. The relational algebra expression for beat-aligning features from segments. Segments and beats from a song are joined on song ID and overlap conditions. Two aggregations are performed and rejoined to form a table with features now aligned to beats instead of segments. The process of generated two results and joining them can be costly and can be aided by using window functions provided in many database systems.

keywords can then be summed over years and used to fit a distribution. The authors show that these chroma keywords fit a power law which has variables that are near invariant over time.

The discretization and sum of keywords can be implemented using our data model by the following SQL-like program:

Query 3. Lines 1-2 scan the appropriate tables. Lines 4-19 (with some lines emitted) create an integer keyword based on the value of each pitch column. Lines 21-26 count keywords by year and line 28 stores the result.

```
songs = SCAN(SongsTable);
pitch = SCAN(PitchTransposed);

keywords = SELECT
    p.song_id AS song_id,
    p.beat_number AS beat_number,
    CASE WHEN p.basis0 >= 0.5
        THEN int(pow(2, 11))
        ELSE 0
    END
    +
    CASE WHEN p.basis1 >= 0.5
        THEN int(pow(2, 10))
        ELSE 0
    END
    +
    ...
    AS keyword
    FROM pitch p;

-- implicit GROUP BY year, keyword
yearPitchKeywords = SELECT
    s.year, k.keyword,
    count(k.keyword)
    FROM songs s, keywords k
```

```
26      WHERE s.song_id = k.song_id;
27
28  store(yearPitchKeywords);
```

4.4 Timbre Keyword Distribution by Year

Similar to the pitch keywords in Section 4.3, the timbre values from the MSD can be discretized and studied over time. Serra et al sample timbre values from by year such that no year is more represented from than any other [20]. From this sample set, they estimate the tertiles for each timbre value. The tertile values are then used to discretise the timbre values similarly to the pitch keywords. For each timbre column, the value is converted to a 0 if it is less than the first tertile, 1 if it is less than the second tertile, and 2 otherwise; concatenating these values makes one timbre keyword.

The authors fit power laws to the timbre keywords as before. However, they find that the power law distributions significantly differ over time. They conclude that while the chroma distribution appears to be time invariant, timbre information (which encodes many complex factors such as instrument use, tone, and production style) has changed over time; additionally, the authors find that while there are local shifts in timbre values, the distribution is slowly converging.

The query for finding timbre keywords is similar to the query in Section 4.3, but slightly more complex. It requires access to a quantiles function that takes a quantile constant and a column, and returns an integer representing which quantile a row's column value falls in (for example, quantile(3, col) returns 0 if col is in the first tertile of the values contained in col, 1 if it in the second, and 2 if it is in the third).

Query 4. Lines 1-2 scan the appropriate tables. Lines 4-15 (with some lines emitted) create an integer keyword based on the tertile each timbre column falls in (the function returns 0 through 2). Lines 17-22 count keywords by year and line 24 stores the result.

```
1   songs = SCAN(SongsTable);
2   timbre = SCAN(TimbreBeatAligned);
3
4   keywords = SELECT
5       t.song_id as song_id,
6       t.beat_number AS beat_number,
7       int(pow(10, 11)) *
8           QUANTILE(3, t.basis1)
9       +
10      int(pow(10, 10)) *
11          QUANTILE(3, t.basis2)
12      +
13      ...
14      AS keyword
15      FROM timbre t;
16
17  -- implicit GROUP BY year, keyword
18  yearTimbreKeywords = SELECT
19      s.year, k.keyword,
20      count(k.keyword)
21      FROM songs s, keywords k
22      WHERE s.song_id = k.song_id;
23
24  store(yearTimbreKeywords);
```

5. EXPERIMENTAL EVALUATION

We evaluate the feasibility of our relationalized approach by measuring the wall-clock performance of our implementation on a 72-worker cluster and comparing performance qualitatively with reports from the literature. We find that the entire MSD dataset can be analyzed in seconds or minutes, where previous results on large-scale platforms report tens of minutes and required custom code, while smaller-scale implementations reported taking hours or days.

5.1 Song Density

We ran the song density query described in Section 4.1 on a MusicDB cluster with 72 worker threads. The computation takes roughly half a minute, a far cry from the 20 minutes described in [15]. These two results are not directly comparable; the example in [15] was run on virtual machines in EC2, so the hardware, software, number of nodes, and most other factors are not comparable. However, the Hadoop implementation required custom code, and the underlying platform on which we implemented these algorithms (a variant of the Myria system [8]) has been previously shown to significantly outperform Hadoop on general tasks.

By using a relational model to represent the MSD and using a distributed analytics database, we can quickly analyze the MSD using a simple query and allowing the system and optimizer handle the complexities of computation.

5.2 Pitch Keyword Distribution by Year

We ran the query described in Section 4.3 on our 72-node production cluster of MusicDB. Computing the pitch keywords took about three minutes, while counting keywords by year took an additional minute. The resulting dataset contains the frequency for each keyword per year, and has the schema (Keyword, Year, Count) with (Keyword, Year) as the primary key. It is small enough ($<$ 1GB) to download locally and perform more complicated statistical tasks, such as fitting power law distributions over counts per year as in [20].

Figures 2, 3, and 4 show the power law distributions for pitch keywords in the years 1965, 1975, and 1985. We confirm the author's results [20] that the power law coefficient is invariant over time. We used the R package poweRlaw [6] to perform this post-processing analysis.

The database system we used does not have complex functions such as power law fitting, so the last step of the computation must be performed out of the system. However, this step could be run in parallel as the data for each year is independent. Alternatively, in a more general distributed system, the keyword counts for each year could be

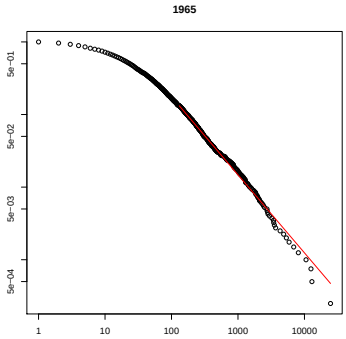


Figure 2. Power law distribution for pitch keywords from songs released in 1965.

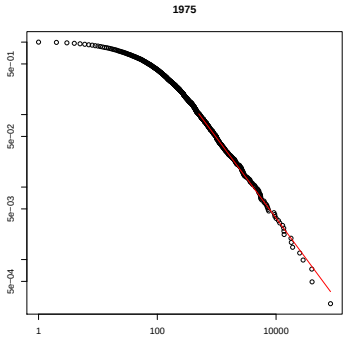


Figure 3. Power law distribution for pitch keywords from songs released in 1975.

partitioned and evaluated in a distributed manner. We did not demonstrate this capability in this case to avoid a contrivance; the resulting data's size was not large enough to justify the approach.

6. DISCUSSION AND FUTURE WORK

Distributed analytics systems have made it easier and faster to perform complex analysis on large datasets. In Section 2 we briefly mentioned several recent studies using the MSD. The authors of these studies ran experiments that ran on single-node systems, often taking hours or days to complete. However, most of these tasks are embarrassingly parallel and could be run not only in parallel on a single machine, but on thousands of nodes in a distributed system. Big data systems exist to empower users to easily analyze data in such a distributed environment. As more large dataset become available in the MIR community, it is no longer feasible or necessary to run single or mutli-core algorithms locally for weeks at a time.

We have shown that representing the data in the MSD as tables can reduce the amount of data necessary for computations (for example, only reading the segment and song tables in Section 3, and only the segment and beat tables in Section 4.2). This works especially well in a relational system, where joining tables is a common task with many optimizations. However, reading and distributed less data is helpful outside of relational systems as well. Even though we showed that many common MIR tasks can be expressed relationally, some tasks are still very difficult to implement in an imperative language. If a distributed system such as Hadoop or Spark is more preferable for a given task, relationalizing the data can still be used to reduce the data necessary for computation in these systems. Finally, since these systems utilize higher level coding models that abstract away parallelization and distributed computation, they may empower musicologists who are less familiar with these concepts to ask quickly questions over larger data sets.

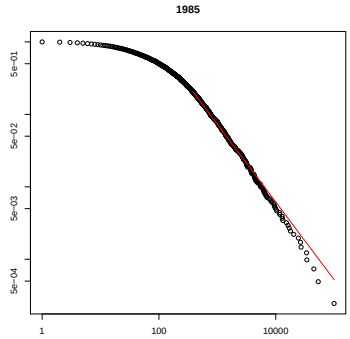


Figure 4. Power law distribution for pitch keywords from songs released in 1985.

7. REFERENCES

[1] James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[2] Thierry Bertin-Mahieux and Daniel PW Ellis. Large-scale cover song recognition using hashed chroma landmarks. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2011 IEEE Workshop on*, pages 117–120. IEEE, 2011.

[3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[4] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008.

[5] Mike Folk, Albert Cheng, and Kim Yates. Hdf5: A file format and i/o library for high performance computing applications. In *Proceedings of Supercomputing*, volume 99, pages 5–33, 1999.

[6] Colin S. Gillespie. Fitting heavy tailed distributions: The poweRlaw package. *Journal of Statistical Software*, 64(2):1–16, 2015.

[7] Yoav Goldberg and Jon Orwant. A dataset of syntactic-ngrams over time from a very large corpus of english books. 2013.

[8] Daniel Halperin, Victor Teixeira de Almeida, Lee Lee Choo, Shumo Chu, Paraschos Koutris, Dominik Moritz, Jennifer Ortiz, Vaspol Ruamviboonsuk, Jingjing Wang, Andrew Whitaker, et al. Demonstration of the myria big data management service. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 881–884. ACM, 2014.

[9] David Hauger, Markus Schedl, Andrej Košir, and Marko Tkalčič. The million musical tweets dataset: What can we learn from microblogs. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR 2013)*, 2013.

[10] Eric J Humphrey, Oriol Nieto, and Juan Pablo Bello. Data driven and discriminative projections for large-scale cover song identification. In *ISMIR*, pages 149–154, 2013.

[11] Marcel Kornacker and Justin Erickson. Cloudera impala: Real time queries in apache hadoop, for real. *ht tp://blog. cloudera. com/blog/2012/10/cloudera-impala-real-time-queries-in-apache-hadoop-for-real*, 2012.

[12] Tim Kraska, Ameet Talwalkar, John C Duchi, Rean Griffith, Michael J Franklin, and Michael I Jordan. Mlbase: A distributed machine-learning system. In *CIDR*, 2013.

[13] Carol L Krumhansl. *Cognitive foundations of musical pitch*, volume 17. Oxford University Press New York, 1990.

[14] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[15] Paul Lamere. How to process a million songs in 20 minutes. http://musicmachinery.com/2011/09/04/how-to-process-a-million-songs-in-20-minutes/, 2011.

[16] Brian McFee, Thierry Bertin-Mahieux, Daniel PW Ellis, and Gert RG Lanckriet. The million song dataset challenge. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 909–916. ACM, 2012.

[17] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, and Theo Vassilakis. Dremel: interactive analysis of web-scale datasets. *Proceedings of the VLDB Endowment*, 3(1-2):330–339, 2010.

[18] Colin Raffel and Daniel PW Ellis. Large-scale content-based matching of midi and audio files. In *16th International Society for Music Information Retrieval Conference (ISMIR 2015)*.

[19] Joan Serra. Identification of versions of the same musical composition by processing audio descriptions. *Department of Information and Communication Technologies*, 2011.

[20] Joan Serrà, Álvaro Corral, Marián Boguñá, Martín Haro, and Josep Ll Arcos. Measuring the evolution of contemporary western popular music. *Scientific reports*, 2, 2012.

[21] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–10. IEEE, 2010.

[22] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, volume 10, page 10, 2010.

NOISE ROBUST MUSIC ARTIST RECOGNITION USING I-VECTOR FEATURES

Hamid Eghbal-zadeh **Gerhard Widmer**
Department of Computational Perception, Johannes Kepler University of Linz, Austria
hamid.eghbal-zadeh@jku.at

ABSTRACT

In music information retrieval (MIR), dealing with different types of noise is important and the MIR models are frequently used in noisy environments such as live performances. Recently, i-vector features have shown great promise for some major tasks in MIR, such as music similarity and artist recognition. In this paper, we introduce a novel noise-robust music artist recognition system using i-vector features. Our method uses a short sample of noise to learn the parameters of noise, then using a Maximum A Postriori (MAP) estimation it estimates clean i-vectors given noisy i-vectors. We examine the performance of multiple systems confronted with different kinds of additive noise in a clean training - noisy testing scenario. Using open-source tools, we have synthesized 12 different noisy versions from a standard 20-class music artist recognition dataset encountered with 4 different kinds of additive noise with 3 different Signal-to-Noise-Ratio (SNR). Using these datasets, we carried out music artist recognition experiments comparing the proposed method with the state-of-the-art. The results suggest that the proposed method outperforms the state-of-the-art.

1. INTRODUCTION

In MIR, the task of music artist recognition [1] is to recognize an artist, from a part of a song. In real life, MIR systems have to cope with different kinds of noise; example situations include music played in a public area such as a pub or in a live performance. MIR systems are usually trained with the high quality data from studio recordings or noise-free audios, yet they may be used in noisy environments.

In this paper, we are targeting a use-case, when an artist recognition mobile app is used in a noisy environment, while the artist recognition models are trained on clean data and are integrated inside the app. In such a use-case, the models can not be trained or adapted on the mobile phone, due to the limitation of computation. But using the app, a short example of the noise can be prepared to improve the performance of the system.

I-vector extraction is an unsupervised high-level feature extraction technique that extracts excerpt-level features using frame-level features of an audio excerpt. I-vectors were proposed for the first time in the field of speaker verification [4], and then they were frequently used in other areas such as emotion [29], language [5], and accent recognition [1] and audio scene detection [9]. Recently, they were imported into the MIR domain, for singing language identification [17], music artist recognition [7] and music similarity [6]. I-vector features provide a fixed-length low-dimensional representation for songs which can capture specific variabilities from acoustic features using Factor Analysis (FA). In [7], i-vector systems used with noise-free data, and clean mp3 audio files were used in the artist recognition experiments.

I-vector based systems consist of 4 main modules: 1) frame-level feature extraction such as Mel-Frequency Cepstrum Coefficients (MFCC), 2) i-vector extraction, 3) inter-class compensation and finally, 4) i-vector scoring. Within-Class Covariance Normalization (WCCN) and Linear Discriminant Analysis (LDA) are examples of methods used in the inter-class compensation step. Cosine similarity and Probabilistic Linear Discriminant Analysis (PLDA) scoring are examples of methods used in the scoring step.

In this paper, we propose a noise-robust artist recognition system using i-vector features that can be adapted to different kinds of additive noise. We add an estimation step after i-vector extraction, which estimates clean i-vectors given noisy i-vectors in a clean training - noisy testing scenario. Our method is superior because it can be used to adapt i-vector based systems to different kinds of noise, without training the models with noisy data. This is done by learning the parameters of noise for different noisy environments given a short example of additive noise and estimating clean i-vectors that perform well with the models trained on clean data.

2. RELATED WORK

To make a MIR system robust to noise, a solution would be to include noise information inside the MIR models by adding noisy samples in training data. For example, in [18] a method called multi-style training is proposed for

[1] We use the term music artist or artist to refer to the singer or the band of a song.

© Hamid Eghbal-zadeh Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Hamid Eghbal-zadeh Gerhard Widmer. "noise robust music artist recognition using i-vector features", 17th International Society for Music Information Retrieval Conference, 2016.

speaker verification in noisy environments. This method uses noisy data provided in the training set to extract noisy training i-vectors, with the i-vector extraction models trained on clean data. But it trains the inter-class compensation and scoring models with noisy training i-vectors.

Because MIR models are usually expensive to train, if no noisy data is available in training, the only option would be to use the models trained on clean data for testing the noisy data and try to reduce the effect of noise on the MIR models.

In recent years, research focused on studying the vulnerability of i-vectors to noise and providing methods for noise compensation. The de-noising techniques used with i-vectors can be categorized according to the level they work on (audio, frame-level features, i-vector extraction, or scoring) [22].

In [8, 25] spectral and wavelet based enhancement approaches on the signal level were examined with i-vectors. In [14], spectrum estimators were used for frame-level noise compensation, while in [19–21], multiple approaches were tested using vector Taylor series (VTS) in the cepstral domain for noise compensation on the feature level. Both signal-level and feature-level noise compensation techniques have shown inconsistencies because of their dependency on the type and level of noise. Also, they are mostly designed for speech enhancement and not for music. And in [18, 24] a method was proposed that improves the scoring but it uses noisy audios in model training.

In [16], a novel method called "i-MAP" is proposed for the purpose of speaker verification in noisy environments that uses an extra estimation step between i-vector extraction and inter-class compensation steps. In the estimation step, it estimates clean i-vectors given noisy i-vectors and further uses these estimations with inter-class compensation and scoring models that are trained on clean data. This method benefits from the Gaussian assumption of i-vectors and proposes a MAP estimation of a clean i-vector given a noisy i-vector. The i-MAP method can be used with different types and levels of additive noise with the models that are trained on clean data.

3. REVIEW OF I-VECTOR SYSTEMS

An i-vector refers to vectors in a low-dimensional space called I-vector Space. The i-vector space models variabilities encountered with both the artist and song [6] where, the song variability defines as the variability exhibited by a given artist from one song to another.

The i-vector space is created using a matrix $\mathbf{T}$ known as i-vector space matrix. This matrix is obtained by factor analysis, via a procedure described in details in [4]. In the resulting space, a given song is represented by an i-vector which indicates the directions that best separate different artists. This representation benefits from its low dimensionality and Gaussian distribution which enables us to use the properties of Gaussians in the i-vector space.

Conceptually, a Gaussian mixture model (GMM) mean supervector $\mathbf{M}$ adapted to a song from artist α can be decomposed as follows:

$$\mathbf{M} = m + \mathbf{T}.y \tag{1}$$

where m is the GMM mean supervector and $\mathbf{T}.y$ is an offset. The low-dimensional subspace vector y is a latent variable with the standard normal prior and the i-vector w is a MAP estimate of y. The UBM is a GMM that is trained unsupervised on acoustic features of sufficient amount of songs. Also, $\mathbf{M}$ is assumed to be normally distributed with mean vector m.

The obtained i-vector is an artist and song dependent vector. The low-rank rectangular matrix $\mathbf{T}$ (i-vector space matrix) is used to extract i-vectors from statistical supervectors of songs which are computed using UBM.

Using the UBM, we calculate statistical supervectors for a specific song s. These statistical supervectors are known as 0^{th} and 1^{st} order statistics (N_s and F_s) of song s:

$$(\ 0^{\text{th}} \text{ order statistics})\ \ {N^s}_c = \sum_{t=1}^{L} \gamma_t(c) \tag{2}$$

$$(\ 1^{\text{st}} \text{ order statistics})\ \ {F^s}_c = \sum_{t=1}^{L} \gamma_t(c) Y_t \tag{3}$$

where $\gamma_t(c)$ is the posterior probability of Gaussian component c of UBM for frame t and Y_t is the MFCC feature vector at frame t.

Using the statistical supervectors of songs in training set, we learn the $\mathbf{T}$ matrix via an Expectation Maximization (EM) algorithm: E-step, computes the probability of $P(w|X)$ where X is the given song and w is its i-vector. M-step, optimizes $\mathbf{T}$ by updating the following equation:

$$w = (\mathbf{I} + \mathbf{T}^t \mathbf{\Sigma}^{-1} N(s) \mathbf{T})^{-1} \cdot \mathbf{T}^t \mathbf{\Sigma}^{-1} F(s) \tag{4}$$

where $N(s)$ and $F(s)$ are diagonal matrices with ${N^s}_c.I$ and ${F^s}_c.I$ on diameter and N_s and F_s are 0^{th} and 1^{st} order statistical supervectors of song s. $\mathbf{\Sigma}$ is the diagonal covariance matrix, estimated during factor analysis training. The actual computation of an i-vector w for a given song s can be done using (4) after training $\mathbf{T}$. More information about the training procedure of $\mathbf{T}$ can be found in [4, 15].

4. MAP ESTIMATION OF A CLEAN I-VECTOR

In cases where only clean songs are available for training but at testing the observations are noisy, the best way to improve the performance of clean models encountered with noisy data would be to have an estimation of how clean data looks like. In an i-vector based approach, this estimation is done in the i-vector space by estimating clean i-vectors given noisy i-vectors.

This section describes a solution for music artist recognition in noisy environments which uses a state-of-the-art i-vector based system with an extra estimation step. Our method benefits from a MAP estimation of clean i-vectors given noisy i-vectors that was proposed in i-MAP method [16] for speaker verification applications. The estimation step is applied after i-vector extraction, as it is

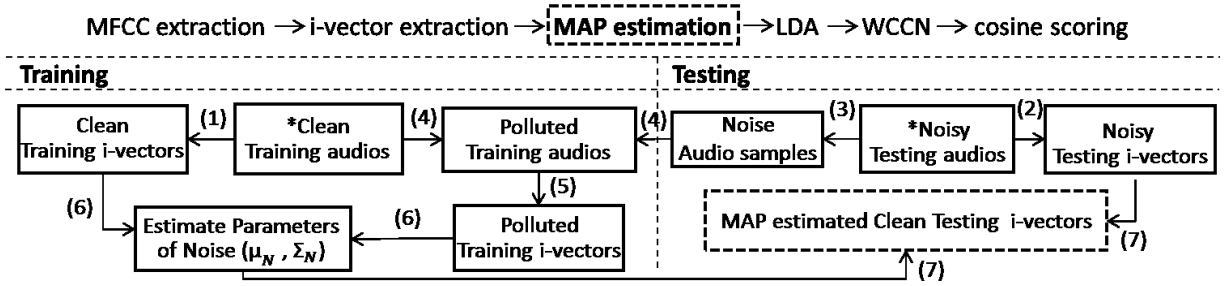


Figure 1. Block diagram of the estimation step in the proposed artist recognition method. The blocks with an asterisk (∗) indicate the training and testing audio data as the starting point of the diagram.

shown on the top of Figure 1. The resulting estimated i-vectors are used for inter-class compensation and scoring. The estimation step in i-MAP consists of: a) detecting noise using a Voice Activity Detector (VAD), b) synthesizing polluted data, c) estimating the mean and covariance of noise in i-vector space, and finally d) estimating clean i-vectors given noisy i-vectors. To estimate a clean i-vector given a noisy i-vector, i-MAP uses an estimation of *the mean and covariance of noise* in i-vector space by using noise audio samples detected in the noisy testing audios.

Based on i-MAP, the estimation step used in our proposed artist recognition method is described as follows.

Considering two sets of clean and noisy i-vectors, we define the following random variables:

- X: corresponding to the clean i-vectors
- Y: corresponding to the noisy i-vectors

And as i-vectors are normally distributed, we define:

$$X \sim \mathcal{N}(\mu_X, \Sigma_X) \tag{5}$$

$$Y \sim \mathcal{N}(\mu_Y, \Sigma_Y) \tag{6}$$

We denote the probability density functions (PDF) of X and Y by $f(X)$ and $f(Y)$ with the parameters of (μ_X, Σ_X) and (μ_Y, Σ_Y) as mean and covariance of clean and noisy i-vectors, respectively.

We now consider $f(X|Y_0)$ as the conditional PDF of clean i-vectors given a noisy i-vector Y_0. By Bayes' rule,

$$f(X|Y_0) = \frac{f(Y_0|X)f(X)}{f(Y_0)} \tag{7}$$

Using a MAP estimator, a clean i-vector $\widehat{X_0}$ can be estimated by maximizing $f(X|Y_0)$:

$$\widehat{X_0} = \underset{X}{\operatorname{argmax}}\{f(X|Y_0)\} \tag{8}$$

using (7) and taking $\ln$ we solve:

$$\frac{\partial}{\partial X}\{\ln f(Y_0|X) + \ln f(X)\} = 0 \tag{9}$$

The solution of (9) would provide an estimation of clean i-vector $\hat{X}_0$ from noisy i-vector Y_0.

4.1 Clean i-vector estimation

In i-MAP [2] it is assumed that when the noise is additive in the speech signal, the noise will be also additive in i-vector space. We keep this assumption and thus, the noise model defines as:

$$Y = X + N \tag{10}$$

where N is a random variable corresponds to noise in i-vector space:

$$N \sim \mathcal{N}(\mu_N, \Sigma_N) \tag{11}$$

where (μ_N, Σ_N) are parameters of noise in i-vector space.

Also, the Gaussian conditional PDF $f(Y_0|X)$ is defined as:

$$f(Y_0|X) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma_N|^{\frac{1}{2}}} e^{-\frac{1}{2}(Y_0-X-\mu_N)^t {\Sigma_N}^{-1}(Y_0-X-\mu_N)} \tag{12}$$

and Gaussian PDF $f(X)$ is:

$$f(X) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma_X|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu_X)^t {\Sigma_X}^{-1}(X-\mu_X)} \tag{13}$$

where (μ_X, Σ_X) are parameters of clean i-vectors, and (μ_N, Σ_N) are parameters of noise in i-vector space. Since $f(Y_0|X)$ and $f(X)$ are Gaussian, the resulting estimate of $f(X|Y_0)$ is also a valid Gaussian PDF as discussed in [22].

Now, by replacing $f(Y_0|X)$ and $f(X)$ by (12) and (13) in (9) and solving it we will have:

$$\widehat{X_0} = ({\Sigma_N}^{-1} + \Sigma_X^{-1})^{-1}(\Sigma_N^{-1}(Y_0 - \mu_N) + \Sigma_X^{-1}\mu_X) \tag{14}$$

$\widehat{X_0}$ is the MAP estimation of a clean i-vector given a noisy i-vector Y_0.

4.2 Parameters of Noise in I-vector Space

To use the MAP estimation provided in (14), we need an estimation of the parameters of clean i-vectors μ_X, Σ_X (which can be estimated from clean training i-vectors [2]) and parameters of noise in i-vector space μ_N, Σ_N.

[2] We use the term clean training audios (e.g. clean training songs) to address the audio data in training set that does not contain any noise. Noisy training audios (e.g. noisy testing songs) indicates audio data in testing set confronted with noise. The word polluted training audios (e.g. polluted training songs) indicates the data that are synthesized from clean

The parameters of clean training i-vectors (μ_X, Σ_X) can be learned by computing the mean and covariance matrix of clean training i-vectors. To learn the parameters of noise in i-vector space, we follow a similar procedure suggested in i-MAP (step numbers can be found in Figure 1): a) we extract clean training i-vectors from clean training songs and noisy testing i-vectors from noisy testing songs (steps 1 and 2). b) we detect noise audio samples in noisy testing songs (step 3). c) we use the noise audio samples detected in step 3 to synthesize polluted training songs from clean training songs (step 4). d) from polluted training songs, we extract a set of i-vectors known as polluted training i-vectors (step 5). e) using clean training i-vectors and polluted training i-vectors, we estimate the parameters of noise in i-vector space (step 6). f) now given noisy testing i-vectors, by having the parameters of noise in i-vector space, we estimate clean testing i-vectors via MAP estimation (step 7). The clean testing i-vectors estimated in step 7 are used for testing experiments in the proposed method.

In step 3, we use a noise detector which has the following steps: We first apply a windowing of 32 ms on the song's audio signal. Then for each window, we calculate the energy of the signal. Using a fixed threshold in energy of each window, we look at the beginning and ending area of each song to detect the areas with lower energy in noisy testing songs. We keep these areas as a set of noise audio samples. These samples are low-activity and assumed to be the examples of the additive noise that we are dealing with. This step provides the short sample of noise (a couple of seconds) in the use-case example described in Section 1.

Further, for each noise area detected in a noisy testing song, we estimate the SNR of the song and the noise sample. We select a limited number of noise areas with longer durations [3]. By adding the selected noise areas (noise audio samples) to clean training songs with the estimated SNR, we synthesize another audio set from our clean training songs which we know as polluted training songs.

Estimating the parameters of noise in step 6 is as follows: After extracting i-vectors from polluted training songs, we compute noise in i-vector space for each song by subtracting a clean training i-vector X_i from polluted training i-vector of that specific song Y_i as follows:

$$N'_i = {Y'}_i - X_i \tag{15}$$

where ${Y'}_i$ is the polluted training i-vector extracted from polluted training song ${S^p}_i$ and X_i is the clean training i-vector extracted from clean training song ${S^c}_i$ and N'_i is the noise related to ${Y'}_i$ in i-vector space. Now the parameters of noise in i-vector space (μ_N, Σ_N) can be calculated by:

$$\mu_N = mean(N') \tag{16}$$

$$\Sigma_N = cov(N') \tag{17}$$

where

$$N' = \{N'_i \mid i = 1, ..., n\} \tag{18}$$

and n is the number of training songs.

To use the proposed method in an adaptive way, we only need new audio samples of noise (which in our use-case we assumed the mobile app can provide, also described a feasible solution in step 3 about how to prepare them) to create a new set of polluted i-vectors to update the parameters of noise in the i-vector space. When the noise is changed, our parameters (μ_N, Σ_N) can also be updated to that noise.

5. EXPERIMENTS

5.1 I-vector Extractor

Our i-vector extractor consists of a UBM with 1024 Gaussian components. This UBM is trained on all the MFCCs of the clean training songs in each fold. The 0^{th} and 1^{st} order statistics (also known as statistical supervectors) are calculated for each song from MFCC features of the song using the UBM. The i-vector space matrix ($\mathbf{T}$) is learned from the statistical supervectors, via an Expectation Maximization (EM) algorithm described in [4, 15] where $\mathbf{T}$ is initialized from random and i-vector space dimensionality is set to 400. Using $\mathbf{T}$ matrix, 400 dimensional i-vectors are extracted for both training and testing set. All the i-vector extraction procedure is done unsupervised. We use 20-dimensional MFCCs in all of our i-vector based systems, extracted with RASTAMAT [10] toolbox with the same configuration as used in [7, 11]. The i-vector space matrix ($\mathbf{T}$) is trained using MSR identity toolbox [26].

5.2 Inter-class Compensation and Scoring

As we described in Section 3, i-vectors contain both artist and song variability. To reduce the song variability in i-vector space, multiple inter-class compensation methods such as length normalization, LDA and WCCN are found effective [4, 12]. Our i-vector inter-class compensation consists of 3 modules: 1) length-normalization, 2) LDA and 3) WCCN. For the scoring, we use a simple cosine scoring approach as detailed in [3].

Length of i-vectors causes negative effects in i-vector space [3, 12]. To discard these effects, we normalize the length of i-vectors by dividing each i-vector by its length. Thus, both training and testing i-vectors are first length normalized [12]. Using the resulting clean training i-vectors, a LDA projection matrix $\mathbf{V}$ is trained and both training and testing i-vectors are projected using $\mathbf{V}$. Then the resulting clean training i-vectors are length normalized again and then used to train a WCCN projection matrix $\mathbf{B}$. The WCCN matrix is used to project both training and testing i-vectors resulted from the LDA step. The final WCCN-projected i-vectors are used for cosine scoring. For each testing i-vector, a similarity score is calculated for each class separately. These scores are calculated given

training audios in training set, using audio samples of noise detected in noisy testing set. Noisy testing i-vectors are i-vectors extracted from noisy audios in testing set, polluted training i-vectors are i-vectors extracted from polluted training audios, and clean training i-vectors are i-vectors extracted from clean training audios. Clean testing i-vectors are estimated via MAP from noisy testing i-vectors.

[3] These noise areas are usually very short in time (a couple of seconds).

the model i-vectors that are computed by averaging LDA-WCCN projected clean training i-vectors for each class. And finally, the class with the maximum score is chosen as the predicted label for the testing i-vector.

5.3 Within-Class Covariance Normalization (WCCN)

Within-Class Covariance Normalization (WCCN) provides an effective compensation that can be used with cosine scoring. After i-vectors are length normalized and projected by LDA, they are encountered with WCCN projection matrix. WCCN scales the i-vector space in the opposite direction of its inter-class covariance matrix, so that directions of intra-artist variability are improved for i-vector scoring. The within-class covariance is estimated using clean training i-vectors as follows:

$$\mathbf{W} = \frac{1}{A}\sum_{a=1}^{\alpha}\frac{1}{n_a}\sum_{i=1}^{n_a}(w^a{}_i - \overline{w^a})(w^a{}_i - \overline{w^a})^t \qquad (19)$$

where $\overline{w^a} = \frac{1}{n_a}\sum_{i=1}^{n_a} w^a{}_i$ is the mean of the LDA projected i-vectors for each artist α. A is the total number of artists, and n_a is the number of songs of each artist α in training set. We use the inverse of the $\mathbf{W}$ matrix to normalize the direction of the projected i-vectors. WCCN projection matrix $\mathbf{B}$ can be estimated such that:

$$\mathbf{B}\mathbf{B}^t = \mathbf{W}^{-1} \qquad (20)$$

5.4 Cosine Scoring

In the i-vector space, a simple cosine scoring has been successfully used to compare two i-vectors [3]. Given a length normalized, LDA and WCCN projected i-vector w_i from an unknown artist, cosine score for artist α is defined as:

$$score(\overline{w_\alpha}, w_i) = \frac{(\overline{w_\alpha})^t . w_i}{\|\overline{w_\alpha}\| \, . \, \|w_i\|} \qquad (21)$$

where $\overline{w_\alpha}$ represents the mean i-vector of artist α, calculated by averaging all the length normalized, LDA and WCCN projected clean training i-vectors extracted from all the songs of artist α. $score(\overline{w_\alpha}, w_i)$ represents the cosine score of testing i-vector w_i for artist α. To predict the artist label for i-vector w_i, the artist with the highest cosine score is chosen as the label for w_i.

5.5 Dataset

In our experiments, we used Artist20 dataset [11], a freely available 20-class artist recognition corpus which consists of 1413 mp3 songs from 20 different artists mostly in pop and rock genres. The dataset is composed of six albums from each of 20 artists. A 6-fold cross validation is also provided with the dataset which is used in all of our experiments. In each fold, 5 out of 6 albums from all the artists were used for training and the rest were used for testing.

For our experiments with noisy data, we synthesized 12 (4×3) different noisy sets from Artist20 dataset with 4 different kinds of additive noise (festival, humming, pink, pub environment) of 3 different SNRs (5db, 10 db and 20 db), by applying the noise to all the songs. For applying the noise, the open-source Audio Degradation Toolbox (ADT) [23] is used.

For all the experiments (except IVEC-CLN and EBLF-CLN), the models are trained on training folds of clean dataset and tested on the testing fold of noisy dataset. For IVEC-CLN and EBLF-CLN experiments, models are trained on training folds of clean dataset and tested on testing fold of clean dataset.

We found noise samples of festival noise in the FreeSound repository [4]. The festival noise sample is an audio recording from a live performance during a festival with a lot of cheering sounds and human speaking in loudspeaker. This noise example is available upon request. For the other additive noises (pub environment, pink-noise, humming) the noise samples provided in ADT are used. The pub environment noise sample, recorded in a crowded pub, and the humming noise is recorded from a damaged speaker.

5.6 Evaluation

To evaluate the performance of different methods dealing with different kinds of noise, the averaged Fmeasure over all the classes is used [5]. The reported results in Table 1 show the mean of the averaged Fmeasures over 6-folds of our cross-validation, $\pm$ the standard deviation (std) of the averaged Fmeasures over folds. To examine the statistical significance, a t-test is applied for each sets of experiments separately, comparing the Fmeasures of 6 folds between the proposed method and each of the baselines (for example: festival noise with 3 db SNR, comparing IVEC-NSY and EBLF-NSY). Each set of experiments for a specific kind of noise with a certain SNR is done independently.

5.7 Baseline Methods

We compare the performance of our proposed method with two baselines. The first baseline is a state-of-the-art standard i-vector based artist recognition system known as *IVEC-NSY*. The reason we chose this baseline is to show the improvements by adding the estimation step to this baseline. We extract 400 dimensional i-vectors using 20-dimensional MFCCs and a 1024 components GMM as UBM. Then we apply length normalization, LDA and WCCN and further apply the cosine scoring to predict the labels.

The second baseline (*EBLF-NSY*) uses an extended version of Block-Level features [28] (EBLF) used in [27]. EBLF are the winner of multiple tasks in MIREX challenge [6] such as music similarity and genre classification and provide a set of 8 song-level descriptors which represent different characteristics for a song. These 8 descriptors contain a good variety of features such as rhythm and

[4] http://freesond.org

[5] Since the number of songs from each artist are more or less the same (6 albums), the averaged Fmeasure seems to be a good measurement for our multi-class artist recognition task.

[6] Annual Music Information Retrieval eXchange (MIREX). More information is available at: http://www.music-ir.org

timbre in a song. Since these features are frequently used for multiple purposes in MIR, we chose them as our second baseline to show how they perform in a noisy environment. For classification, a WEKA [13] implementation of SMO support vector machine is used as described in [27]. The SVM model in this baseline is trained on features of clean training songs and tested on features of noisy testing songs.

To demonstrate the maximum power of our baselines on clean data, we also provided the performance of these methods, dealing with only clean data. In these specific experiments, we train and test the baselines using clean data and the results can be found in the description of Table 1 as *IVEC-CLN* and *EBLF-CLN*.

The performance of *EBLF-CLN* is comparable with the baselines in [7] which compares different approaches for music artist recognition and therefore is a competitive method to be used as a baseline. Also, the performance of *IVEC-CLN* is comparable with the the best results achieved using a state-of-the-art i-vector based artist recognition system reported in [7] (which are known to be the best artist recognition results on Artist20 dataset published so far). Both *IVEC-CLN* and [7] use similar i-vector extractors. The difference between *IVEC-CLN* and [7] is in the inter-class compensation and scoring. We used LDA followed by WCCN as inter-class compensation and a simple cosine scoring, while in [7] only LDA is used as inter-class compensation and the best performance achieved with a Discriminant Analysis classifier.

5.8 The Proposed Method

Our proposed method is shown in Table 1 as *IVEC-MAP*. All the i-vector extraction (UBM,$\mathbf{T}$), inter-class compensation (LDA,WCCN) and scoring models are only trained with clean training data and the only extra information used in the proposed method compared to the baselines, is the mean and covariance of noise in i-vector space. The number of at least 500 i-vectors are needed to estimate the parameters of noise as reported in [16] in a speaker verification scenario. We used all the polluted training i-vectors for parameter estimation, since our training set is not very big (~ 1100 songs).

5.9 Results and Discussion

By looking at Table 1 it can be seen that the proposed method outperformed the baselines in all 12 cases of 4 different additive noises with 3 different SNRs. Having a closer look at the results suggests the IVEC-NSY baseline performed much better and more robust than EBLF-NSY.By looking at the results dealing with noises of different SNR levels, it can be seen that as expected the lower the SNR (more noise), the lower the performance of our baselines are. Unlike the baselines, the change in SNR does not affect the performance of our proposed method significantly in festival and humming noises. Considering the standard deviation of the averaged Fmeasures for all the 6-folds, results suggest that the proposed method is always higher than 1 std from the performance of EBLF-NSY in all the noises with all different SNRs. When the noise is in its highest level (SNR=5 db) the proposed method's Fmeasure is higher than IVEC-NSY by 1 std. On average, the proposed method achieved the relative averaged Fmeasure of 28.41, 12.99 and 7.21 percentage points higher than IVEC-NSY encountering additive noises with 5, 10 and 20 db SNR, respectively. Applying a *t-test hypothesis testing* on the averaged Fmeasures for different folds to examine the performance between the proposed method and the baselines (IVEC-NSY and EBLF-NSY), shows that the test rejects the null hypothesis at 5% significance level for all the experiments and our results are statistically significant.

Averaged Fmeasure (%)

snr	nois.	IVEC-NSY	EBLF-NSY	IVEC-MAP
5 db	fest.	68.22±8.85	19.01±23.31	**81.08±7.68**
	hum.	75.28±8.14	5.21±5.15	**82.56±6.82**
	pink	60.44±10.27	6.64±11.58	**74.88±6.67**
	pub	44.11±8.13	14.01±22.27	**71.12±8.09**
10 db	fest.	74.27±8.97	24.32±27.39	**81.91±7.55**
	hum.	77.15±8.97	25.71±26.5	**82.64±7.24**
	pink	68.58±8.22	11.89±16.38	**78.05±7.2**
	pub	66.15±9.04	22.7±25.91	**79.87±7.31**
20 db	fest.	77.28±7.6	36.12±26.76	**81.89±7.47**
	hum.	77.32±7.43	36.89±25.3	**82.86±7.1**
	pink	74.57±7.63	13.93±21.31	**80.54±7.53**
	pub	76.74±7.8	26.17±29.05	**82.63±7.2**

Table 1. Comparison of artist recognition performance of different methods on Artist20 dataset dealing with different kinds and levels of additive noise. The numbers indicate the averaged Fmeasure (as described in Section 5.6) with the standard deviation over all the folds. The performance of the baselines IVEC-CLN and EBLF-CLN on clean data are 83.73±7.58 and 72.26±7.42 respectively.

6. CONCLUSION

In this paper, we proposed a noise-robust artist recognition system using i-vector features and a MAP estimation of clean i-vectors given noisy i-vectors. Our method outperformed the state-of-the-art standard i-vector system and EBLF, also showed a stable performance dealing with multiple kinds of additive noise with different SNRs. We showed that by adding an estimation step to a standard i-vector based artist recognition system, the performance in noisy environments can be significantly improved.

7. ACKNOWLEDGMENTS

We would like to acknowledge the help by Mohamad Hasan Bahari of KU Leuven University to this work. We also appreciate helpful suggestions of Waad Ben-Kheder from University of Avignon. This work was supported by the Austrian Science Fund (FWF) under grant no. Z159 (Wittgenstein Award).

8. REFERENCES

[1] Mohamad Hasan Bahari, Rahim Saeidi, David Van Leeuwen, et al. Accent recognition using i-vector, gaussian mean supervector and gaussian posterior probability supervector for spontaneous telephone speech. In *ICASSP*. IEEE, 2013.

[2] Waad Ben Kheder, Driss Matrouf, Jean-François Bonastre, Moez Ajili, and Pierre-Michel Bousquet. Additive noise compensation in the i-vector space for speaker recognition. In *ICASSP*. IEEE, 2015.

[3] Najim Dehak, Reda Dehak, James R Glass, Douglas A Reynolds, and Patrick Kenny. Cosine similarity scoring without score normalization techniques. In *Odyssey*, page 15, 2010.

[4] Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 2011.

[5] Najim Dehak, Pedro A Torres-Carrasquillo, Douglas A Reynolds, and Reda Dehak. Language recognition via i-vectors and dimensionality reduction. In *INTERSPEECH*. Citeseer, 2011.

[6] Hamid Eghbal-zadeh, Bernhard Lehner, Markus Schedl, and Gerhard Widmer. I-vectors for timbre-based music similarity and music artist classification. In *ISMIR*, 2015.

[7] Hamid Eghbal-zadeh, Markus Schedl, and Gerhard Widmer. Timbral modeling for music artist recognition using i-vectors. In *EUSIPCO*, 2015.

[8] A El-Solh, A Cuhadar, and RA Goubran. Evaluation of speech enhancement techniques for speaker identification in noisy environments. In *ISMW*. IEEE, 2007.

[9] Benjamin Elizalde, Howard Lei, and Gerald Friedland. An i-vector representation of acoustic environments for audio-based video event detection on user generated content. In *ISM*. IEEE, 2013.

[10] Daniel PW Ellis. PLP and RASTA (and MFCC, and inversion) in Matlab, 2005. online web resource.

[11] Daniel PW Ellis. Classifying music audio with timbral and chroma features. In *ISMIR*, 2007.

[12] Daniel Garcia-Romero and Carol Y Espy-Wilson. Analysis of i-vector length normalization in speaker recognition systems. In *INTERSPEECH*, 2011.

[13] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 2009.

[14] Cemal Hanilçi, Tomi Kinnunen, Rahim Saeidi, Jouni Pohjalainen, Paavo Alku, F Ertaş, Johan Sandberg, and Maria Hansson-Sandsten. Comparing spectrum estimators in speaker verification under additive noise degradation. In *ICASSP*. IEEE, 2012.

[15] Patrick Kenny. Joint factor analysis of speaker and session variability: Theory and algorithms. *CRIM, Montreal,(Report) CRIM-06/08-13*, 2005.

[16] Waad Ben Kheder, Driss Matrouf, Pierre-Michel Bousquet, Jean-François Bonastre, and Moez Ajili. Robust speaker recognition using map estimation of additive noise in i-vectors space. In *Statistical Language and Speech Processing*. Springer, 2014.

[17] Anna M Kruspe. Improving singing language identification through i-vector extraction. In *DAFx*, 2011.

[18] Yun Lei, Lukas Burget, Luciana Ferrer, Martin Graciarena, and Nicolas Scheffer. Towards noise-robust speaker recognition using probabilistic linear discriminant analysis. In *ICASSP*. IEEE, 2012.

[19] Yun Lei, Lukas Burget, and Nicolas Scheffer. A noise robust i-vector extractor using vector taylor series for speaker recognition. In *ICASSP*. IEEE, 2013.

[20] Yun Lei, Moray McLaren, Luciana Ferrer, and Nicolas Scheffer. Simplified vts-based i-vector extraction in noise-robust speaker recognition. In *ICASSP*. IEEE, 2014.

[21] D Martinez, Lukas Burget, Themos Stafylakis, Yun Lei, Patrick Kenny, and Eduardo Lleida. Unscented transform for ivector-based noisy speaker recognition. In *ICASSP*. IEEE, 2014.

[22] Driss Matrouf, Waad Ben Kheder, Jean-François Bonastre, Moez Ajili, and Pierre-Michel Bousquet. Dealing with additive noise in speaker recognition systems based on i-vector approach. In *EUSIPCO*. IEEE, 2015.

[23] Matthias Mauch and Sebastian Ewert. The audio degradation toolbox and its application to robustness evaluation. In *ISMIR*, 2013.

[24] Simon JD Prince and James H Elder. Probabilistic linear discriminant analysis for inferences about identity. In *Computer Vision, ICCV*. IEEE, 2007.

[25] Seyed Omid Sadjadi and John HL Hansen. Assessment of single-channel speech enhancement techniques for speaker identification under mismatched conditions. In *INTERSPEECH*, 2010.

[26] Seyed Omid Sadjadi, Malcolm Slaney, and Larry Heck. Msr identity toolbox-a matlab toolbox for speaker recognition research. *Microsoft CSRC*, 2013.

[27] Klaus Seyerlehner, Markus Schedl, Tim Pohle, and Peter Knees. Using block-level features for genre classification, tag classification and music similarity estimation. *MIREX*, 2010.

[28] Klaus Seyerlehner, Gerhard Widmer, and Tim Pohle. Fusing block-level features for music similarity estimation. In *DAFx*, 2010.

[29] Rui Xia and Yang Liu. Using i-vector space model for emotion recognition. In *INTERSPEECH*, 2012.

ON THE USE OF NOTE ONSETS FOR IMPROVED LYRICS-TO-AUDIO ALIGNMENT IN TURKISH MAKAM MUSIC

Georgi Dzhambazov Ajay Srinivasamurthy
Sertan Şentürk Xavier Serra
Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
georgi.dzhambazov@upf.edu

ABSTRACT

Lyrics-to-audio alignment aims to automatically match given lyrics and musical audio. In this work we extend a state of the art approach for lyrics-to-audio alignment with information about note onsets. In particular, we consider the fact that transition to next lyrics syllable usually implies transition to a new musical note. To this end we formulate rules that guide the transition between consecutive phonemes when a note onset is present. These rules are incorporated into the transition matrix of a variable-time hidden Markov model (VTHMM) phonetic recognizer based on MFCCs. An estimated melodic contour is input to an automatic note transcription algorithm, from which the note onsets are derived. The proposed approach is evaluated on 12 a cappella audio recordings of Turkish Makam music using a phrase-level accuracy measure. Evaluation of the alignment is also presented on a polyphonic version of the dataset in order to assess how degradation in the extracted onsets affects performance. Results show that the proposed model outperforms a baseline approach unaware of onset transition rules. To the best of our knowledge, this is the one of the first approaches tackling lyrics tracking, which combines timbral features with a melodic feature in the alignment process itself.

1. INTRODUCTION

Lyrics are one of the most important aspects of vocal music. When a performance is heard, most listeners will follow the lyrics of the main vocal line. The goal of automatic lyrics-to-audio alignment is to generate a temporal relationship between lyrics and recorded singing. In this particular work, the goal is to detect the start and end times of every phrase (1-4 words) from lyrics.

In recent years there has been a substantial amount of work on the extraction of pitch of predominant singing voice from polyphonic music [18]. Some algorithms have been tailored to the music characteristics of a particular singing tradition [12]. This has paved the way to an increased accuracy of note transcription algorithms. One of the reasons for this is that a correctly detected melody contour is a fundamental precondition for note transcription. On the other hand, lyrics-to-audio alignment is a challenging task: to track the timbral characteristics of singing voice might not be straightforward [4]. An additional challenge is posed when accompanying instruments are present: their spectral peaks might overlap and occlude the spectral components of voice. Despite that, most work has focused on tracking the transitions from one phoneme to another only by timbral features [4, 14]. In fact, at a phoneme transition, in parallel to timbral change, a change of pitch or an articulation accent may be present, which contributes to the perception of a distinct vocal note onset . For example, a note onset occurs simultaneously with the first vowel in a syllable. This fact has been exploited successfully to enhance the naturalness of synthesized singing voice [21].

In this work we present a novel idea of how to extend a standard approach for lyrics-to-audio alignment by using automatically detected vocal note onsets as a complementary cue. We apply a state of the art note transcription method to obtain candidate note onsets. The proposed approach has been evaluated on time boundaries of short lyrics phrases on a cappella recordings from Turkish Makam music. An experiment on polyphonic audio reveals the potential of the approach for real-world applications.

2. RELATED WORK

2.1 Lyrics-to-audio alignment

The problem of lyrics-to-audio alignment has an inherent relation to the problem of text-to-speech alignment. For this reason most of current studies exploit an approach adopted from speech: building a model for each phoneme based on acoustic features [5, 14]. To model phoneme timbre usually mel frequency cepstral coefficients (MFCCs) are employed. A state of the art work following this approach [5] proposes a technique to adapt a phonetic recognizer trained on speech: the MFCC-based speech phoneme models are adapted to the specific acoustics of singing voice by means of Maximum Likelihood Linear Regression. Further, automatic segregation of the vocal line is performed, in order to reduce the spectral content of back-

© Georgi Dzhambazov, Ajay Srinivasamurthy, Sertan Şentürk , Xavier Serra . Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Georgi Dzhambazov, Ajay Srinivasamurthy, Sertan Şentürk , Xavier Serra . "On the use of note onsets for improved lyrics-to-audio alignment in Turkish Makam music", 17th International Society for Music Information Retrieval Conference, 2016.

ground instruments. In general, in this approach authors consider only models of phonetic timbre and are thus focused on making them more robust as a mean to improve performance.

Few works for tracking lyrics combine timbral features with other melodic characteristics. For example in [7] a system for automatic score-following of singing voice combines melodic and lyrics information: observation probabilities of pitch templates and vowel templates are fused to improve alignment. In [13] lyrics-to-audio alignment has been aided on a coarser level by chord-to-audio alignment, assuming chord annotations are available in a paired chord-lyrics format. However, to our knowledge, no work so far has employed note onsets as additional cue to alignment.

2.2 Automatic note segmentation

While the general problem of automatic music transcription has been long-investigated, automatic singing transcription has attracted the attention of MIR researchers only in recent years [6, 11, 15]. A fundamental part of singing transcription is automatic note segmentation. A probabilistic note event model, using a HMM trained on manual transcriptions is presented in [11]. The idea is that a note consists of different states representing its attack, sustain and decay phase. Then an onset is detected when the decoding path goes through an attack state of a new note.

A recent work on singing transcription with high onset accuracy has been developed for singing voice from the flamenco genre [12]. It consists of two stages: predominant vocal extraction and note transcription. As a primary step of the note transcription stage, notes are segmented by a set of onset detection functions based on pitch contour and volume characteristics, which take into account the peculiar for flamenco singing high degree of microtonal ornamentation.

3. PROPOSED APPROACH

A general overview of the proposed approach is presented in Figure 1. An audio recording and its lyrics are input. A variable time hidden Markov model (VTHMM), guided by phoneme transition rules, returns start and end timestamps of aligned words. For brevity in the rest of the paper our approach will be referred to as VTHMM.

First an audio recording is manually divided into segments corresponding to structural sections (e.g. verse, chorus) as indicated in a structural annotation, whereby instrumental-only sections are discarded. All further steps are performed on each audio segment. If we had used automatic segmentation instead, potential erroneous lyrics and features could have biased the comparison of a baseline system and VTHMM. As we focus on evaluating the effect of VTHMM, manual segmentation is preferred. In what follows each of the modules is described in details.

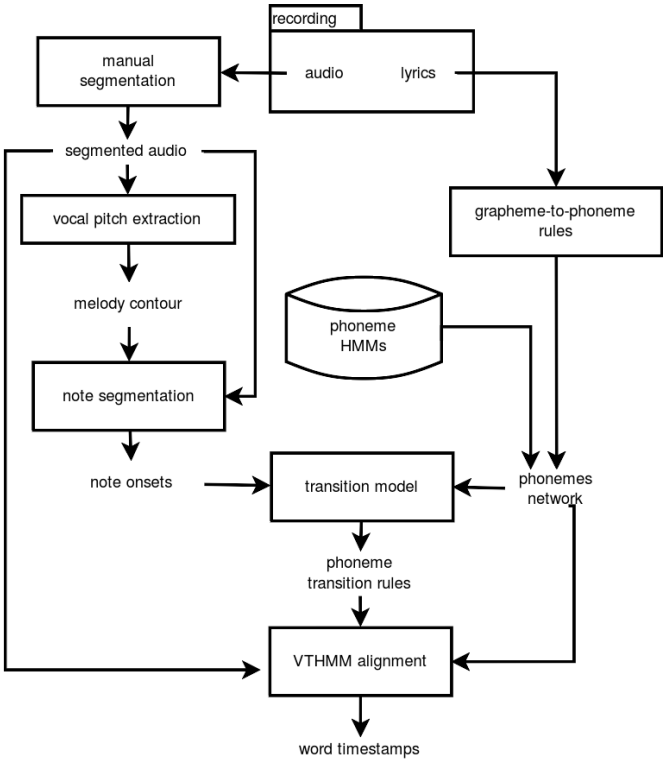


Figure 1. Overview of the modules of the proposed approach. One can see how phoneme transition rules are derived. Then together with the phonemes network and the features extracted from audio segments are input to the VTHMM alignment.

3.1 Vocal pitch extraction

To extract the melody contour of singing voice, we utilize a method that performs detection of vocal segments and in the same time pitch extraction for the detected segments [1]. It relies on the basic methodology of [19], but modifies the way in which the final melody contour is selected from a set of candidate contours, in order to reflect the specificities of Turkish Makam music: 1) It chooses a finer bin resolution of only 7.5 cents that approximately corresponds to the smallest noticeable change in Makam melodic scales. 2) Unlike the original methodology, it does not discard time intervals where the peaks of the pitch contours have relatively low magnitude. This accommodates time intervals at the end of the melodic phrases, where Makam singers might sing softer.

3.2 Note segmentation

In a next step, to obtain reliable estimate of singing note onsets, we adapt the automatic singing transcription method, developed for polyphonic flamenco recordings [12]. It has been designed to handle singing with high degree of vocal pitch ornamentation. We expect that this makes it suitable for material from Makam classical singing having heavily vibrato and melismas, too. We replace the original first stage predominant vocal extraction

method with the vocal pitch detection method described above.

The algorithm [12] considers two cases of onsets: interval onsets and steady pitch onsets. A Gaussian derivative filter detects interval onsets as long-term changes of the pitch contour, whereas steady-pitch onsets are inferred from pitch discontinuities. As in the current work phoneme transitions are modified only when onsets are present, we opt for increasing recall at the cost of losing precision. This is achieved by reducing the value of the parameter cF: the minimum output of the Gaussian filter. The extracted note onsets are converted into a binary onset activation at each frame $\Delta n_t = (0, 1)$. Recall rates of extracted note onsets are reported in Table 2.

3.3 Phoneme models

The formant frequencies of spoken phonemes can be induced from the spectral envelope of speech. To this end, we utilize the first 12 MFCCs and their delta to the previous time instant, extracted as described in [24]. For each phoneme a one-state HMM, for which a 9-mixture Gaussian distribution is fitted on the feature vector. The lyrics are expanded to phonemes based on grapheme-to-phoneme rules for Turkish [16, Table 1] and the trained HMMs are concatenated into a phonemes network. The phoneme set utilized has been developed for Turkish and is described in [16]. A HMM for silent pause *sp* is added at the end of each word, which is optional on decoding. This way it will appear in the detected sequence only if there is some non-vocal part or the singer makes a break for breathing.

3.4 Transition model

We utilize a transition matrix with time-dependent self-transition probabilities which falls in the general category of variable time HMM (VTHMM) [9]. For particular states, transitions are modified depending on the presence of time-adjacent note onset. Let t' be the timestamp of the closest to given time t onset $\Delta n_{t'} = 1$. Now the transition probability can be rewritten as

$$a_{ij}(t) = \begin{cases} a_{ij} - g(t,t')q, & R1 \text{ or } R3 \\ a_{ij} + g(t,t')q, & R2 \text{ or } R4 \end{cases} \quad (1)$$

$R1$ to $R4$ stand for phoneme transition rules, which are applied in the phonemes network by picking the states i and j for two consecutive phonemes. The term q is a constant whereas $g(t, t')$ is a weighting factor sampled from a normal distribution with its peak (mean) at t':

$$g(t,t') = \begin{cases} f(t;t',\sigma^2) \sim \mathcal{N}(t',\sigma^2), & |t-t'| \leq \sigma \\ 0 & else \end{cases} \quad (2)$$

Since singing voice onsets are regions in time, they span over multiple consecutive frames. To reflect that fact, $g(t, t')$ serves to smooth in time the influence of the discrete detected Δn_t, where σ has been selected to be 0.075 seconds. In this way an onset influences a region of 0.15 seconds - a threshold suggested for vocal onset detection evaluation by [6] and used in [12]. Furthermore, this allows to handle slight timestamp inaccuracies of the estimated note onsets.

3.4.1 Phoneme transition rules

Let V denote a vowel, C denote a consonant and L denote a vowel, liquid (LL, M, NN) or the semivowel Y. Rules $R1$ and $R2$ represent inter-syllable transition, e.g. phoneme i is followed by phoneme j from the following syllable:

$$\begin{array}{lll} R1: & i = V & j = \neg L \\ R2: & i = C & j = L \end{array} \quad (3)$$

For example, for rule $R2$ if a syllable ends in a consonant, a note onset imposes with high probability that a transition to the following syllable is done, provided that it starts with a vowel. Same rule applies if it starts with a liquid, according to the observation that pitch change takes place during a liquid preceding the vowel [21, timing of pitch change]. Rules $R3$ and $R4$ are for intra-syllabic phoneme patterns:

$$\begin{array}{lll} R3: & i = V & j = C \\ R4: & i = \neg L & j = V \end{array} \quad (4)$$

Essentially, if the current phoneme is vocal and the next is non-voiced (e.g. $R1$, $R3$), Eq. (1) discourages transition no next phoneme and encourages transition in the opposite cases. An example of $R4$ can be seen for the syllable KK-AA in Figure 2 where the note onset triggers the change to the vowel AA, opposed, for example, to onset at Y for the syllable Y-E-T. Note that these rules assume

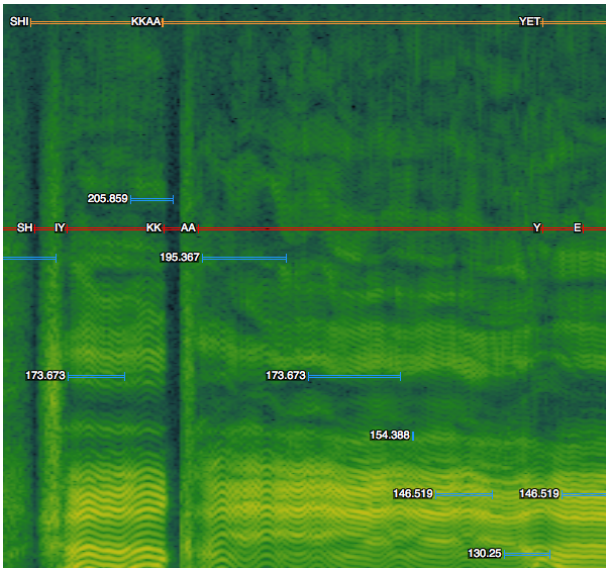


Figure 2. Ground truth annotation of syllables (in orange/top), phonemes (in red/middle) and notes (with blue/changing position). Audio excerpt corresponding to word şikayet with syllables SH-IY, KK-AA and Y-E-T.

total #sections	#phrases per section	#words per phrase
75	2 to 5	1 to 4

Table 1. Phrase and section statistics about the dataset.

that a syllable has one vowel, which is the case for Turkish [1] . The optional silent phoneme *sp* is handled as a special case: transition probability from any phoneme to *sp* is derived according to intra-syllable rules, and the one from any phoneme skipping to the phoneme following *sp* is derived according to inter-syllable rules.

3.4.2 Alignment

The most likely state sequence is found by means of a forced alignment Viterbi decoding.

$$\delta_t(j) = \max_{i \in (j,\, j-1)} \delta_{t-1}(i)\, a_{ij}(t)\, b_j(O_t) \qquad (5)$$

Here $b_j(O_t)$ is the observation probability for state i for feature vector O_t and $\delta_t(j)$ is the probability for the path with highest probability ending in state j at time t (complying with the notation of [17, III. B] [2] .

4. DATASET

The test dataset consists of 12 a cappella performances of 11 compositions with total duration of 19 minutes. The performances are drawn from *CompMusic* corpus of classical Turkish Makam repertoire with provided annotations of musical sections [23]. Solo vocal versions of the originals have been sung by professional singers, especially recorded for this study, due to the lack of appropriate a cappella material in this music tradition. A performance has been recorded in sync with the original recording, whereby instrumental sections are left as silence. This assures that the order, in which sections are performed, is kept the same. One of the contributions of this work is that we make available the annotated phrase boundaries [3] . A musical phrase spans 1 to 4 words depending on the duration of the words (as proposed in [10]). Table 1 presents statistics about phrases, while the total number of words in the dataset is 732.

Additionally, the singing voice for 6 recordings (with a total duration of 10 minutes) from the dataset has been annotated with MIDI notes complying to the musical score [4] . On annotation special care is taken to place the note onset on the time instant, at which the pitch becomes steady. Thus we avoid placing the onset on an unvoiced phoneme at the beginning of a syllable, which is assures rules R3 and R4 make sense (see Figure 2) [5] .

[1] Among one-vowel syllabic languages are also Japanese and to some extent Italian

[2] To encourage reproducibility of this research an efficient open-source implementation together with documentation is available at https://github.com/georgid/AlignmentDuration/tree/noteOnsets

[3] The audio and the annotations are available under a CC license at http://compmusic.upf.edu/turkish-makam-acapella-sections-dataset

[4] Creating the annotation is a time-consuming task, but we plan to annotate the whole dataset in the future

[5] Onset annotations are available at http://compmusic.upf.edu/node/233

4.1 Evaluation metric

Alignment is evaluated in terms of alignment accuracy as the percentage of duration of correctly aligned regions from total audio duration (see [5, Figure 9] for an example). A value of 100 means perfect matching of all phrase boundaries in the evaluated audio. Thus accuracy can be reported not only for an audio segment, but also on total for a recording, or as a total for all the recordings.

5. EXPERIMENTS

5.1 Experiment 1: alignment with oracle onsets

As a precursor to the following experiments, lyrics-to-audio alignment is run on these 6 recordings with manually annotated MIDI notes, which serve as an oracle for note onsets. This is done to test the general feasibility of the proposed model on the dataset, unbiased from errors in the note segmentation algorithm, and to set a glass-ceiling alignment accuracy. We have tested with different values of q from Eq. 1 achieving best accuracy of 83.5% at $q = 0.23$, which is used on all further reported experiments.

5.2 Experiment 2: recognition of phonemes

In general, the comparison to other lyrics alignment systems is not feasible, because there is no current work developed for Turkish language. However, to have an idea of how adequate the trained phoneme HMMs are, we have annotated phoneme boundaries for some excerpts of total length of 6 minutes. In [8] phonemes are recognized in a cappella singing with no lyrics given in advance. With phoneme MFCC-based HMMs - the same as our modeling setting - a phoneme recall rate of 44% is reported. Even though for forced alignment the recognition of phonemes is relatively easier, given that they are ordered in a sequence, we measured lower overall phoneme recall of 37%. This indicates that our phoneme models trained only on speech might not be the most optimal choice.

5.3 Experiment 3: comparison to a baseline

As a baseline we conduct alignment of the test dataset with unaffected phoneme transition probabilities, e.g. setting all $\Delta n_t = 0$, which resulted in alignment accuracy of 70.2%. Further, we measured the impact of the note segmentation module (introduced in Section 3.2), varying onset detection recall by changing the minimum output of the Gaussian filter (controlled by the parameter cF). Table 2 summarizes the alignment accuracy with VTHMM depending on recall. On a cappella best improvement over the baseline is achieved at recall of 72.3% (at $cF = 3.5$). This is somewhat lower than the best recall of 81-84% achieved for flamenco [12]. Setting recall higher than that degraded performance because there are too many false alarms, resulting in forcing false transitions.

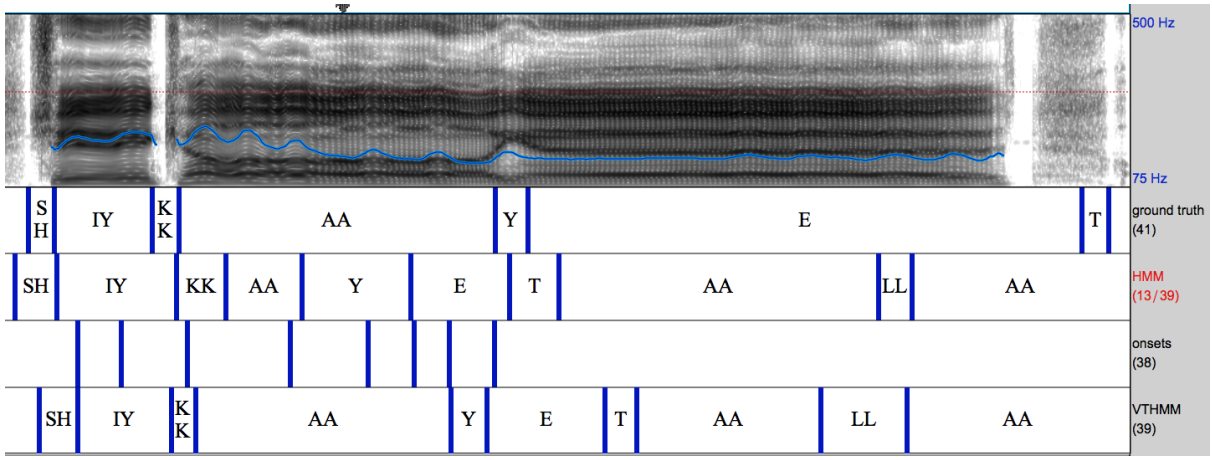


Figure 3. Example of boundaries of phonemes for the word şikayet (SH-IY-KK-AA-Y-E-T): *on top*: spectrum and pitch; *then from top to bottom*: ground truth boundaries, phonemes detected with HMM, detected onsets, phonemes detected with VTHMM; (excerpt from the recording 'Kimseye etmem şikayet' by Bekir Unluater).

	cF	5	4.5	4.0	3.5	3.0
a cappella	**OR**	57.2	59.7	66.8	72.3	73.2
	AA	71.1	73.3	74.5	**75.7**	72.0
polyphonic	**OR**	52.8	58.2	65.9	66.2	68.4
	AA	61.2	63.3	**64.8**	64.6	60.3

Table 2. VTHMM performance on a cappella and polyphonic audio, depending on onset detection recall (OR). Alignment accuracy (AA) is reported as a total for all the recordings.

Figure 3 allows a glance at the level of detected phonemes: the baseline HMM switches to the following phoneme after some amount of time, similar for all phonemes. One reason for this might be that the waiting time in a state in HMMs with a fixed transition matrix cannot be randomly long [25]. In contrast, for VTHMM the presence of note onsets at vowels activates rules $R1$ or $R3$, which allows waiting in the same state longer, as there are more onsets (for example AA from the word SH-IY-KK-AA-Y-E-T has five associated onsets). We chose to modify cF because setting it to lower values increases the recall of *interval onsets*: Often in our dataset several consecutive notes with different pitch correspond to the same vowel. In fact, it is characteristic of Turkish classical music that a single syllable may have a complex melodic progression spanning many notes (up to 12 in our dataset) [3]. However, for cases of vowels held long on same pitch, conceptually VTHMM is not capable of bringing any benefit. This is illustrated in Figure 3 by the prematurely detected end boundary of E from the word SH-IY-KK-AA-Y-E-T.

In addition to that, we examined alignment accuracy per recording (Figure 4). It can be observed that VTHMM performs consistently better than the baseline HMM (with some exceptions of where accuracy is close).

6. EXTENSION TO POLYPHONIC MATERIAL

To test the feasibility of the proposed approach on polyphonic material, the alignment is evaluated on the original versions of the recordings in the dataset. Typical for Turkish Makam is that vocal and accompanying instruments follow the same melodic contour in their corresponding registers, with slight melodic variations. However, the vocal line usually has melodic predominance. This special type of polyphonic musical interaction is termed heterophony [3]. In the dataset used in this study, a singer is accompanied by one to several string instruments.

We applied the vocal pitch extraction and note segmentation methods directly, since both are developed for singing voice in a setting that has heterophonic characteristics. However, instrumental spectral peaks deteriorate significantly the shape of the vocal spectrum. To attenuate the negative influence of instrumental spectrum, a vocal resynthesis step is necessary.

6.1 Vocal resynthesis

For the regions with predominant voice, the vocal content is resynthesized as separate vocal part. Resynthesis is conducted based on the harmonic model of [20]: Based on the extracted predominant pitch (see Section 3.1) and a set of peaks from the original spectrum, the harmonic partials of the predominant voice are selected and resynthesized. Then MFCCs are extracted from the resynthesized vocal part as if it were monophonic singing. This is a viable step, because the harmonic partials preserve well the overall spectral shape of the singing voice, including the formant frequencies, which encode the phoneme identities [22] [6]. More details and examples of the resynthesis can be found in previous work, which showed that the application of a harmonic model is suitable for aligning lyrics in Makam music [2]. A conceptually similar resynthesis

[6] The resynthesis allowed us to verify that vocals are intelligible despite some distortions from overlap with instrumental harmonic partials

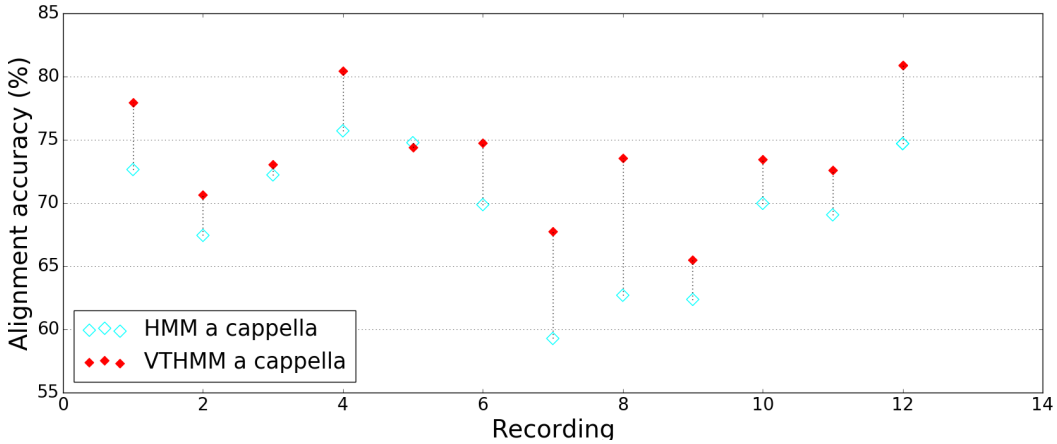


Figure 4. Comparison between results for VTHMM and baseline HMM on a cappella.

step is an established part also in current methods for alignment of lyrics in polyphonic Western pop music [5, 14].

6.2 Experiment 4: comparison of a cappella and polyphonic

The onset recall rates on polyphonic material after note segmentation are not much worse than a cappella as presented in Table 2. Even though the degree of degradation in onset detection is slight, degradation in alignment accuracy is significant. This can be attributed most probably to the fact that our MFCC-based models are not very discriminative and get confused by artifacts, induced from other instruments on resynthesis. However, applying VTHMM on polyphonic recordings still improves over the baseline (see Table 3). Note that the margin in accuracy between the baseline and the oracle glass ceiling is only about 6%, which is about twice much in the case of solo voice.

	HMM	VTHMM	oracle
a cappella	70.2	75.7	83.5
polyphonic	61.5	64.8	67.1

Table 3. Comparison of accuracy of baseline HMM, VTHMM and, VTHMM with oracle onsets. VTHMM shown are the best accuracies reported in Table 2. Alignment accuracy is reported on total for all recordings.

7. CONCLUSION

In this work we evaluated the behavior of a HMM-based phonetic recognizer for lyrics-to-audio alignment in two settings: with and without considering singing voice onsets as additional cue. Compared to existing work on lyrics alignment, this is, to our knowledge, the first attempt to include onsets of the vocal melody in the inference process. Updating transition probabilities according to onset-aware phoneme transition rules resulted in an improvement of absolute 5.5 percent for aligning phrases of solo voice from Turkish Makam recordings. In particular, due to rules discouraging premature transition, the states of sustained vowels could have longer durations.

Alignment on same data with instrumental accompaniment brought also some small improvement over a baseline with no onset modeling. Having onset detection performing not substantially worse than a cappella indicates that improving the phoneme acoustic models in the future could probably lead to even more significant improvement.

A practical limitation of the current alignment system is the prerequisite for manual structural segmentation, which we plan to automate in the future.

Acknowledgements We thank Nadine Kroher for providing help with running the note segmentation module. This work is partly supported by the European Research Council under the European Union's Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583) and partly by the AGAUR research grant. We acknowledge as well financial support from the Spanish Ministry of Economy and Competitiveness, through the "María de Maeztu" Programme for Centres/Units of Excellence in R&D" (MDM-2015-0502)

8. REFERENCES

[1] Hasan Sercan Atlı, Burak Uyar, Sertan Şentürk, Barış Bozkurt, and Xavier Serra. Audio feature extraction for exploring Turkish makam music. In *3rd International Conference on Audio Technologies for Music and Media*, Ankara, Turkey, 2014. Bilkent University, Bilkent University.

[2] Georgi Dzhambazov, Sertan Sentürk, and Xavier Serra. Automatic lyrics-to-audio alignment in classical Turkish music. In *The 4th International Workshop on Folk Music Analysis*, pages 61–64, 2014.

[3] Eric Bernard Ederer. *The Theory and Praxis of Makam in Classical Turkish Music 1910–2010*. University of California, Santa Barbara, 2011.

[4] Hiromasa Fujihara and Masataka Goto. Lyrics-to-audio alignment and its application. *Dagstuhl Follow-Ups*, 3, 2012.

[5] Hiromasa Fujihara, Masataka Goto, Jun Ogata, and Hiroshi G Okuno. Lyricsynchronizer: Automatic synchronization system between musical audio signals and lyrics. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1252–1261, 2011.

[6] Emilia Gómez and Jordi Bonada. Towards computer-assisted flamenco transcription: An experimental comparison of automatic transcription algorithms as applied to a cappella singing. *Computer Music Journal*, 37(2):73–90, 2013.

[7] Rong Gong, Philippe Cuvillier, Nicolas Obin, and Arshia Cont. Real-time audio-to-score alignment of singing voice based on melody and lyric information. In *Interspeech 2015*, Dresden, Germany, 06/09/2015 2015.

[8] Jens Kofod Hansen. Recognition of phonemes in a-cappella recordings using temporal patterns and mel frequency cepstral coefficients. In *Proceedings of the 9th Sound and Music Computing Conference*, pages 494–499, Copenhagen, Denmark, 2012.

[9] Michael T Johnson. Capacity and complexity of HMM duration modeling techniques. *Signal Processing Letters, IEEE*, 12(5):407–410, 2005.

[10] M. Kemal Karaosmanoğlu, Barış Bozkurt, Andre Holzapfel, and Nilgün Doğrusöz Dişiaçık. A symbolic dataset of Turkish makam music phrases. In *Fourth International Workshop on Folk Music Analysis (FMA2014)*, 2014.

[11] Willie Krige, Theo Herbst, and Thomas Niesler. Explicit transition modelling for automatic singing transcription. *Journal of New Music Research*, 37(4):311–324, 2008.

[12] Nadine Kroher and Emilia Gómez. Automatic transcription of flamenco singing from polyphonic music recordings. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 24(5):901–913, 2016.

[13] Matthias Mauch, Hiromasa Fujihara, and Masataka Goto. Integrating additional chord information into hmm-based lyrics-to-audio alignment. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(1):200–210, 2012.

[14] Annamaria Mesaros and Tuomas Virtanen. Automatic alignment of music audio and lyrics. In *Proceedings of the 11th Int. Conference on Digital Audio Effects (DAFx-08)*, 2008.

[15] Emilio Molina, Lorenzo J Tardón, Ana M Barbancho, and Isabel Barbancho. Sipth: Singing transcription based on hysteresis defined on the pitch-time curve. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(2):252–263, 2015.

[16] Özgül Salor, Bryan L Pellom, Tolga Çiloğlu, and Mübeccel Demirekler. Turkish speech corpora and recognition tools developed by porting sonic: Towards multilingual speech recognition. *Computer Speech and Language*, 21(4):580 – 593, 2007.

[17] Lawrence Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[18] Justin Salamon, Emila Gómez, Dan Ellis, and Gaël Richard. Melody extraction from polyphonic music signals: Approaches, applications and challenges. *IEEE Signal Processing Magazine*, 31:118–134, 02/2014 2014.

[19] Justin Salamon and Emilia Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[20] Xavier Serra. A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition. Technical report, 1989.

[21] Johan Sundberg. The KTH synthesis of singing. *Advances in Cognitive Psychology*, 2(2-3):131–143, 2006.

[22] Johan Sundberg and Thomas D Rossing. The science of singing voice. *the Journal of the Acoustical Society of America*, 87(1):462–463, 1990.

[23] Burak Uyar, Hasan Sercan Atlı, Sertan Şentürk, Barış Bozkurt, and Xavier Serra. A corpus for computational research of Turkish makam music. In *1st International Digital Libraries for Musicology Workshop*, pages 57–63, London, United Kingdom, 2014.

[24] Steve J Young. *The HTK hidden Markov model toolkit: Design and philosophy*. Citeseer, 1993.

[25] Shun-Zheng Yu. Hidden semi-Markov models. *Artificial Intelligence*, 174(2):215–243, 2010.

QUERYING XML SCORE DATABASES: XQUERY IS NOT ENOUGH!

Raphaël Fournier-S'niehotta
CNAM
fournier@cnam.fr

Philippe Rigaux
CNAM
philippe.rigaux@cnam.fr

Nicolas Travers
CNAM
nicolas.travers@cnam.fr

ABSTRACT

The paper addresses issues related to the design of query languages for searching and restructuring collections of XML-encoded music scores. We advocate against a direct approach based on XQuery, and propose a more powerful strategy that first extracts a structured representation of music notation from score encodings, and then manipulates this representation in closed form with dedicated operators. The paper exposes the content model, the resulting language, and describes our implementation on top of a large Digital Score Library (DSL).

1. INTRODUCTION

It is now common to serialize scores as XML documents, using encodings such as MusicXML [11, 16] or MEI [15, 18]. Ongoing work held by the recently launched W3C Music Notation Community Group [20] confirms that we can expect in a near future the emergence of large collections of digital scores.

1.1 Issues with Querying XML score databases

A natural question in a collection management perspective is the definition of a query and manipulation language to access, search, and possibly analyze these collections. While XQuery appears as a language of choice, we consider that it does not constitute, as such, a suitable solution. There are several reasons that prevent XQuery from being able to address the complex requirements of music notation manipulation, at least beyond the simplest operations.

1. **Issue 1: Heterogeneity.** Score encodings closely mix information related to the *content* (e.g., the sequence of notes of a voice) and to a specific *rendering* of this content (e.g., voice allocation to a staff, positioning of notes/lines/pages, and other options pertaining to scores visualization). While it is not always obvious to clearly distinguish content from rendering instructions, mixing both concerns leads to an intricate encoding from which selecting the relevant information becomes extremely difficult. Extracting for instance melodic information from either MusicXML or MEI turns out to be difficult; and more sophisticated extractions (e.g., alignment of melodic information for voices) are almost impossible.

2. **Issue 2: Operations and closure.** One of the main requirements of a query language is a set of well-defined operations that operate in closed form: given as input objects that comply to some structural constraints (a data model), the language should guarantee that any output obtained by combining the operators is model-compliant as well. This requirement allows an arbitrary composition of operations, and ensures the safety of query results. In our context, it concretely means that we need operators that manipulate music notation, and that their output should consist of valid music notation as well. There is however no means to achieve that with XQuery, nor even to simply know whether a query supplies a compliant output or not.

3. **Issue 3: Formats.** Finally, a last, although less essential, obstacle is the number of possible encodings available nowadays, from legacy formats such as HumDrum to recent XML proposal mentioned above. Abstracting away from the specifics of these formats in favor of the core information set that they commonly aim at representing would allow to get rid of their idiosyncrasies.

To summarize, we consider that XML representation of scores are so far mostly intended as a serialization of documents that encapsulate all kinds of information, from metadata (creation date, encoding agent) to music information (symbolic representation of sounds and sounds synchronization) via rendering instructions. They are by no means designed to supply a structured representation of a core aspect (music "content"), subject to investigation and manipulation via a dedicated, model-aware query language.

1.2 Our approach

We make the case for an approach that leverages music content information as virtual XML objects. Coupled with a specialization of XQuery to this specific representation, we obtain a system apt at providing search, restructuration, extraction, analytic services on top of large collections of XML-encoded scores. The system architecture is summarized in Figure 1, and addresses the above issues.

© Raphaël Fournier-S'niehotta, Philippe Rigaux, Nicolas Travers. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Raphaël Fournier-S'niehotta, Philippe Rigaux, Nicolas Travers. "QUERYING XML SCORE DATABASES: XQUERY IS NOT ENOUGH!", 17th International Society for Music Information Retrieval Conference, 2016.

Issue 1: Bringing homogeneity. The bottom layer is a Digital Score Library managing collections of scores serialized in MusicXML, MEI, or any other legacy format (e.g., Humdrum). This encoding is *mapped* toward a model layer where the content structured in XML corresponds to the model structures. This defines, in intention, collections of music notation objects that we will call *vScore* in the following. A vScore abstracts the part of the encoding (here the content) we wish to focus on, and gets rid of information considered as useless, at least in this context.

Issue 2: Defining a domain-specific language. In order to manipulate these vScores, we equip XQuery with two classes of operations dedicated to music content: *structural operators* and *functional operators*. The former implement the idea that structured scores management corresponds, at the core level, to a limited set of fundamental operations, grouped in a score algebra, that can be defined and implemented only once. The latter acknowledges that the richness of music notation manipulations calls for a combination of these operations with user-defined functions at early steps of the query evaluation process. Modeling the invariant operators and combining them with user-defined operations constitutes the operational part of the model. This yields a query language whose expressions unambiguously define the set of transformations that produce new *vScores* from the base collections.

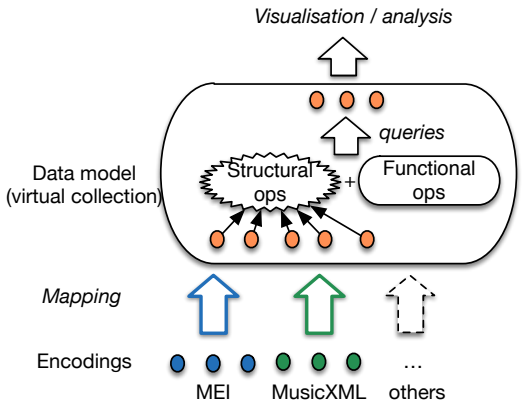


Figure 1. Envisioned system

Issue 3: Serialization independence. One mapper has to be defined for each possible encoding, as shown by the figure which assumes that MusicXML and MEI documents cohabit in a single DSL. Adding a new source represented with a new encoding is just a matter of adding a new mapper. Each document in the DSL is then mapped to a (virtual) XML document, instance of the model.

1.3 Contributions

In the present paper, we describe the implementation of the above ideas in NEUMA [17]. The focus is on the model layer, defined as a virtual XML schema, on the mapping from raw documents to vScores, and on the integration of XQuery with structural operators and external functions. The score algebra, not presented here, is implemented in our system as XQuery functions, whose meaning should be clear from the context.

Section 2 gives the virtual XML schema for music content notation, and Section 3 shows how to create an XML database referring to vScores. Section 4 presents the query language and Section 5 discusses salient implementation choices. Section 6 covers related work and Section 7 concludes the paper.

2. MUSIC NOTATION: THE SCHEMA

We now describe the virtual data model with XML Schema [22]. The model aims at representing polyphonic scores in Common Music Notation (CMN). A score is composed of voices, and a voice is a sequence of events.

2.1 Event type

An event is a value (complex or simple) observed during a time interval. Events are polymorphic: the value may be a note representation, a chord representation, a syllable or any other value (e.g., an integer representing an interval).

The abstract definition of an event is a complex type with a duration attribute.

```
<xs:complexType abstract="true" name="eventType">
  <xs:attribute type="xs:integer" name="duration"
       use="required"/>
</xs:complexType>
```

From this abstract type, we can derive concrete event types with specific element names. The most important are events denoting sounds, which covers simple $n \geq 0$ simultaneous sounds, either rests ($n = 0$), notes ($n = 1$) or chords ($n > 1$). The `soundType` is derived from the `eventType` as follows:

```
<xs:complexType name="soundType">
  <xs:complexContent>
    <xs:extension base="eventType">
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="note" type="noteType"/>
        <xs:element name="rest" type="restType"/>
        <xs:element name="chord" type="chordType"/>
      </xs:choice>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Due to space restrictions, we do not detail `noteType` (containing 'pitch' and 'octave' attributes), `restType` (empty element) and `chordType` (list of `noteType`). As another example of a concrete event type, lyrics can be represented with syllabic events (and rests), with type:

```
<xs:complexType name="syllableType">
  <xs:complexContent>
    <xs:extension base="eventType">
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="syll" type="xs:string"/>
        <xs:element name="rest" type="restType"/>
      </xs:choice>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Events are not restricted to musical domains. Events in the `xs:integer` domain, for instance, can be used to represent intervals, obtained by a 2-voices scores analysis.

```
<monody>
  <rest duration="24"/>
  <note duration="16" p="D" o="5"/>
  <rest duration="4"/>
  <note duration="2" p="E" o="5"/>
  <note duration="2" p="F" o="5"/>...
</monody>
```

```
<lyrics>
  <rest duration="24"/>
  <syll duration="16"/>Ah,</syll>
  <rest duration="4"/>
  <syll duration="2"/>que</syll>
  <syll duration="2"/>je</syll>...
</lyrics>
```

```
<bass>
  <note duration="8" p="D" o="4"/>
  <note duration="4" p="C" o="4"/>
  <chord duration="4">
    <note p="D" o="4" a="-1"/>
    <note p="B" o="3" a="-1"/></chord>
  <note duration="4" p="A" o="3"/>
  <note duration="4" p="G" o="3"/>...
</bass>
```

Figure 2. Voices representation

2.2 Voice type

A voice is a sequence of events. Its abstract definition is given by the following schema:

```
<xs:complexType name="voiceType" abstract="true">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element type="eventType"/>
  </xs:sequence>
</xs:complexType>
```

This type actually represents a function from the time domain to the domain of events. There is an implicit non-overlapping constraint: an event begins when its predecessor ends. We can instantiate concrete voice types by simply replacing the abstract `eventType` by one of its derived types (e.g., `soundType`, `syllableType`, `intType`), like in the following example:

```
<xs:complexType name="lyricsType">
 <xs:extension base="voiceType">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
   <xs:element type="syllableType"/>
  </xs:sequence>
 </xs:extension>
</xs:complexType>
```

2.3 Score type

Finally, our virtual notation schema contains a score type which describes a recursive structure defined as follows:

- if v a voice, then v is a score.
- if $s_1, \cdots, s_n$ are scores, the sequence $< s_1, \cdots, s_n >$ is a score.

The generic definition of the XML schema for this structure is given below. Note that element names for scores and voices will be specified for each specific corpus.

```
<xs:complexType name="scoreType" abstract="true">
  <xs:sequence minOccurs="1" maxOccurs="unbounded">
    <xs:choice>
      <xs:element type="scoreType"/>
      <xs:element type="voiceType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

2.4 Example

Let's illustrate by an example our types and structure. Fig. 3 is partially represented by the vScore in Fig. 2.

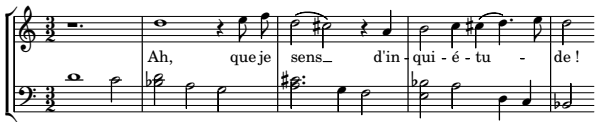


Figure 3. A score example

Our model decomposes the score in three voices. The first one represents the monody of the vocal part. It consists of a sequence of `soundType` events. The second voice represents lyrics with `syllableType` events. And, finally, the last voice is the bass. Its representation is a sequence of `soundType` events (here, notes and chords).

The vScore itself illustrates the recursive structure that encapsulates the voices in a tree of `score` elements. The upper level combines the `bass` voice and an embedded `score` which combines the `monody` and `lyrics` voices. The structure is as follows.

```
<air>
  <vocal>
    <monody>(...)</monody>
    <lyrics>(...)</lyrics>
  </vocal>
  <bass>(...)</bass>
</air>
```

It should be clear that this representation abstracts a part of the content that can be found in all the encodings we are aware of. The choice of the information subset which is selected is here minimal, for the sake of conciseness. We can obviously extend the representation with additional details as long as it does not affect the structure. A voice can be "decorated" by an instrument name, an event by the current metric or the measure number, a score by its composer, all represented as additional elements. In general, the issue relates to what is considered as "content" subject to search and analysis operations, and what is the suitable representation for this content. We will stick in the following to the simple model given above which is sufficient to our needs.

Recall that the schema intends to define a *virtual* score representation which is derived at search time (according to rules explained in the next sections) from the actual serialization. We briefly explain the mapping from MusicXML or MEI documents to vScores.

2.5 Mapping from MusicXML or MEI

In MusicXML, scores are organized as a tree of `score`, `part-group`, and `part` elements. Voices are numbered with respect to the part they belong to, and represented as nested elements of notes, rest and chords. Our mapping unifies the score, groups and parts in a recursive nesting of `score` elements and (virtually) splits the music and lyrics as two associated voices.

In MEI, the score structure is based on score, staves and groups of staves. Voices are represented as `layer` objects deeply nested in a hierarchy of `measure` and `staff` containers. Our mapping extracts the voice events from the complex imbrication of MEI elements or organizes them according to our recursive `score` structure.

3. MUSIC NOTATION: THE DATABASE

XML databases are collections of documents. Although the definition of a schema for a collection is not mandatory, it is a safe practice to ensure that all the documents it contains share a similar structure. In our context, a collection of digital scores is a regular XML collection where one or several elements are of `scoreType` type. Here is a possible example:

```
<xs:complexType name="opusType">
  <xs:sequence>
    <xs:element name="title" type="xs:string"/>
    <xs:element name="composer" type="xs:string"/>
    <xs:element name="published" type="xs:string"/>
    <xs:element type="scoreType"/>
  </xs:sequence>
  <xs:attribute type="xs:ID" name="id"/>
</xs:complexType>
```

Note that this schema is strict regarding meta-data (title, composer) but very flexible for the music notation because we are using here the generic `scoreType`. It follows that, from one document to the other in standard notations (musicXML, MEI, HumDrum), the score structure, the number of voices and their identifiers may vary. Such a flexibility is definitely not convenient when it comes to querying a collection, since we cannot safely refer to the components of a score. It seems more appropriate to base the organization of score collections on an homogeneous score structure. A *Quartet* collection for instance would only accept scores complying to the following structure:

```
Quartet(id: int, title: string,
        composer: string, published: date,
        music: Score [v1, v2, alto, cello])
```

The XML Schema formalism accepts the definition of *restriction* of a base type. In our case, the restriction consists in specializing the `scoreType` definition to list the number and names of voices, like in `quartetType`:

```
<xs:complexType name="quartetType">
 <xs:complexContent>
  <xs:restriction base="scoreType">
   <xs:sequence>
    <xs:element name="v1" type="soundVoiceType"/>
    <xs:element name="v2" type="soundVoiceType"/>
    <xs:element name="alto" type="soundVoiceType"/>
    <xs:element name="cello" type="soundVoiceType"/>
   </xs:sequence>
  </xs:restriction>
  <xs:attribute type="xs:ID" name="id"/>
 </xs:complexContent>
 </xs:complexType>
```

Using `QuartetType` in place of `ScoreType` in the collection schema ensures that all the vScores in the collection match the definition. Voices' names are specified and can then be used to access to the (virtual) music notation to start applying operations and transformations. This can be done with XQuery, as explained in the next section.

4. XQUERY + SCORE ALGEBRA = QUERIES

With a well-defined collection and a clear XML model to represent the notation of music content, we can now express queries over this collection with XQuery. However, executing such queries gives rise to the following issues:

Issue 1: We *cannot* directly evaluate an XQuery expression, since they are interpreted over instances which are partly virtual (scores, voices and events) and partly materialized (all the rest: title, composer, etc.).
Issue 2: Pure XQuery expression would remain limited to exploit the richness of music notation;

The first issue is solved by executing, at run-time, the mapping that transforms the serialized score (say, in MusicXML) to a vScore, instance of our model. This is further explained in the next section, devoted to implementation choices. To solve the second issue, we implemented a set of XQuery functions forming a score algebra. We introduce it and illustrate the resulting querying mechanism with examples.

4.1 Designing a score algebra

We designed a score algebra in a database perspective, as a set of operators that operate in closed form: each takes one or two vScores (instances of `scoreType`) as input and produces a vScore as output. This brings composition, expressiveness, and safety of query results, since they are guaranteed to consist of vScore instances that can, if needed, be serialized back in some standard encoding (see the discussion in Section 1 and the system architecture, Fig. 1). The algebra is formalized in [8], and implemented as a set of query functions whose meaning should be clear from the examples given next.

4.2 Queries

The examples rely on the *Quartet* corpus (refer to the Section 3 for its schema). Our first example creates a list of *Haydn*'s quartets, reduced to the titles and violin's parts.

```
for $s in collection("Quartet")
where $s/composer="Haydn"
return $s/title, Score($s/music/v1, $s/music/v2)
```

Recall that `music` is a `QuartetType` element in the *Quartet* schema. This first query shows two basic operators to manipulate scores: projection on voices (obtained with XPath), and creation of new scores from components (voices or scores) with the `Score()` operator.

A third operator illustrated next is MAP. It represents a higher-order function that applies a given function f to each event in a vScore, and returns the score built from f's results. Here is an example: we want the quartets where the v1 part is played by a B-flat clarinet. We need to transpose the `v1` part 2 semi-tones up.

```
for $s in collection("Quartet")
where $s/composer="Haydn"
let $clarinet := Map ($s/music/v1, transpose (2))
let $clrange := ambitus ($clarinet)
return $s/title, $clrange,
         Score($clarinet, $s/music/v2,
                 $s/music/alto, $s/music/cello)
```

This second query shows how to define variables that hold new content derived from the vScore via *user defined functions* (UDFs). For the sake of illustration we create two variables, `$clarinet` and `$clrange`, calling respectively `transpose()` and `ambitus()`.

In the first case, the function has to be applied to each event of the violin voice. This is expressed with MAP which yields a new voice with the transposed events. By contrast, `ambitus()` is directly applied to the voice as a whole. It produces a scalar value.

MAP is the primary means by which new vScores can be created by applying all kinds of transformations. MAP is also the operator that opens the query language to the integration of *external* functions: any library can be integrated as a first-class component of the querying system, providing some technical work to "wrap" it conveniently.

The selection operator takes as input a vScore, a Boolean expression e, and filters out the events that do not satisfy e, replacing them by a `null` event. Note that this is different from *selecting* a score based on some property of its voice(s). The next query illustrates both functionalities: a user-defined function "lyricsContains" selects all the psalms (in a *Psalters* collection) such that the vocal part contains a given word ("*Heureux*"), "nullify" the events that do not belong to the measures five to ten, and trim the voice to keep only non-null events.

```
for $s in collection("Psalters")
let $sliced := trim(select ($s/air/vocal/monody,
                            measure(5, 10)))
where lyricsContains ($s/air/vocal/lyrics, "Heureux")
return $s/title, Score($sliced)
```

We can take several vScores as input and produce a document with several vScores as output. The following example takes three chorals, and produces a document with two vScores associating respectively the alto and tenor voices.

```
for $c1 in collection("Chorals")[@id="BWV49"]/music,
    $c2 in collection("Chorals")[@id="BWV56"]/music,
    $c3 in collection("Chorals")[@id="BWV12"]/music
return <title>Excerpts of chorals 49, 56, 12</title>,
          Score($c1/alto, $c2/alto, $c3/alto),
          Score($c1/tenor, $c2/tenor, $c3/tenor)
```

Finally, our last example shows the extended concept of score as a voice synchronization which are not necessarily "music" voices. The following query produces, for each quartet, a vScore containing the violin 1 and cello voices, and a third one measuring the interval between the two.

```
for $s in collection("Quartet")/music
let $intervals := Map(Score($s/v1,$s/cello),interval())
return Score ($s/v1, $s/cello, $intervals)
```

Such a "score" cannot be represented with a traditional rendering. Additional work on visualization tools that would closely put in perspective music fragments along with some computed analytic feature is required.

5. IMPLEMENTATION

Our system integrates an implementation of our score algebra, a mapping that transforms serialized scores to vScores, and off-the-shelf tools (a native XML database, BASEX [1], a music notation library for UDFs, MUSIC21 [2] [4]). This simple implementation yields a query system which is both powerful and extensible (only add new functions wrapped in XQuery/BASEX). We present its salient aspects.

[1] http://basex.org
[2] http://web.mit.edu/music21

5.1 Query processing

The architecture presented in Figure 4 summarizes the components involved in query processing. Data is stored in BASEX in two collections: the semi-virtual collection (e.g., Quartet) of music documents (called *opus*), and the collection of serialized scores, in MusicXML or MEI. Each virtual element `scoreType` in the former is linked to an actual document in the latter.

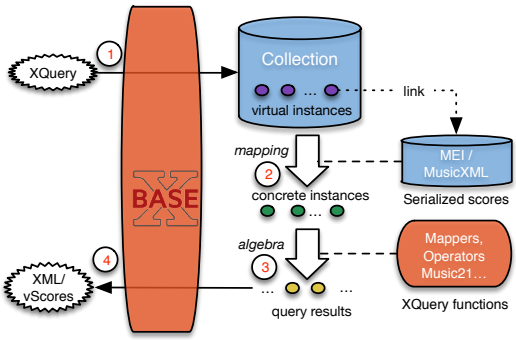


Figure 4. Architecture

The evaluation of a query proceeds as follows. First (step 1), BASEX scans the virtual collection and retrieves the opus matching the `where` clause (at least for fields that do not belong to the virtual part, see the discussion at the end of the section). Then (step 2), for each opus, the embedded virtual element `scoreType` has to be materialized. This is done by applying the mapping that extracts a vScore instance from the serialized score, thanks to the link in each opus.

Once a vScore is instantiated, algebraic expressions, represented as composition of functions in the XQuery syntax, can be evaluated (step 3). We wrapped several Python and Java libraries as XQuery functions, as permitted by the BASEX extensible architecture. In particular, algebraic operators and mappers are implemented in Java, whereas additional, music-content manipulations are mostly wrapped from the Python Music21 toolbox.

The XQuery processor takes in charge the application of functions, and builds a collection of results (that includes instances of `scoreType`), finally sent to the client application (step 4). It is worth noting that the whole mechanism behaves like an ActiveXML [1] document which *activates* the XML content on demand by calling an external service (here, a function).

5.2 Mappers

In order to map scores from the physical representation to the virtual one, references to physical musical parts are matched, according to the collection schema. To achieve this, an ID is associated to each voice element of the materialized score. This ID directly identifies the underlying part of the physical document.

The main mapping challenge is to identify virtual and serialized voices, in particular when they are not standardized according to the collection schema. We need to gen-

erate IDs for each parts on the underlying encoding (MusicXML, MEI, etc.), even for monody and lyrics parts which can be merged on the physical document. Most of them can be done automatically with metadata information given by the collection schema.

5.3 Manipulating vScores with SCORELIB

The SCORELIB Java library is embedded in every BASEX query in order to process our algebraic operations on vScores. The library is responsible for managing the link between the virtual and the physical score, whatever the encoding format. Whenever a function *activates* a vScore by calling a function (whether a structural function of the algebra, or a user-defined function), the link is used to get and materialize the corresponding score.

Once vScore has been materialized, it is kept in a cache in order to avoid repeated and useless applications of the mapping. Temporary vScores produced by applying algebraic operators are kept in the cache as well.

The `Score()` operator creates the final XML document featuring one or several instances of `scoreType`. It combines scores produced by operators and referred to by XQuery variables.

5.4 Indexing music features

User Defined Functions (*UDFs*) are necessary to produce derived information from music notation, and have to be integrated as external components. Getting the highest note of a voice for instance is difficult to express in XQuery, even on the regular structures of our model. In general, getting sophisticated features would require awfully complex expressions. In our current implementation, UDFs are taken from MUSIC21 and wrapped as XQuery functions. This works with quite limited implementation efforts, but can be very inefficient since every score must be materialized before evaluating the UDF. Consider the following example which retrieves all the quartets such that the first violin part gets higher than e6:

```
for $s in collection("Quartet")
where highest($s/music/v1)  > 'e6' return $s
```

A naive, direct evaluation maps the MusicXML (or MEI) document as a vScore, passes it to the XQuery function that delegates the computation to the user library (e.g., MUSIC21 or any other) and gets the result. This has to be done for each score in the collection, even though they do not all match the selection criteria.

A solution is to materialize the results of User Defined Functions as metadata in the virtual document and to index this new information in BASEX. This can directly serve as a search criteria without having to materialize the vScore. The result of the *highest()* function is such a feature. Index creation simply scans the whole physical collections, runs the functions and records it result in a dedicated `index` sub-element of each opus, automatically indexed in BASEX. To evaluate the query above, it uses the access path to directly get the relevant opus.

```
for $s in collection("Quartet")[index/v1/highest  > 'e6']
return $s
```

6. RELATED WORK

Accessing to structured music notation for search, analysis and extraction is a long-term endeavor. Humdrum [13] works on plain text (ASCII) file format, whereas MUSIC21 [4] deals with MIDI channels modeled as musical layers. Both can import widely used formats like MusicXML or MEI. Both are powerful toolkits, but their main focus is on the development of scripts and not database-like access to structured content. As a result, using, say MUSIC21 to express the equivalent of our queries would require to develop ad-hoc scripts possibly rather complex. It becomes all the more complicated when dealing with huge collections of scores. On the other hand, there are many computations that a database language cannot express, which motivated our introduction of UDFs in the language.

Other musical score formalisms rather target generative process and computer-aided composition. This is the case of Euterpea [12] (in `Haskell`), musical programming approaches [3, 6, 7, 14] and operations on *tiled streams* in T-Calculus [14]. They follow the paradigm of abstract data types for music representation, bringing a simplification to the music programming task, but they are not adapted to the conciseness of a declarative query language.

Since modern score formats adopt an XML-based serialization, XQuery [23] has been considered as the language of choice for score manipulation [9]. THoTH [21] also proposes to query MusicXML with patterns analysis. For reasons developed in the introduction, we believe that a pure XQuery approach is too generic to handle the specifics of music representation.

Our work is inspired by XQuery mediation [10, 5, 2, 19], and can be seen as an application of method that combines queries on physical and virtual instances. It borrows ideas from ActiveXML [1], and in particular the definition of some elements as "triggers" that activate external calls.

7. CONCLUSION

We propose in the present paper a complete methodology to view a repository of XML-structured music scores as a structured database, equipped with a domain-specialized query language. Our approach aims at limiting the amount of work needed to implement a working system. We model music notation as structured scores that can easily be extracted from existing standards at run-time; we associate to the model an algebra to access to the internal components of the scores; we allow the application of external functions; and finally we integrate the whole design in XQuery, with limited implementation requirements.

We believe that this work brings a simple and promising framework to define a query interface on top of Digital Libraries, with all the advantages of a concise and declarative approach for data management. It also offers several interesting perspectives: automatic content management (split a score in parts, distribute them to digital music stands), advanced content-based search, and finally advanced mining tasks (derivation of features, annotation of scores with these features).

8. REFERENCES

[1] Serge Abiteboul, Omar Benjelloun, and Tova Milo. The active XML project: an overview. *VLDB J.*, 17(5):1019–1040, 2008.

[2] Serge Abiteboul and et al. WebContent: Efficient P2P Warehousing of Web Data. In *VLDB'08 Very Large Data Base*, pages 1428–1431, August 2008.

[3] Mira Balaban. The music structures approach to knowledge representation for music processing. *Computer Music Journal*, 20(2):96–111, 1996.

[4] Michael Scott Cuthbert and Christopher Ariza. Music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 637–642, 2010.

[5] AnHai Doan, Alon Halevy, and Zachary Ives. *Principles of Data Integration*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2012.

[6] Dominique Fober, Stéphane Letz, Yann Orlarey, and Frédéric Bevilacqua. Programming Interactive Music Scores with INScore. In *Sound and Music Computing*, pages 185–190, Stockholm, Sweden, July 2013.

[7] Dominique Fober, Yann Orlarey, and Stéphane Letz. Scores level composition based on the guido music notation. In ICMA, editor, *Proceedings of the International Computer Music Conference*, pages 383–386, 2012.

[8] R. Fournier-S'niehotta, P. Rigaux, and N. Travers. An Algebra for Score Content Manipulation. Technical Report CEDRIC-16-3616, CEDRIC laboratory, CNAM-Paris, France, 2016.

[9] Joachim Ganseman, Paul Scheunders, and Wim D'haes. Using XQuery on MusicXML Databases for Musicological Analysis. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2008.

[10] H. Garcia-Molina, J.D. Ullman, and J. Widom. *Database System Implementation*. Prentice Hall, 2000.

[11] Michael Good. *MusicXML for Notation and Analysis*, pages 113–124. W. B. Hewlett and E. Selfridge-Field, MIT Press, 2001.

[12] Paul Hudak. *The Haskell School of Music – From Signals to Symphonies*. (Version 2.6), January 2015.

[13] David Huron. Music information processing using the humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):11–26, July 2002.

[14] David Janin, Florent Berthaut, Myriam Desainte-Catherine, Yann Orlarey, and Sylvain Salvati. The T-Calculus : towards a structured programing of (musical) time and space. In *Proceedings of the first ACM SIGPLAN workshop on Functional art, music, modeling and design (FARM'13)*, pages 23–34, 2013.

[15] Music Encoding Initiative. http://music-encoding.org, 2015. Accessed Oct. 2015.

[16] MusicXML. http://www.musicxml.org, 2015. Accessed Oct. 2015.

[17] Philippe Rigaux, Lylia Abrouk, H. Audéon, Nadine Cullot, C. Davy-Rigaux, Zoé Faget, E. Gavignet, David Gross-Amblard, A. Tacaille, and Virginie Thion-Goasdoué. The design and implementation of neuma, a collaborative digital scores library - requirements, architecture, and models. *Int. J. on Digital Libraries*, 12(2-3):73–88, 2012.

[18] Perry Rolland. The Music Encoding Initiative (MEI). In *Proc. Intl. Conf. on Musical Applications Using XML*, pages 55–59, 2002.

[19] Nicolas Travers, Tuyêt Trâm Dang Ngoc, and Tianxiao Liu. Tgv: A tree graph view for modeling untyped xquery. In *12th International Conference on Database Systems for Advanced Applications (DASFAA)*, pages 1001–1006. Springer, 2007.

[20] W3C Music Notation Community Group. https://www.w3.org/community/music-notation/, 2015. Last accessed Jan. 2016.

[21] Philip Wheatland. Thoth music learning software, v2.5, Feb 27, 2015. http://www.melodicmatch.com/.

[22] XML Schema. World Wide Web Consortium, 2001. http://www.w3.org/XML/Schema.

[23] XQuery 3.0: An XML Query Language. World Wide Web Consortium, 2007. https://www.w3.org/TR/xquery-30/.

RECURRENT NEURAL NETWORKS FOR DRUM TRANSCRIPTION

Richard Vogl, Matthias Dorfer, Peter Knees
Department of Computational Perception
Johannes Kepler University Linz, Austria
richard.vogl@jku.at

ABSTRACT

Music transcription is a core task in the field of music information retrieval. Transcribing the drum tracks of music pieces is a well-defined sub-task. The symbolic representation of a drum track contains much useful information about the piece, like meter, tempo, as well as various style and genre cues. This work introduces a novel approach for drum transcription using recurrent neural networks. We claim that recurrent neural networks can be trained to identify the onsets of percussive instruments based on general properties of their sound. Different architectures of recurrent neural networks are compared and evaluated using a well-known dataset. The outcomes are compared to results of a state-of-the-art approach on the same dataset. Furthermore, the ability of the networks to generalize is demonstrated using a second, independent dataset. The experiments yield promising results: while F-measures higher than state-of-the-art results are achieved, the networks are capable of generalizing reasonably well.

1. INTRODUCTION AND RELATED WORK

Automatic music transcription (AMT) methods aim at extracting a symbolic, note-like representation from the audio signal of music tracks. It comprises important tasks in the field of music information retrieval (MIR), as — with the knowledge of a symbolic representation — many MIR tasks can be address more efficiently. Additionally, a variety for direct applications of AMT systems exists, for example: sheet music extraction for music students, MIDI generation/re-synthesis, score following for performances, as well as visualizations of different forms.

Drum transcription is a sub-task of AMT which addresses creating a symbolic representation of all notes played by percussive instruments (drums, cymbals, bells, etc.). The source material is usually, as in AMT, a monaural audio source—either from polyphonic audio containing multiple instruments, or a solo drum track. The symbolic representation of notes played by the percussive instruments can be used to derive rhythmical meta-information like tempo, meter, and downbeat. The repetitive rhythmical structure of the drum track, as well as changes therein, can be used as features for high-level MIR tasks. They provide information about the overall structure of the song which can be utilized for song segmentation [18]. The drum rhythm patterns can also be utilized for genre classification [7]. Other applications for rhythmic patterns include query-by-tapping and query-by-beat-boxing [11,19].

A common approach to the task of drum transcription is to apply methods used for source separation like non-negative matrix factorization (NMF), independent component analysis (ICA), or sparse coding. In recent work, Dittmar and Gärtner [5] use an NMF approach to transcribe solo drum tracks into three drum sound classes representing bass drum, snare drum, and hi-hat. They achieve F-measure values of up to 95%. Their approach focuses on real-time transcription of solo drum tracks for which training instances of each individual instrument are present. This is a very specific use case and in many cases separate training instances for each instrument are not available. A more general and robust approach which is able to transcribe different sounding instruments is desirable. Smaragdis [28] introduces a convolutional NMF method. It uses two-dimensional matrices (instead of one-dimensional vectors used in NMF) as *temporal-spectral bases* which allow to consider temporal structures of the components. Smaragdis shows that this method can be applied to transcribe solo drum tracks. Lindsay-Smith and McDonald [21] extend this method and use convolutive NMF to build a system for solo drum track transcription. They report good results on a non-public, synthetic dataset.

Fitzgerald et al. [9] introduce prior subspace analysis, an ICA method using knowledge of the signals to be separated, and demonstrate the application for drum transcription. Spich et al. [29] extend this approach by incorporating a statistical music language model. These works focus on transcription of three and two instruments, respectively.

Scholler and Purwins [26] use a sparse coding approach to calculate a similarity measure for drum sound classification. They use eight basis vectors to represent the sounds for bass drum, snare drum, and hi-hat in the time domain. Yoshii et al. [33] present an automatic drum transcription system based on template matching and adaptation, similar to sparse coding approaches. They focus on transcription of snare and bass drum only, from polyphonic audio signals.

Algorithms based on source separation usually use the input signal to produce prototypes (or components) rep-

© Richard Vogl, Matthias Dorfer, Peter Knees. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Richard Vogl, Matthias Dorfer, Peter Knees. "Recurrent Neural Networks for Drum Transcription", 17th International Society for Music Information Retrieval Conference, 2016.

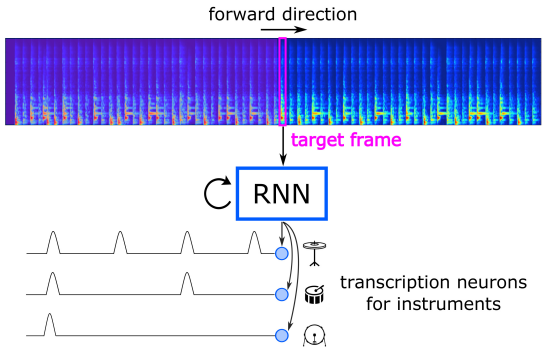


Figure 1. Overview of the proposed method. The extracted spectrogram is fed into the trained RNN which outputs activation functions for each instrument. A peak picking algorithms selects appropriate peaks as instrument onset candidates.

resenting individual instruments and so called activation curves which indicate the activity of them. A peak picking algorithm is needed to identify the instrument onsets in the activation curves. Additionally, the identified component prototypes have to be assigned to instruments. This is usually done using machine learning algorithms in combination with standard audio features [6, 16].

Another approach found in the literature is to first segment the audio stream using onset detection and classify the resulting fragments. Gillet and Richard [13] use a combination of a source separation technique and a support vector machine (SVM) classifier to transcribe drum sounds from polyphonic music. Miron et al. use a combination of frequency filters, onset detection and feature extraction in combination with a k-nearest-neighbor [23] and a k-means [22] classifier to detect drum sounds in a solo drum audio signal in real-time. Hidden Markov Models (HMMs) can be used to perform segmentation and classification in one step. Paulus and Klapuri [24] use HMMs to model the development of MFCCs over time. Decoding the most likely sequence yields activation curves for bass drum, snare drum, and hi-hat and can be applied for both solo drum tracks as well as polyphonic music.

Artificial neural networks consist of nodes (neurons) forming a directed graph, in which every connection has a certain weight. Since the discovery of gradient descent training methods which make training of complex architectures computationally feasible [17], artificial neural networks regained popularity in the machine learning community. They are being successfully applied in many different fields. Recurrent neural networks (RNNs) feature additional connections (recurrent connections) in each layer, providing the outputs of the same layer from the last time step as additional inputs. These connections can serve as memory for neural networks which is beneficial for tasks with sequential input data. RNNs have been shown to perform well, e.g., for speech recognition [25] and handwriting recognition [15]. Böck and Schedl use RNNs to improve beat tracking results [3] as well as for polyphonic

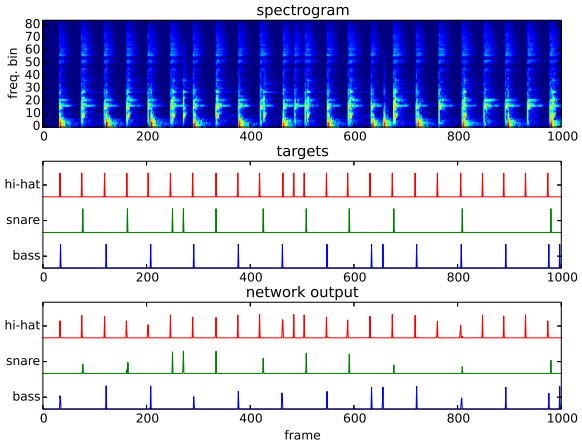


Figure 2. Spectrogram of a drum track and target functions for bass drum, snare drum, and hi-hat. The target function has a value of 1.0 at the frames at which annotations for the instruments exist and 0.0 otherwise. The frame rate of the target function is 100Hz, the same as for the spectrogram. The third graph shows the output of the trained RNN for the spectrogram in the first image.

piano transcription [4]. Sigtia et al. [27] use RNNs in the context of automatic music transcription as music language models to improve the results of a frame-level acoustic classifier. Although RNNs have been used in the past for transcription systems [4], we are not aware of any work using RNNs for transcription of drum tracks.

2. TASK AND MOTIVATION

In this work, we introduce a new method for automatic transcription of solo drum tracks using RNNs. While it is a first step towards drum transcription from polyphonic music, there also exist multiple applications for the transcription of solo drum tracks. In electronic music production, it can be used to transcribe drum loops if a re-synthesis using different sounds is desired. The transcription of recorded solo drum tracks can be used in the context of recording and production of rock songs. Nowadays it is not unusual to use sampled drums, e.g., in low-budget productions or in modern heavy metal genres. One the one hand, this is due to the complexity and costs of recording drums. On the other hand, with sampled drums it is easier to achieve the high precision and even robotic sounding style desired in some genres. Instead of manually programming the drum track, automatic transcription of a simple low-quality drum recording can be used as basis for the production of a song. As in other works, we focus on the transcription of bass drum, snare drum, and hi-hat. These instruments usually define the main rhythmic patterns [24], depending on genre and play style. They also cover most (>80% in the case of the ENST-Drums dataset, see Section 4.1) of the played notes in full drum kit recordings. Since simple RNN architectures already provide good transcription results (cf. Section 5), it is worthwhile exploring their application in this task further.

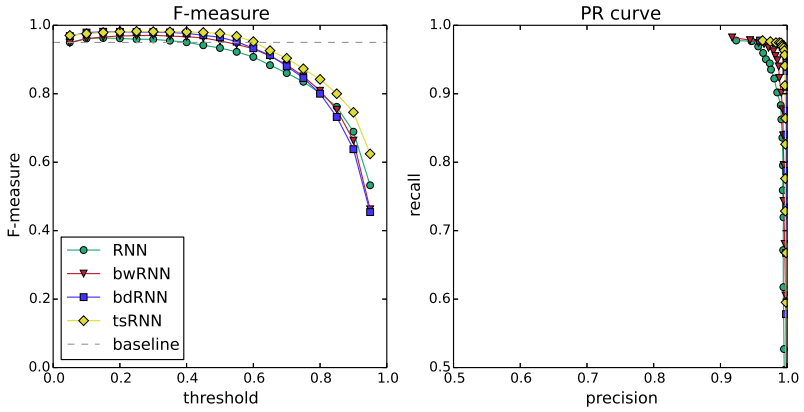


Figure 3. Result visualization for the evaluation on the IDMT-SMT-Drums dataset. The left plot shows the F-measure curve, the right plot the precision-recall curve for different threshold levels for peak picking.

3. METHOD

To extract the played notes from the audio signal, first a spectrogram of the audio signal is calculated. This is frame-wise fed into an RNN with three output neurons. The outputs of the RNN provide activation signals for the three drum instruments. A peak picking algorithm then identifies the onsets for each instrument's activation function, which yields the finished transcript (cf. Figure 1).

In this work, we compare four RNN architectures designed for transcribing solo drum tracks. These are: *i.* a simple RNN, *ii.* a backward RNN (bwRNN), *iii.* a bidirectional RNN (bdRNN), and *iv.* an RNN with time shift (tsRNN). The next section will cover the preprocessing of the audio signal, which is used for all four RNNs. After that, the individual architectures are presented in detail.

3.1 Signal Preprocessing

All four RNN architectures use the same features extracted from the audio signal. As input, mono audio files with 16 bit resolution at 44.1 kHz sampling rate are used. The audio is normalized and padded with 0.25 seconds of silence, to avoid onsets occurring immediately at the beginning of the audio file. First a logarithmic power spectrogram is calculated using a 2048 samples window size and a resulting frame rate of 100Hz. The frequency axis is then transformed to a logarithmic scale using twelve triangular filters per octave for a frequency range from 20 to 20,000 Hz. This results in a total number of 84 frequency bins.

3.2 Network Architectures

In this work, four different architectures of RNNs are compared. The four architectures comprise a plain RNN and three variations which are described in detail in the following.

3.2.1 Recurrent Neural Network

The plain RNN features a 84-node input layer which is needed to handle the input data vectors of the same size. The recurrent layer consists of 200 recurrently connected rectified linear units (ReLUs [14]). Although RNNs with ReLU activations can be difficult to train [20], good results without special initialization or treatment were achieved in this work. The connections between the input and the recurrent layer, the recurrent connections, and the connections between the recurrent layer and the output layer are all realized densely (every node is connected to all other nodes). The output layer consists of three nodes with sigmoid transfer functions, which provide the activation functions for the three instrument classes defined earlier. The sigmoid transfer function was chosen because binary cross-entropy was used as loss function for training, which turned out to be easier to train in the experiments.

3.2.2 Backward RNN

This RNN is very similar to the basic RNN with the only difference being that the recurrent connections are backward instead of forward in time. This was done in order to evaluate if the short sustain phase of percussive instruments provides additional information for the classification. The plain RNN has to identify the instruments at exactly the time frame of the onset annotation, thus the sustain phase of the notes can not be considered by it. This architecture is not real-time-capable since the audio to be transcribed is analyzed in reverse. Moreover, it might be more hard for this architecture to find the exact position of the onsets since the steep slope of the onset is only seen in forward direction.

3.2.3 Bidirectional RNN

The architecture of the bidirectional RNN used in this work consists of 100 nodes in a forward layer and 100 nodes in a backward layer. Both the forward and backward layers are directly connected to the input layer. Bidirectional RNNs often produce better results than unidirectional RNNs because they can also use the context of future frames for classification. In this work, they are meant to combine both the strengths of the forward and backward RNN. Unfortunately, this system has the same limitations as the backward RNN, making it not usable for real-time applications.

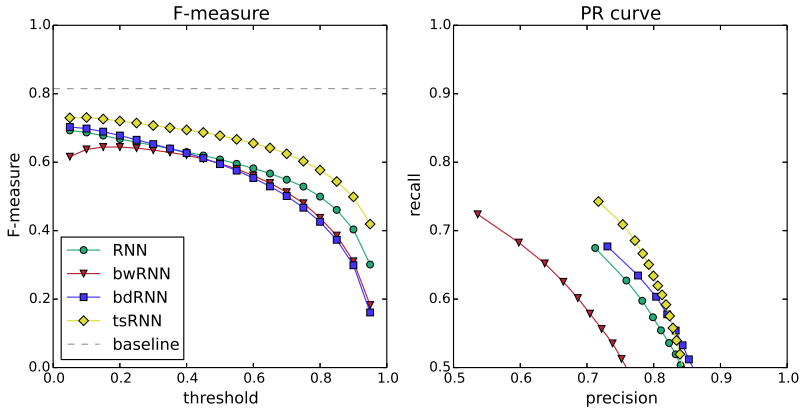


Figure 4. Result visualization for the evaluation on the ENST-Drums dataset. The left plot shows the F-measure curve, the right plot the precision-recall curve for different threshold levels for peak picking.

3.2.4 RNN with Time Shift

This approach is architecturally the same as the simple forward RNN, with the addition that this network can see more frames of the spectrogram to identify the instruments active at an onset. For training, the annotations are shifted into the future by 25ms and after transcription the detected onsets are shifted back by the same time. Doing this, the RNN can take a small portion of the sustain phase of the onset's spectrogram also into account. This is meant to improve the performance of the classification in the same way the backward connections do, without losing the real-time capabilities. The system is to a limited degree still real-time capable—depending on the length of the time shift. The used delay of 25ms in this work might still be sufficiently small for certain applications like score following and other visualizations and it can be tuned to meet the demands of certain applications.

3.3 Peak Picking

The output neurons of the RNNs provide activation functions for every instrument. To identify the instrument onsets, a simple and robust peak picking method designed for onset detection is used [2]. Peaks are selected at a frame n of the activation function $F(n)$ if the following three conditions are met:

1. $F(n) = max(F(n - pre\_max : n + post\_max))$,
2. $F(n) \geq mean(F(n - pre\_avg : n + post\_avg)) + \delta$,
3. $n - n_{lastpeak} > combination\_width$,

where δ is a threshold varied for evaluation. Simply put, a peak has to be the maximum of a certain window, and higher than the mean plus some threshold of another window. Additionally there has to be a distance of at least $combination\_width$ to the last peak. Parameters for the windows were chosen to achieve good results on a development data set while considering that 10 ms is the threshold of hearing two distinct events (values are converted from frames to ms): $pre\_max = 20ms$, $post\_max = 0ms$, $pre\_avg = 20ms$, $post\_avg = 0ms$, and $combination\_width = 20ms$. Setting $post\_max$ and $post\_avg$ to zero allows the application in online scenarios.

3.4 RNN Training

The task which has to be solved by the RNNs in this work is a three-way binary classification problem. When provided with the input spectrogram, the RNN has to identify the onsets of the three instrument classes by predicting the activation functions at the output neurons. The training algorithm has to adapt the weights and biases of the network in a way to achieve this functionality. In this work, the *rmsprop* method proposed by Hinton and Tieleman [31] is used as training algorithm. Additionally, dropout [30] between the recurrent and the output layer of the RNNs is used for training. When using dropout, randomly chosen connections are disabled for a single training iteration. The amount of disabled connections is determined by the dropout rate.

The goal of the training algorithm is to minimize a loss function. The loss function measures how much error the networks makes while reproducing the target functions. As loss function for training, the mean of the binary cross-entropy of the values of the three output neurons and the target functions is used (see Figure 2). The training with *rmsprop* is based on mini batches. In this work, mini batches with a size of eight instances were used. The training instances consist of 100-frame-segments of the extracted spectrogram. These are generated by extracting the spectrogram as described in Section 3.1 from the training files and cutting it into 100-frame-segments with 90 frames overlap (i.e. 10 frames hop-size). The order of the segments for training is randomized.

During one epoch the training data is used to adapt the weights and biases of the network. At the end of an epoch, the validation data is used to estimate the quality of the trained network. The training of the RNNs is aborted as soon as the resulting loss for the validation set has not decreased for 10 epochs. As learning rate decay strategy, the following method is applied: after every seven epochs the

Results for IDMT-SMT-Drums

algorithm	best F-measure[%]	at threshold
RNN	96.3	0.15
bwRNN	97.1	0.30
bdRNN	98.1	0.15
tsRNN	**98.2**	0.25
NMF [5]	**95.0**	-

Table 1. Evaluation results on the IDMT-SMT-Drums dataset. The NMF approach serves as state-of-the-art baseline.

Results for ENST-Drums

algorithm	best F-measure[%]	at threshold
RNN	69.3	0.05
bwRNN	64.4	0.15
bdRNN	70.3	0.05
tsRNN	**73.1**	0.10
HMM [24]	**81.5**	-

Table 2. Evaluation results on the ENST-Drums dataset. The HMM approach serves as state-of-the-art baseline.

learning rate is halved. For the simple RNN, backward RNN, and time shifted RNN the following parameter settings are used: initial learning rate $r_l = 0.001$ and dropout rate $r_d = 0.2$. In case of the bidirectional RNN the following parameter settings are used: initial learning rate $r_l = 0.0005$ and the dropout rate $r_d = 0.3$. The network is initialized with weights randomly sampled from a uniform distribution in the range ± 0.01, and zero-value biases.

All hyperparameters like network architecture, dropout rate, and learning rate were chosen according to empirical experimentation on a development data set, experience, and best practice examples.

3.5 Implementation

The implementation was done in Python using Lasagne [8] and Theano for RNN training and evaluation. The madmom [1] framework was used for signal processing and feature extraction, as well as for peak picking and evaluation metric calculation (precision, recall, and F-measure).

4. EVALUATION

To evaluate the presented system, the audio files of the test subset are preprocessed as explained in Section 3.1. Subsequently the spectrogram of the audio file is fed into the input layer of the RNN. The three neurons of the output layer provide the activation functions for the three instruments for which the peak picking algorithm then identifies the relevant peaks. These peaks are interpreted as instrument onsets. The true positive and false positive onsets are then identified by using a 20 ms tolerance window. It should be noted that in the state-of-the-art methods for the ENST-Drums dataset [24] as well as for the IDMT-SMT-Drums dataset [5], less strict tolerance windows of 30 ms and 50 ms, respectively, are used. Using these values, precision, recall, and F-measure for the onsets are calculated.

4.1 Datasets

For training and evaluation the IDMT-SMT-Drums [5] dataset was used. Some missing annotations have been added and additionally annotations for the *#train* tracks have been created. The *#train* tracks are tracks containing separated strokes of the individual instruments. These are only used as additional training examples and not used in the test set, to maintain a fair comparison with the results in [5]. The dataset was split into train, validation, and test subsets using 70%, 15%, and 15% of the files, respectively.

Additionally, the audio portion of the ENST-Drums [12] dataset was used as a second independent dataset to evaluate the generalization capabilities of the RNNs. From this dataset, the wet mixes of the drum-only tracks of all three drummers were used. Since all models were trained on the IDMT-SMT-Drums dataset, no splitting of this dataset was necessary.

For both datasets the three instruments' target functions are created by calculating the correct active frames (for a target frame rate of 100 Hz) using the annotations for each instrument. The target functions are one at the frames in which an annotation is present and zero otherwise. See Figure 2 for a visualization of the target functions in the context of the input spectrogram.

4.2 Experiments

For all four architectures, two different experiments were performed. First, the model was trained using the training and validation subsets of the IDMT-SMT-Drums dataset. Then, using the trained model, the tracks of the test split of the dataset were transcribed and the resulting precision, recall, and F-measure were calculated. Second, the trained model was evaluated by transcribing the ENST-Drums dataset and calculating the validation metrics. This was done to evaluate how well the trained models are able to generalize and if the models are over-fitted to the training dataset. Since the ENST-Drums dataset contains more than just the three instruments with which the model was trained, only the snare, bass, and hi-hat annotations were used. This makes it on the one hand easier to identify all annotated notes, on the other hand, there are some percussive onsets in the audio, which should not be transcribed and which are counted as false positives if the network falsely interprets them as snare, bass, or hi-hat hits. The percentage of snare, bass, and hi-hat annotations is 81.2% (i.e., 18.8% are other instruments which are ignored and potential false positives). The ENST-Drums dataset contains more expressive and faster drumming styles than the IDMT-SMT-Drums dataset, making it a more difficult dataset to transcribe. This fact is reflected in the transcription performances of both the state-of-the-art algorithms as well as the proposed methods. This behavior can also be observed in the work of Wu and Lerch [32] who apply their method to both datasets.

5. RESULTS AND DISCUSSION

Table 1 summarizes the results of all methods on the IDMT-SMT-Drums dataset. It can be seen that the F-measure values for all RNNs are higher than the state-of-the-art. It should be noted at this point, that the approach presented in [5] was introduced for real-time transcription. Nevertheless, the used NMF approach uses audio samples of the exact same instruments which should be transcribed as prototypes, which is the best-case scenario for NMF transcription. In contrast, our approach is trained on a split of the full dataset which contains many different instrument sounds, and thus is a more general model than the one used in the state-of-the-art approach used as baseline. It can be observed that the backward RNN performs slightly better than the plain RNN, which indicates that indeed the short sustain phases of the drum instruments contain information which is useful for classification. The bidirectional RNN again performs slightly better than the backward RNN, which comes as no surprise since it combines the properties of the plain forward and backward RNNs. The results of the forward RNN with time shift are not significantly different from the results of the bidirectional RNN. This indicates that the short additional time frame provided by the time shift provides sufficient additional information to achieve similar classification results as with a bidirectional RNN. Figure 3 shows a F-measure curve as well as a precision-recall curve for different threshold levels for peak picking.

The results of the evaluation of the models trained on the IDMT-SMT-Drums dataset used to transcribe the ENST-Drums dataset are shown in Table 2. The achieved F-measure values are not as high as the state-of-the-art in this case but this was expected. In contrast to [24], the model used in this work is not trained on splits of the ENST-Drums dataset and thus not optimized for it. Nevertheless, reasonable high F-measure values are achieved with respect to the fact that the model was trained on completely different and more simple data. This can be interpreted as an indication that the model in fact learns, to some degree, general properties of the three different drum instruments. Figure 4 shows an F-measure curve as well as a precision-recall curve for different threshold levels for peak picking.

In Figures 3 and 4, it can be seen that the highest F-measure values are found for low values for the threshold of the peak picking algorithm. This suggests that the RNNs are quite selective and the predicted activation functions do not contain much noise—which can in fact be observed (see Figure 2). This further implies that choices for peak picking window sizes are not critical, which was also observed in empiric experiments.

6. FUTURE WORK

Next steps for using RNNs for drum transcription will involve adapting the method to work on polyphonic audio tracks. It can be imagined to combine the presented method with a harmonic/percussive separation stage, using, e.g., the method introduced by Fitzgerald et al. [10], which would yield a drum track transcript from a full polyphonic audio track. As we show in this work, the transcription methods using RNNs are quite selective and therefore expected to be robust regarding artifacts resulting from source separation. Training directly on full audio tracks may also be a viable option to work on full audio tracks.

Another option is to use more instrument classes than the three instruments used in this and many other works. Theoretically, RNNs are not as vulnerable as source separation approaches when it comes to the number of instruments to transcribe. It has been shown that RNNs can perform well when using a much greater number of output neurons, for example 88 neurons in the case of piano transcription [4]. Although, for this, a dataset which has a balanced amount of notes played by different instruments has to be created first.

7. CONCLUSION

In this work, four architectures for drum transcription methods of solo drum tracks using RNNs were introduced. Their transcription performances are better than the results of the state-of-the-art approach which uses an NMF method—even with the NMF approach having the advantage of being trained on exactly the same instruments used in the drum tracks. The RNN approaches seem to be able to generalize quite well, since reasonable high transcription results are yielded on another, independent, and more difficult dataset. The precision-recall curves show that the best results are obtained when using a low threshold for peak picking. This implies that the used transcription methods are quite selective, which is an indication that they are robust and not bound to be influenced by noise or artifacts when using additional preprocessing steps.

8. ACKNOWLEDGMENTS

This work is supported by the European Union's Seventh Framework Programme FP7 / 2007-2013 for research, technological development and demonstration under grant agreement no. 610591 (GiantSteps). This work is supported by the Austrian ministries BMVIT and BMWFW, and the province of Upper Austria via the COMET Center SCCH. We gratefully acknowledge the support of NVIDIA Corporation with the donation of one Titan Black and one Tesla K40 GPU used for this research.

9. REFERENCES

[1] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. madmom: a new python audio and music signal processing library. `https://arxiv.org/abs/1605.07008`, 2016.

[2] Sebastian Böck, Florian Krebs, and Markus Schedl. Evaluating the online capabilities of onset detection methods. In *Proc 13th Intl Soc for Music Information Retrieval Conf*, pages 49–54, Porto, Portugal, October 2012.

[3] Sebastian Böck and Markus Schedl. Enhanced beat tracking with context-aware neural networks. In *Proc 14th Conf on Digital Audio Effects*, Paris, France, September 2011.

[4] Sebastian Böck and Markus Schedl. Polyphonic piano note transcription with recurrent neural networks. In *Proc IEEE Intl Conf on Acoustics, Speech and Signal Processing*, pages 121–124, Kyoto, Japan, March 2012.

[5] Christian Dittmar and Daniel Gärtner. Real-time transcription and separation of drum recordings based on nmf decomposition. In *Proc 17th Intl Conf on Digital Audio Effects*, Erlangen, Germany, September 2014.

[6] Christian Dittmar and Christian Uhle. Further steps towards drum transcription of polyphonic music. In *Proc 116th Audio Engineering Soc Conv*, Berlin, Germany, May 2004.

[7] Simon Dixon, Fabien Gouyon, Gerhard Widmer, et al. Towards characterisation of music via rhythmic patterns. In *Proc 5th Intl Conf on Music Information Retrieval*, Barcelona, Spain, October 2004.

[8] Sander Dieleman et al. Lasagne: First release. `http://dx.doi.org/10.5281/zenodo.27878`, 2015.

[9] Derry FitzGerald, Robert Lawlor, and Eugene Coyle. Prior subspace analysis for drum transcription. In *Proc 114th Audio Engineering Soc Conf*, Amsterdam, Netherlands, March 2003.

[10] Derry FitzGerald, Antoine Liukus, Zafar Rafii, Bryan Pardo, and Laurent Daudet. Harmonic/percussive separation using kernel additive modelling. In *Irish Signals & Systems Conf and China-Ireland Intl Conf on Information and Communications Technologies*, pages 35–40, Limerick, Ireland, June 2014.

[11] Olivier Gillet and Gaël Richard. Drum loops retrieval from spoken queries. *Journal of Intelligent Information Systems*, 24(2-3):159–177, 2005.

[12] Olivier Gillet and Gaël Richard. Enst-drums: an extensive audio-visual database for drum signals processing. In *Proc 7th Intl Conf on Music Information Retrieval*, pages 156–159, Victoria, BC, Canada, October 2006.

[13] Olivier Gillet and Gaël Richard. Transcription and separation of drum signals from polyphonic music. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(3):529–540, 2008.

[14] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proc 14th Intl Conf on Artificial Intelligence and Statistics*, pages 315–323, Ft. Lauderdale, FL, USA, April 2011.

[15] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for improved unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5), 2009.

[16] Marko Helen and Tuomas Virtanen. Separation of drums from polyphonic music using non-negative matrix factorization and support vector machine. In *Proc 13th European Signal Processing Conf*, Antalya, Turkey, September 2005.

[17] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, July 2006.

[18] Kristoffer Jensen. Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal on Applied Signal Processing*, 2007(1):159–159, 2007.

[19] Ajay Kapur, Manj Benning, and George Tzanetakis. Query-by-beat-boxing: Music retrieval for the dj. In *Proc 5th Intl Conf on Music Information Retrieval*, pages 170–177, Barcelona, Spain, October 2004.

[20] David Krueger and Roland Memisevic. Regularizing rnns by stabilizing activations. In *Proc 4th Intl Conf on Learning Representations*, San Juan, Puerto Rico, May 2015.

[21] Henry Lindsay-Smith, Skot McDonald, and Mark Sandler. Drumkit transcription via convolutive nmf. In *Intl Conf on Digital Audio Effects*, York, UK, September 2012.

[22] Marius Miron, Matthew EP Davies, and Fabien Gouyon. Improving the real-time performance of a causal audio drum transcription system. In *Proc Sound and Music Computing Conf*, pages 402–407, Stockholm, Sweden, July 2013.

[23] Marius Miron, Matthew EP Davies, and Fabien Gouyon. An open-source drum transcription system for pure data and max msp. In *Proc IEEE Intl Conf on Acoustics, Speech and Signal Processing*, pages 221–225, Vancouver, BC, Canada, May 2013.

[24] Jouni Paulus and Anssi Klapuri. Drum sound detection in polyphonic music with hidden markov models. *EURASIP Journal on Audio, Speech, and Music Processing*, 2009.

[25] Haşim Sak, Andrew W. Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Proc 15th Annual Conf of the Intl Speech Communication Association*, pages 338–342, Singapore, September 2014.

[26] Simon Scholler and Hendrik Purwins. Sparse approximations for drum sound classification. *IEEE Journal of Selected Topics in Signal Processing*, 5(5):933–940, 2011.

[27] Siddharth Sigtia, Emmanouil Benetos, Nicolas Boulanger-Lewandowski, Tillman Weyde, Artur S d'Avila Garcez, and Simon Dixon. A hybrid recurrent neural network for music transcription. In *Proc IEEE Intl Conf on Acoustics, Speech and Signal Processing*, pages 2061–2065, Brisbane, Australia, April 2015.

[28] Paris Smaragdis. Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs. In *Independent Component Analysis and Blind Signal Separation*, volume 3195 of *Lecture Notes in Computer Science*, pages 494–499. Springer, Granada, Spain, September 2004.

[29] Andrio Spich, Massimiliano Zanoni, Augusto Sarti, and Stefano Tubaro. Drum music transcription using prior subspace analysis and pattern recognition. In *Proc 13th Intl Conf of Digital Audio Effects*, Graz, Austria, September 2010.

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, June 2014.

[31] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5rmsprop: Divide the gradient by a running average of its recent magnitude. In *COURSERA: Neural Networks for Machine Learning*, October 2012.

[32] Chih-Wei Wu and Alexander Lerch. Drum transcription using partially fixed non-negative matrix factorization with template adaptation. In *Proc 16th Intl Soc for Music Information Retrieval Conf*, pages 257–263, Málaga, Spain, October 2015.

[33] Kazuyoshi Yoshii, Masataka Goto, and Hiroshi G. Okuno. Automatic drum sound description for real-world music using template adaptation and matching methods. In *Proc 5th Intl Conf on Music Information Retrieval*, pages 184–191, Barcelona, Spain, October 2004.

SINGING VOICE MELODY TRANSCRIPTION USING DEEP NEURAL NETWORKS

François Rigaud and Mathieu Radenen
Audionamix R&D
171 quai de Valmy, 75010 Paris, France
<firstname>.<lastname>@audionamix.com

ABSTRACT

This paper presents a system for the transcription of singing voice melodies in polyphonic music signals based on Deep Neural Network (DNN) models. In particular, a new DNN system is introduced for performing the f_0 estimation of the melody, and another DNN, inspired from recent studies, is learned for segmenting vocal sequences. Preparation of the data and learning configurations related to the specificity of both tasks are described. The performance of the melody f_0 estimation system is compared with a state-of-the-art method and exhibits highest accuracy through a better generalization on two different music databases. Insights into the global functioning of this DNN are proposed. Finally, an evaluation of the global system combining the two DNNs for singing voice melody transcription is presented.

1. INTRODUCTION

The automatic transcription of the main melody from polyphonic music signals is a major task of Music Information Retrieval (MIR) research [19]. Indeed, besides applications to musicological analysis or music practice, the use of the main melody as prior information has been shown useful in various types of higher-level tasks such as music genre classification [20], music retrieval [21], music desoloing [4, 18] or lyrics alignment [15, 23]. From a signal processing perspective, the main melody can be represented by sequences of fundamental frequency (f_0) defined on voicing instants, *i.e.* on portions where the instrument producing the melody is active. Hence, main melody transcription algorithms usually follow two main processing steps. First, a representation emphasizing the most likely f_0s over time is computed, *e.g.* under the form of a salience matrix [19], a vocal source activation matrix [4] or an enhanced spectrogram [22]. Second, a binary classification of the selected f_0s between melodic and background content is performed using melodic contour detection/tracking and voicing detection.

© François Rigaud and Mathieu Radenen. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** François Rigaud and Mathieu Radenen. "Singing Voice Melody Transcription using Deep Neural Networks", 17th International Society for Music Information Retrieval Conference, 2016.

In this paper we propose to tackle the melody transcription task as a supervised classification problem where each time frame of signal has to be assigned into a pitch class when a melody is present and an 'unvoiced' class when it is not. Such approach has been proposed in [5] where melody transcription is performed applying Support Vector Machine on input features composed of Short-Time Fourier Transforms (STFT). Similarly for noisy speech signals, f_0 estimation algorithms based on Deep Neural Networks (DNN) have been introduced in [9, 12].

Following such fully data driven approaches we introduce a singing voice melody transcription system composed of two DNN models respectively used to perform the f_0 estimation task and the Voice Activity Detection (VAD) task. The main contribution of this paper is to present a DNN architecture able to discriminate the different f_0s from low-level features, namely spectrogram data. Compared to a well-known state-of-the-art method [19], it shows significant improvements in terms of f_0 accuracy through an increase of robustness with regard to musical genre and a reduction of octave-related errors. By analyzing the weights of the network, the DNN is shown somehow equivalent to a simple harmonic-sum method for which the parameters usually set empirically are here automatically learned from the data and where the succession of non-linear layers likely increases the power of discrimination of harmonically-related f_0. For the task of VAD, another DNN model, inspired from [13] is learned. For both models, special care is taken to prevent over-fitting issues by using different databases and perturbing the data with audio degradations. Performance of the whole system is finally evaluated and shows promising results.

The rest of the paper is organized as follows. Section 2 presents an overview of the whole system. Sections 3 and 4 introduce the DNN models and detail the learning configurations respectively for the VAD and the f_0 estimation task. Then, Section 5 presents an evaluation of the system and Section 6 concludes the study.

2. SYSTEM OVERVIEW

2.1 Global architecture

The proposed system, displayed on Figure 1, is composed of two independent parallel DNN blocks that perform respectively the f_0 melody estimation and the VAD.

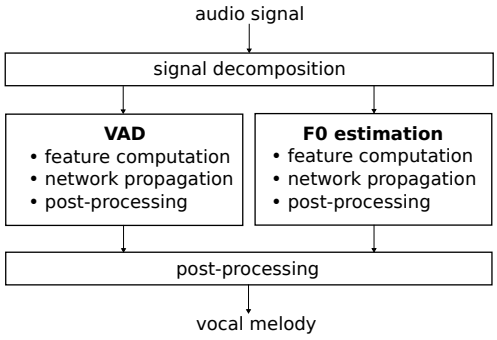


Figure 1: Architecture of the proposed system for singing voice melody transcription.

In contrast with [9,12] that propose a single DNN model to perform both tasks, we did not find such unified functional architecture able to discriminate successfully a time frame between quantified f_0s and 'unvoiced' classes. Indeed the models presented in these studies are designed for speech signals mixed with background noise for which the discrimination between a frame of noise and a frame of speech is very likely related to the presence or absence of a pitched structure, which is also probably the kind of information on which the system relies to estimate the f_0. Conversely, with music signals both the melody and the accompaniment exhibit harmonic structures and the voicing discrimination usually requires different levels of information, *e.g.* under the form of timbral features such as Mel-Frequency Cepstral Coefficients.

Another characteristic of the proposed system is the parallel architecture that allows considering different types of input data for the two DNNs and which arises from the application restricted to vocal melodies. Indeed, unlike generic systems dealing with main melody transcription of different instruments (often within a same piece of music) which usually process the f_0 estimation and the voicing detection sequentially, the focus on singing voice here hardly allows for a voicing detection relying only on the distribution and statistics of the candidate pitch contours and/or their energy [2, 19]. Thus, this constraint requires to build a specific VAD system that should learn to discriminate the timbre of a vocal melody from an instrumental melody, such as for example played by a saxophone.

2.2 Signal decomposition

As shown on Figure 1, both DNN models are preceded by a signal decomposition. At the input of the global system, audio signals are first converted to mono and re-sampled to 16 kHz. Then, following [13], it is proposed to provide the DNNs with a set of pre-decomposed signals obtained by applying a double-stage Harmonic/Percussive Source Separation (HPSS) [6,22] on the input mixture signal. The key idea behind double-stage HPSS is to consider that within a mix, melodic signals are usually less stable/stationary than the background 'harmonic' instruments (such as a bass or a piano), but more than the percussive instruments (such as the drums). Thus, according to the frequency resolution that is used to compute a STFT, applying a harmonic/percussive decomposition on a mixture spectrogram lead to a rough separation where the melody is mainly extracted either in the harmonic or in the percussive content.

Using such pre-processing, 4 different signals are obtained. First, the input signal s is decomposed into the sum of h_1 and p_1 using a high-frequency resolution STFT (typically with a window of about 300 ms) where p_1 should mainly contain the melody and the drums, and h_1 the remaining stable instrument signals. Second, p_1 is further decomposed into the sum of h_2 and p_2 using a low-frequency resolution STFT (typically with a window of about 30 ms), where h_2 mainly contains the melody, and p_2 the drums. As presented latter in Sections 3 and 4, different types of these 4 signals or combinations of them will be used to experimentally determine optimal DNN models.

2.3 Learning data

Several annotated databases composed of polyphonic music with transcribed melodies are used for building the train, validation and test datasets used for the learning (*cf.* Sections 3 and 4) and the evaluation (*cf.* Section 5) of the DNNs. In particular, a subset of RWC Popular Music and Royalty Free Music [7] and MIR-1k [10] databases are used for the train dataset, and the recent databases MedleyDB [1] and iKala [3] are split between train, validation and test datasets. Note that for iKala the vocal and instrumental tracks are mixed with a relative gain of 0 dB.

Also, in order to minimize over-fitting issues and to increase the robustness of the system with respect to audio equalization and encoding degradations, we use the Audio Degradation Toolbox [14]. Thus, several files composing the train and validation datasets (50% for the VAD task and 25% for the f_0 estimation task) are duplicated with one degraded version, the degradation type being randomly chosen amongst those available preserving the alignment between the audio and the annotation (*e.g.* not producing time/pitch warping or too long reverberation effects).

3. VOICE ACTIVITY DETECTION WITH DEEP NEURAL NETWORKS

This section briefly describes the process for learning the DNN used to perform the VAD. It is largely inspired from a previous study presented in more detail in [13]. A similar architecture of deep recurrent neural network composed of Bidirectional Long Short-Term Memory (BLSTM) [8] is used. In our case the architecture is arbitrarily fixed to 3 BLSTM layers of 50 units each and a final feed-forward logistic output layer with one unit. As in [13], different types of combination of the pre-decomposed signals (*cf.* Section 2.2) are considered to determine an optimal network: s, p_1, h_2, h_1p_1, h_2p_2 and $h_1h_2p_2$. For each of these pre-decomposed signals, timbral features are computed under the form of mel-frequency spectrograms obtained using a STFT with 32 ms long Hamming windows and 75 % of overlap, and 40 triangular filters distributed on a mel scale between 0 and 8000 Hz. Then, each feature of the input

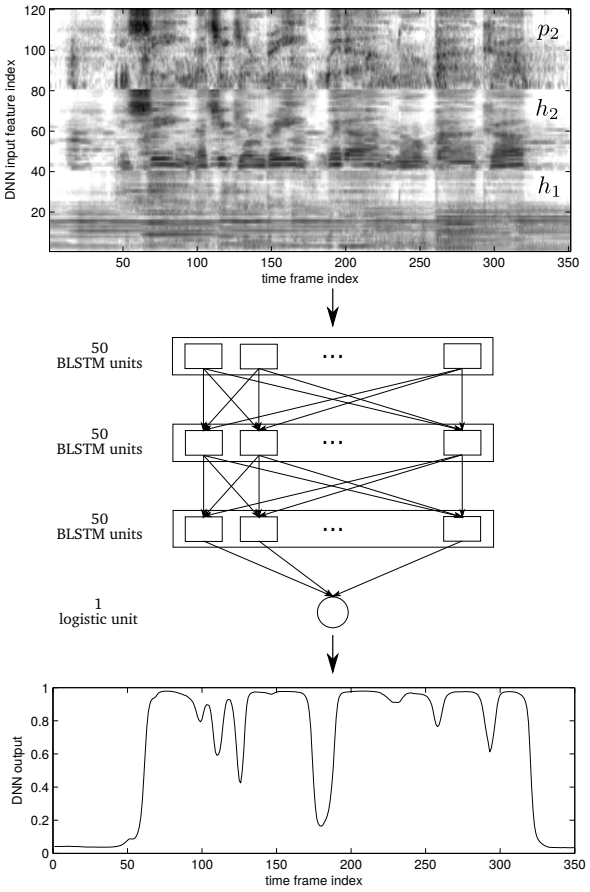


Figure 2: VAD network illustration.

data is normalized using the mean and variance computed over the train dataset. Contrary to [13] the learning is performed in a single step, *i.e.* without adopting a layer by layer training.

Finally, the best architecture is obtained for the combination of h_1, h_2 and p_2 signals, thus for an input of size 120, which corresponds to a use of the whole information present in the original signal ($s = h_1 + h_2 + p_2$). An illustration of this network is presented in Figure 2.

A simple post-processing of the DNN output consisting in a threshold of 0.5 is finally applied to take the binary decision of voicing frame activation.

4. F_0 ESTIMATION WITH DEEP NEURAL NETWORKS

This section presents in detail the learning configuration for the DNN used for performing the f_0 estimation task. An interpretation of the network functioning is finally presented.

4.1 Preparation of learning data

As proposed in [5] we decide to keep low level features to feed the DNN model. Compared to [12] and [9] which use as input pre-computed representations known for highlighting the periodicity of pitched sounds (respectively based on an auto-correlation and a harmonic filtering), we expect here the network to be able to learn an optimal transformation automatically from spectrogram data. Thus the set of selected features consists of log-spectrograms (logarithm of the modulus of the STFT) computed from a Hamming window of duration 64 ms (1024 samples for a sampling frequency of 16000 Hz) with an overlap of 0.75, and from which frequencies below 50 Hz and above 4000 Hz are discarded. For each music excerpt the corresponding log-spectrogram is rescaled between 0 and 1. Since, as described in Section 2.1, the VAD is performed by a second independent system, all time frames for which no vocal melody is present are removed from the dataset. These features are computed independently for 3 different types of input signal for which the melody should be more or less emphasized: s, p_1 and h_2 (*cf.* Section 2.2).

For the output, the f_0s are quantified between C#2 ($f_0 \simeq 69.29$ Hz) and C#6 ($f_0 \simeq 1108.73$ Hz) with a spacing of an eighth of tone, thus leading to a total of 193 classes.

The train and validation datasets including audio degraded versions are finally composed of, respectively, 22877 melodic sequences (resp. 3394) for a total duration of about 220 minutes (resp. 29 min).

4.2 Training

Several experiments have been run to determine a functional DNN architecture. In particular, two types of neuron units have been considered: the standard feed-forward sigmoid unit and the Bidirectional Long Short-Term Memory (BLSTM) recurrent unit [8].

For each test, the weights of the network are initialized randomly according to a Gaussian distribution with 0 mean and a standard deviation of 0.1, and optimized to minimize the cross-entropy error function. The learning is then performed by means of a stochastic gradient descent with shuffled mini-batches composed of 30 melodic sequences, a learning rate of 10^{-7} and a momentum of 0.9. The optimization is run for a maximum of 10000 epochs and an early stopping is applied if no decrease is observed on the validation set error during 100 consecutive epochs. In addition to the use of audio degradations during the preparation of the data for preventing over-fitting (*cf.* Section 2.3), the training examples are slightly corrupted during the learning by adding a Gaussian noise with variance 0.05 at each epoch.

Among the different architectures tested, the best classification performance is obtained for the input signal p_1 (slightly better than for s, *i.e.* without pre-separation) by a 2-hidden layer feed-forward network with 500 sigmoid units each, and a 193 output softmax layer. An illustration of this network is presented in Figure 3. Interestingly, for that configuration the learning did not suffered from overfitting so that it ended at the maximum number of epochs, thus without early stopping.

While the temporal continuity of the f_0 along timeframes should provide valuable information, the use of BLSTM recurrent layers (alone or in combination with

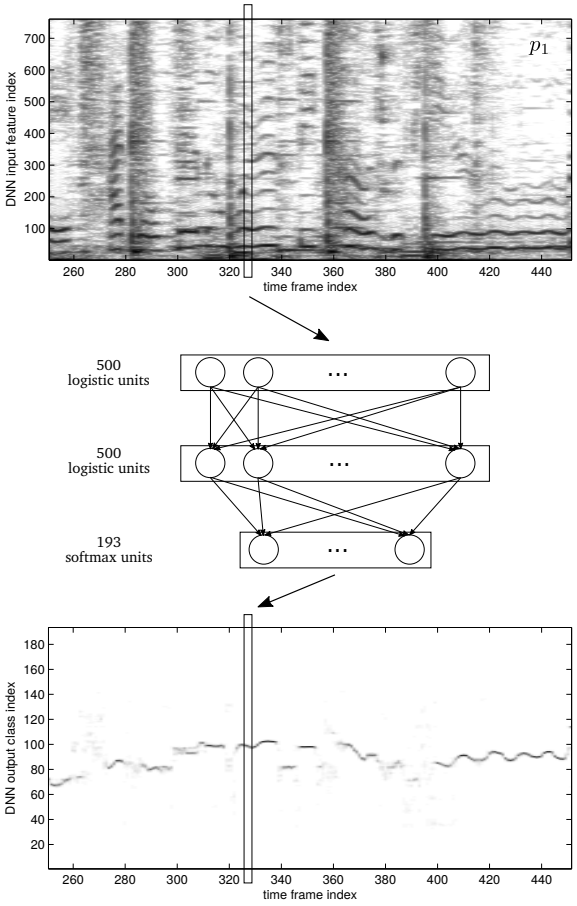


Figure 3: f_0 estimation network illustration.

feed-forward sigmoid layers) did not lead to efficient systems. Further experiments should be conducted to enforce the inclusion of such temporal context in a feed-forward DNN architecture, for instance by concatenating several consecutive time frames in the input.

4.3 Post-processing

The output layer of the DNN composed of softmax units returns a f_0 probability distribution for each time frame that can be seen for a full piece of music as a pitch activation matrix. In order to take a final decision that account for the continuity of the f_0 along melodic sequences, a Viterbi tracking is finally applied on the network output [5, 9, 12]. For that, the log-probability transition between two consecutive time frames and two f_0 classes is simply arbitrarily set inversely proportional to their absolute difference in semi-tones. For further improvement of the system, such transition matrix could be learned from the data [5], however this simple rule gives interesting performance gains (when compared to a simple 'maximum picking' post-processing without temporal context) while potentially reducing the risk of over-fitting to a particular music style.

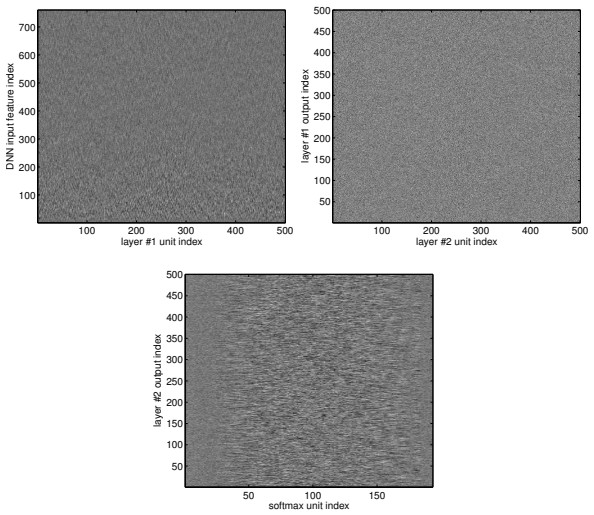


Figure 4: Display of the weights for the two sigmoid feed-forward layers (top) and the softmax layer (down) of the DNN learned for the f_0 estimation task.

4.4 Network weights interpretation

We propose here to have an insight into the network functioning for this specific task of f_0 estimation by analyzing the weights of the DNN. The input is a short-time spectrum and the output corresponds to an activation vector for which a single element (the actual f_0 of the melody at that time frame) should be predominant. In that case, it is reasonable to expect that the DNN somehow behaves like a harmonic-sum operator.

While the visualization of the distribution of the hidden-layer weights usually does not provide with straightforward cues to analyse a DNN functioning (*cf.* Figure 4) we consider a simplified network for which it is assumed that each feed-forward logistic unit is working in the linear regime. Thus, removing the non-linear operations, the output of a feed-forward layer with index l composed of N_l units writes

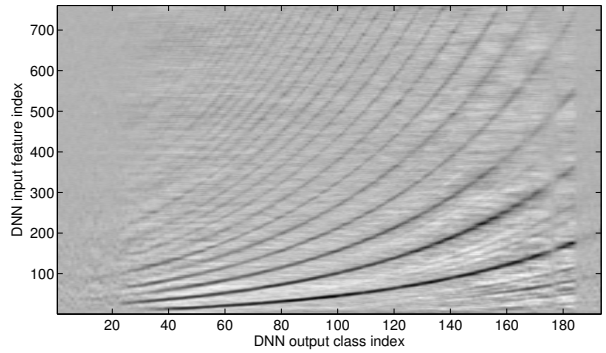
$$x_l = W_l \cdot x_{l-1} + b_l, \tag{1}$$

where $x_l \in \mathbb{R}^{N_l}$ (resp. $x_{l-1} \in \mathbb{R}^{N_{l-1}}$) corresponds to the ouput vector of layer l (resp. $l-1$), $W_l \in \mathbb{R}^{N_l \times N_{l-1}}$ is the weight matrix and $b_l \in \mathbb{R}^{N_l}$ the bias vector. Using this expression, the output of a layer with index L expressed as the propagation of the input x_0 through the linear network also writes

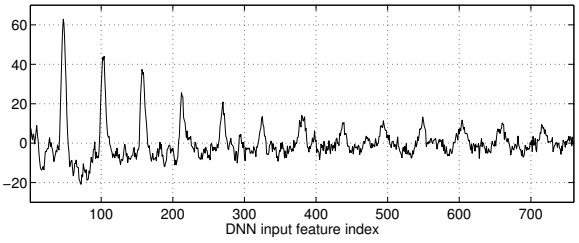
$$x_L = \mathbf{W} \cdot x_0 + \mathbf{b}, \tag{2}$$

where $\mathbf{W} = \prod_{l=1}^{L} W_l$ corresponds to a global weight matrix, and $\mathbf{b}$ to a global bias that depends on the set of parameters $\{W_l, b_l, \forall l \in [1..L]\}$.

As mentioned above, in our case x_0 is a short-time spectrum and x_L is a f_0 activation vector. The global weight matrix should thus present some characteristics of a pitch detector. Indeed as displayed on Figure 5a, the matrix $\mathbf{W}$ for the learned DNN (which is thus the product of the 3 weight matrices depicted on Figure 4) exhibits an harmonic structure for most output classes of f_0s; except for



(a)



(b)

Figure 5: Linearized DNN illustration. (a) Visualization of the (transposed) weight matrix $\mathbf{W}$. The x-axis corresponds to the output class indices (the f_0s) and the y-axis represents the input feature indices (frequency channel of the spectrum input). (b) Weights display for the f_0 output class with index 100.

some f_0s in the low and high frequency range for which no or too few examples are present in the learning data.

Most approaches dealing with main melody transcription usually relies on such types of transformations to compute a representation emphasizing f_0 candidates (or salience function) and are usually partly based on handcrafted designs [11, 17, 19]. Interestingly, using a fully data driven method as proposed, parameters of a comparable weighted harmonic summation algorithm (such as the number of harmonics to consider for each note and their respective weights) do not have to be defined. This can be observed in more details on Figure 5b which depicts the linearized network weights for the class index 100 ($f_0 \simeq$ 289.43 Hz). Moreover, while this interpretation assumes a linear network, one can expect that the non-linear operations actually present in the network help in enhancing the discrimination between the different f_0 classes.

5. EVALUATION

5.1 Experimental procedure

Two different test datasets composed of full music excerpts (*i.e.* vocal and non vocal portions) are used for the evaluation. One is composed of 17 tracks from MedleyDB (last songs comprising vocal melodies, from *MusicDelta_Reggae* to *Wolf_DieBekherte*, for a total of $\sim$ 25.5 min of vocal portions) and the other is composed of 63 tracks from iKala (from *54223_chorus* to *90587_verse* for a total of $\sim$ 21 min of vocal portions).

The evaluation is conducted in two steps. First the performance of the f_0 estimation DNN taken alone (thus without voicing detection) is compared with the state-of the art system *melodia* [19] using f_0 accuracy metrics. Second, the performance of our complete singing voice transcription system (VAD and f_0 estimation) is evaluated on the same datasets. Since our system is restricted to the transcription of vocal melodies and that, to our knowledge all available state-of-the-art systems are designed to target main melody, this final evaluation presents the results for our system without comparisons with a reference.

For all tasks and systems, the evaluation metrics are computed using the *mir_eval* library [16]. For Section 5.3, some additional metrics related to voicing detection, namely precision, f-measure and voicing accuracy, were not present in the original *mir_eval* code and thus were added for our experiments.

5.2 f_0 estimation task

The performance of the DNN performing the f_0 estimation task is first compared to *melodia* system [19] using the plug-in implementation with f_0 search range limits set equal to those of our system (69.29-1108.73 Hz, *cf.* Sec. 4.1) and with remaining parameters left to default values. For each system and each music track the performance is evaluated in terms of raw pitch accuracy (RPA) and raw chroma accuracy (RCA). These metrics are computed on vocal segments (*i.e.* without accounting for potential voicing detection errors) for a f_0 tolerance of 50 cents.

The results are presented on Figure 6 under the form of a box plot where, for each metric and dataset, the ends of the dashed vertical bars delimit the lowest and highest scores obtained, the 3 vertical bars composing each center box respectively correspond to the first quartile, the median and the third quartile of the distribution, and finally the star markers represent the mean. Both systems are characterized by more widespread distributions for MedleyDB than for iKala. This reflects the fact that MedleyDB is more heterogeneous in musical genres and recording conditions than iKala. On iKala, the DNN performs slightly better than *melodia* when comparing the means. On MedleyDB, the gap between the two systems increases significantly. The DNN system seems much less affected by the variability of the music examples and clearly improve the mean RPA by 20% (62.13% for *melodia* and 82.48% for the DNN). Additionally, while exhibiting more similar distributions of RPA and RCA, the DNN tends to produce less octave detection errors. It should be noted that this result does not take into account the recent post-processing improvement proposed for *melodia* [2], yet it shows the interest of using such DNN approach to compute an enhanced pitch salience matrix which, simply combined with a Viterbi post-processing, achieves good performance.

5.3 Singing voice transcription task

The evaluation of the global system is finally performed on the two same test datasets. The results are displayed as

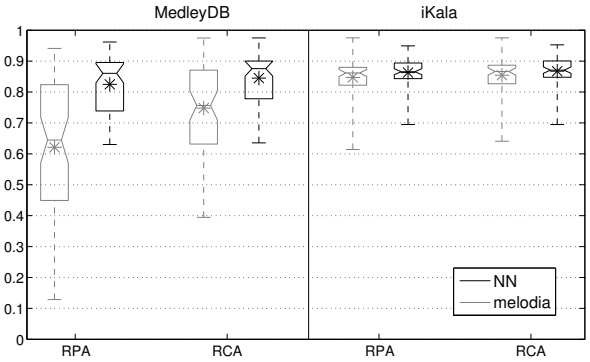


Figure 6: Comparative evaluation of the proposed DNN (in black) and *melodia* (in gray) on MedleyDB (left) and iKala (right) test sets for a f_0 vocal melody estimation task.

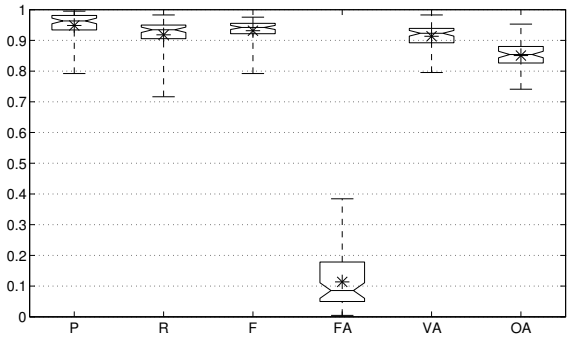
boxplots (*cf.* description Section 5.2) on Figures 7a and 7b respectively for the iKala and the MedleyDB datasets. Five metrics are computed to evaluate the voicing detection, namely the precision (P), the recall (R), the f-measure (F), the false alarm rate (FA) and the voicing accuracy (VA). A sixth metric of overall accuracy (OA) is also presented for assessing the global performance of the complete singing voice melody transcription system.

In accordance with the previous evaluation, the results on MedleyDB are characterized by much more variance than on iKala. In particular, the voicing precision of the system (*i.e.* it's ability to provide correct detections, no matter the number of forgotten voiced frames) is significantly degraded on MedleyDB. Conversely, the voicing recall which evaluate the ability of the system to detect all voiced portions actually present no matter the number of false alarm, remains relatively good on MedleyDB. Combining both metrics, a mean f-measure of 93.15 % and 79.19 % are respectively obtained on iKala and MedleyDB test datasets.

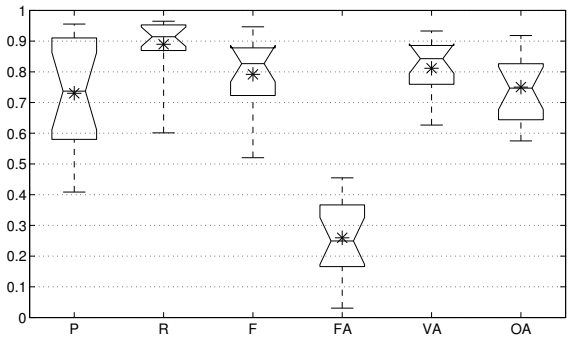
Finally, the mean scores of overall accuracy obtained for the global system are equal to 85.06 % and 75.03 % respectively for iKala and MedleyDB databases.

6. CONCLUSION

This paper introduced a system for the transcription of singing voice melodies composed of two DNN models. In particular a new system able to learn a representation emphasizing melodic lines from low level data composed of spectrograms has been proposed for the estimation of the f_0. For this DNN, the performance evaluation shows a relatively good generalization (when compared to a reference system) on two different test datasets and an increase of robustness to western music recordings that tend to be representative of the current music industry productions. While for these experiments the systems have been learned from a relatively low amount of data, the robustness, particularly for the task of VAD, could very likely be improved by increasing the number of training examples.



(a) iKala test dataset



(b) MedleyDB test dataset

Figure 7: Voicing detection and overall performance of the proposed system for iKala and MedleyDB test datasets.

7. REFERENCES

[1] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. Bello. MedleyDB: A multitrack dataset for annotation-intensive MIR research. In *Proc. of the 15th Int. Society for Music Information Retrieval (ISMIR) Conference*, October 2014.

[2] R. M. Bittner, J. Salamon, S. Essid, and J. P. Bello. Melody extraction by contour classification. In *Proc. of the 16th Int. Society for Music Information Retrieval (ISMIR) Conference*, October 2015.

[3] T.-S. Chan, T.-C. Yeh, Z.-C. Fan, H.-W. Chen, L. Su, Y.-H. Yang, and R. Jang. Vocal activity informed singing voice separation with the ikala dataset. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 718–722, April 2015.

[4] J.-L. Durrieu, G. Richard, and B. David. An iterative approach to monaural musical mixture de-soloing. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 105–108, April 2009.

[5] D. P. W. Ellis and G. E. Poliner. Classification-based melody transcription. *Machine Learning*, 65(2):439–456, 2006.

[6] D. FitzGerald and M. Gainza. Single channel vocal separation using median filtering and factorisation techniques. *ISAST Trans. on Electronic and Signal Processing*, 4(1):62–73, 2010.

[7] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. Rwc music database: Popular, classical, and jazz music databases. In *Proc. of the 3rd Int. Society for Music Information Retrieval (ISMIR) Conference*, pages 287–288, October 2002.

[8] A. Graves, A.-R. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6645–6649, May 2013.

[9] K. Han and DL. Wang. Neural network based pitch tracking in very noisy speech. *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, 22(12):2158–2168, October 2014.

[10] C.-L. Hsu and J.-S. R. Jang. On the improvement of singing voice separation for monaural recordings using the mir-1k dataset. *IEEE Trans. on Audio, Speech, and Language Processing*, 18(2):310–319, 2010.

[11] S. Jo, S. Joo, and C. D. Yoo. Melody pitch estimation based on range estimation and candidate extraction using harmonic structure model. In *Proc. of INTERSPEECH*, pages 2902–2905, 2010.

[12] B. S. Lee and D. P. W. Ellis. Noise robust pitch tracking by subband autocorrelation classification. In *Proc. of INTERSPEECH*, 2012.

[13] S. Leglaive, R. Hennequin, and R. Badeau. Singing voice detection with deep recurrent neural networks. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 121–125, April 2015.

[14] M. Mauch and S. Ewert. The audio degradation toolbox and its application to robustness evaluation. In *Proc. of the 14th Int. Society for Music Information Retrieval (ISMIR) Conference*, November 2013.

[15] A. Mesaros and T. Virtanen. Automatic alignment of music audio and lyrics. In *Proc. of 11th Int. Conf. on Digital Audio Effects (DAFx)*, September 2008.

[16] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis. mir_eval: a transparent implementation of common MIR metrics. In *Proc. of the 15th Int. Society for Music Information Retrieval (ISMIR) Conference*, October 2014.

[17] M. Ryynänen and A. P. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, 2008.

[18] M. Ryynänen, T. Virtanen, J. Paulus, and A. Klapuri. Accompaniment separation and karaoke application based on automatic melody transcription. In *Proc. of the IEEE Int. Conf. on Multimedia and Expo*, pages 1417–1420, April 2008.

[19] J. Salamon, E. Gómez, D. P. W. Ellis, and G. Richard. Melody extraction from polyphonic music signals. Approaches, applications, and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134, March 2014.

[20] J. Salamon, B. Rocha, and E. Gómez. Musical genre classification using melody features extracted from polyphonic music. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 81–84, March 2012.

[21] J. Salamon, J. Serrà, and E. Gómez. Tonal representations for music retrieval: From version identification to query-by-humming. *Int. Jour. of Multimedia Information Retrieval, special issue on Hybrid Music Information Retrieval*, 2(1):45–58, 2013.

[22] H. Tachibana, T. Ono, N. Ono, and S. Sagayama. Melody line estimation in homophonic music audio signals based on temporal-variability of melodic source. In *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 425–428, March 2010.

[23] C. H. Wong, W. M. Szeto, and K. H. Wong. Automatic lyrics alignment for Cantonese popular music. *Multimedia Systems*, 4-5(12):307–323, 2007.

SPARSE CODING BASED MUSIC GENRE CLASSIFICATION USING SPECTRO-TEMPORAL MODULATIONS

Kai-Chun Hsu **Chih-Shan Lin** **Tai-Shih Chi**
Department of Electrical and Computer Engineering, National Chiao Tung University, Taiwan
{kch610596, g104972}@gmail.com, tschi@mail.nctu.edu.tw

ABSTRACT

Spectro-temporal modulations (STMs) of the sound convey timbre and rhythm information so that they are intuitively useful for automatic music genre classification. The STMs are usually extracted from a time-frequency representation of the acoustic signal. In this paper, we investigate the efficacy of two kinds of STM features, the Gabor features and the rate-scale (RS) features, selectively extracted from various time-frequency representations, including the short-time Fourier transform (STFT) spectrogram, the constant-Q transform (CQT) spectrogram and the auditory (AUD) spectrogram, in recognizing the music genre. In our system, the dictionary learning and sparse coding techniques are adopted for training the support vector machine (SVM) classifier. Both spectral-type features and modulation-type features are used to test the system. Experiment results show that the RS features extracted from the log. magnituded CQT spectrogram produce the highest recognition rate in classifying the music genre.

1. INTRODUCTION

For a classification task, selected features and the classifier are critical to the performance of the system. Since the last decade, lots of researchers have proposed music genre classification systems using designed features or classifiers. For instance, the mel-frequency cepstral coefficients (MFCCs), the pitch histogram and the beat histogram were used in [1] as effective features to describe characteristics of timbre, pitch and rhythm of music. The SVM was used in a multi-layer fashion for genre classification [2]. Later on, parameters of autoregressive models of spectral raw features were used for classification by including the temporal variations of the raw features [3]-[5]. In addition to SVM, the adaptive boosting algorithm was used to train the classifier [6]. The non-negative tensor factorization (NTF) was also considered to reduce the dimensionality in a sparse representation classifier (SRC) [7][8]. Another approach was to extract features from the separated cleaner signal [9] by first applying the harmonic-percussion signal separation (HPSS) algorithm [10] to the music clip. The Gaussian supervector, which has been successfully used in speaker identification, was also investigated in music genre classification [11]. A supervised dictionary learning process was proposed for genre classification by using codebooks generated from existing coding techniques [12]. All these methods were operated on audio signals only. In addition, one can also combine features from other sources such as MIDI or lyrics [13]-[17].

In recent years, sparse coding technique has been applied to music genre classification. Most sparse coding based automatic music genre classification systems transform the music signal into frame-level raw features, and then encode the frame-level features into frame-level sparse codes. Since the encoding only considers information in one frame, temporal pooling technique has been included in this kind of system. For instance, the combinations of statistical moments of a multiple frame representation were used for temporal pooling on raw features [18]. Histogram and pyramid based bag-of-segments schemes were also considered for temporal pooling on encoding [19].

In addition to considering temporal pooling on spectral features, amplitude modulations shown on the short-time Fourier transform (STFT) spectrogram, which depict the spectral patterns varying across time, were extracted using a set of 2-D Gabor filters for genre classification [20]. It has been shown that joint spectro-temporal modulations (STMs) on the auditory (AUD) spectrogram are helpful for music signal categorization, hence helpful for music separation [21]. No doubt that STMs carry critical information and are suitable for genre classification. However, does the information conveyed by the STMs provide more benefit than the spectral features? If so, what kind of spectrogram provides the most informative STMs for genre classification? Is it the STFT spectrogram or the hearing-morphic spectrogram such as the constant-Q transform (CQT) spectrogram or the AUD spectrogram? This paper is trying to answer these questions. Here, we built a sparse coding based genre classification system for evaluations.

The rest of this paper is organized as follows. A brief introduction of the tested spectrograms and STM features are presented in Section 2. Section 3 describes the sparse coding and dictionary learning. Section 4 describes the genre classification method and shows evaluation results. Lastly, Section 5 draws the conclusion.

2. FEATURE EXTRACTION

In this section, we introduce the various features used in this paper. Two types of raw features are considered: the frame-level features extracted from STFT, CQT and AUD spectrograms; and their corresponding STM fea-


© Kai-Chun Hsu, Chih-Shan Lin, Tai-Shih Chi.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Kai-Chun Hsu, Chih-Shan Lin, Tai-Shih Chi. "Sparse Coding Based Music Genre Classification using Spectro-Temporal Modulations", 17th International Society for Music Information Retrieval Conference, 2016.

tures. For the STM features, we apply Gabor filters to STFT spectrogram [20] and rate-scale (RS) filters to the hearing-morphic CQT and AUD spectrograms. Features mentioned in this section are considered as raw features in the dictionary for the sparse coding system.

2.1 Frame-based Features

2.1.1 STFT Spectrogram

The STFT spectrogram is the most conventional time-frequency representation of audio signals. In this paper, we computed 1024-point FFT for each frame and adjacent frames are with 50% overlap. This computation resulted in a 513-dimensional magnitude spectrum which served as a feature vector.

2.1.2 CQT Spectrogram

The constant-Q transform (CQT) produces another kind of time-frequency audio representation with logarithmic frequency scale and different temporal/spectral resolutions at different frequency bands. The CQT spectrogram is considered closely suited to human perception of sound.

In this paper, we set 8 octaves for the frequency range with the frequency resolution of 64 bins per octave, resulting in 512-dimensional feature vectors. For implementation, we used the Constant-Q Transform Toolbox [22][23] which implements the computationally-efficient CQT transform based on FFT [24].

2.1.3 Auditory (AUD) Spectrogram

The AUD spectrogram is produced by the cochlear module of the auditory model [25]. An input sound is first filtered by a bank of 128 overlapping asymmetric bandpass filters which mimic the frequency selectivity of the cochlea. The center frequencies of the cochlear filters are evenly distributed along a logarithmic frequency axis, over 5.3 octaves (180Hz ~ 7246Hz) with the frequency resolution of 24 filters per octave. The output of each filter is fed into a non-linear compression stage, which models the saturation of inner hair cells while transducing the vibrations of the basilar membrane into intracellular potentials. Next, a simple lateral inhibitory network (LIN) is implemented by a first-order differentiator across filters to account for the masking effect between adjacent filters. A half-wave rectifier combined with a lowpass filter serves as an envelope extractor after the LIN. At the end, the cochlear module produces 128-dimensional feature vectors.

The block diagram of the cochlear module is shown in Figure 1. Outputs at different stages can be formulized as follows:

$$y_1(f,t) = s(t) *_t h(t;f) \quad (1)$$

$$y_2(f,t) = g(\partial_t y_1(f,t)) *_t l(t) \quad (2)$$

$$y_3(f,t) = \max(\partial_f y_2(f,t), 0) \quad (3)$$

$$y_4(f,t) = y_3(f,t) *_t \mu(t;\tau) \quad (4)$$

where $s(t)$ is the input audio signal, $h(t;f)$ is the impulse response of the cochlear filter with the center frequency f, $*_t$ depicts convolution in time, $g(.)$ is a sigmoid function, $l(t)$ a lowpass filter, ∂_t, ∂_f are partial derivative along t, f axes, $\mu(t;\tau) = e^{-t/\tau}u(t)$ is the integration window with the time constant τ, and $u(t)$ is the unit step function. Detailed discussions about this module can be assessed in [26].

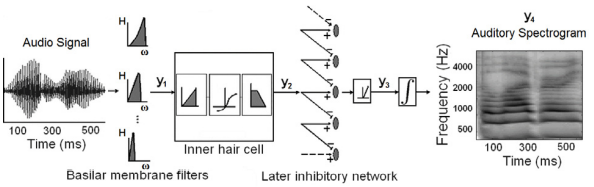


Figure 1. Block diagrams for deriving an AUD spectrogram.

2.2 Modulation Features

2.2.1 Gabor Features

Gabor features are the spectro-temporal "visual features" extracted from a STFT spectrogram as proposed in [20]. To obtain these features, an input audio signal was first transformed into a STFT spectrogram. Then, the STFT spectrogram was divided into 7 sub-spectrograms according to the following 7 subbands: 0Hz ~ 200Hz, 200Hz ~ 400Hz, 400Hz ~ 800Hz, 800Hz ~ 1600Hz, 1600Hz ~ 3200Hz, 3200Hz ~ 8000Hz, and 8000Hz ~ half sampling frequency. Third, each sub-spectrogram was filtered by a set of 42 pre-defined 2-D Gabor filters:

$$\psi(x',y') = \exp\left(-\left(\frac{x'^2+y'^2}{2\sigma^2}\right)\right)\exp\left(\frac{j2\pi x'}{\lambda}\right) \quad (5)$$

$$x' = x\cos(\theta) + y\sin(\theta) \quad (6)$$

$$y' = -x\sin(\theta) + y\cos(\theta) \quad (7)$$

where x and y represent the time and frequency axes of the STFT sub-spectrogram, $\theta \in \{0°, 30°, \ldots, 150°\}$ indicates the orientation of the Gabor filter, $\lambda \in \{2.5, 5, \ldots, 17.5\}$ denotes the thickness of the Gabor filter, and $\sigma = 0.5\lambda$ for the standard deviation of the Gaussian function. This process transformed the STFT spectrogram into 294 modulation sub-spectrograms. The energy contour of each modulation sub-spectrogram was obtained by averaging the modulation sub-spectrogram along the frequency axis. At this stage, the STFT spectrogram was transformed into 294 modulation energy contours in the time domain. Finally, the mean and standard deviation of these contours were concatenated to form the "visual features", referred to as the Gabor features in this paper.

Figure 2 demonstrates the meaning of the Gabor features. The upper panel shows a segment of a sample STFT sub-spectrogram, while the bottom panel shows two energy contours derived from outputs of the two Gabor filters ($\lambda = 7.5,\ \theta = 0°$ and $\lambda = 7.5,\ \theta = 90°$). We can observe that strong responses of the contours result from strong vertical and horizontal patterns in the spectrogram.

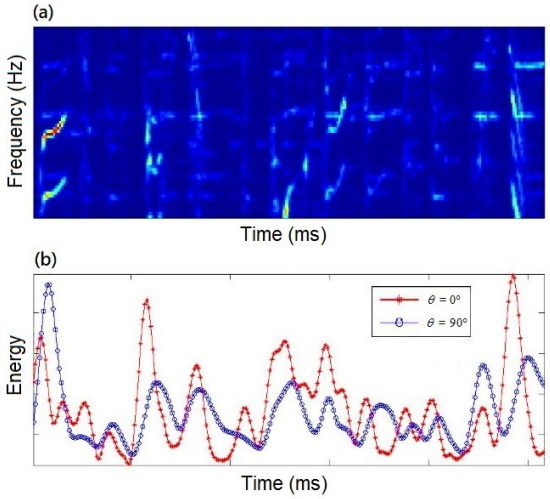


Figure 2. (a) A sample STFT sub-spectrogram. (b) The energy contours derived from outputs of two Gabor filters ($\lambda = 7.5,\ \theta = 0°$ and $\lambda = 7.5,\ \theta = 90°$).

2.2.2 Rate-Scale (RS) Features

RS feature is another kind of modulation feature extracted by the cortical module [25]. The cortical module, which is inspired by neural recordings of the auditory cortex (A1), models the spectro-temporal selectivity of neurons in A1 [26].

Specifically, in the auditory model, the AUD spectrogram is further analyzed by neurons in A1. From functional point of view, the cortical neurons are modeled as a bank of two-dimensional filters with different spectro-temporal selectivity. These two-dimensional filters can be characterized by spectro-temporal modulation parameters, rate and scale. The rate parameter characterizes the velocity of the modulation varying along the temporal axis on the AUD spectrogram and the scale parameter characterizes the density of the modulation distributed along the logarithmic frequency axis on the AUD spectrogram. The filtering process can be formulized as follows:

$$r(f,t,\omega,\Omega) = y_4(f,t) *_{ft} STIR(f,t;\omega,\Omega) \qquad (8)$$

where r denotes the 4-dimensional output of the cortical module, y_4 is the AUD spectrogram, $*_{ft}$ denotes the two-dimensional convolution along temporal and logarithmic frequency axes, $STIR$ is the impulse response of the two-dimensional modulation filter, ω and Ω denote the rate and the scale parameter respectively. Figure 3 demonstrates examples of rate-scale features of three time-frequency (T-F) units in the AUD spectrogram. The top panel shows a sample AUD spectrogram and the bottom three panels show the rate-scale plots, which record the local amplitude resolved by each of the rate-scale modulation filters, of the three T-F units indicated by the arrows. The rate-scale plot reflects local modulation energy distribution and the sweeping direction of the modulation (positive/negative rate representing the downward/upward directivity) of a particular T-F unit in the AUD spectrogram. Detailed explanations about the information encoded by the rate-scale plot can be assessed in [21].

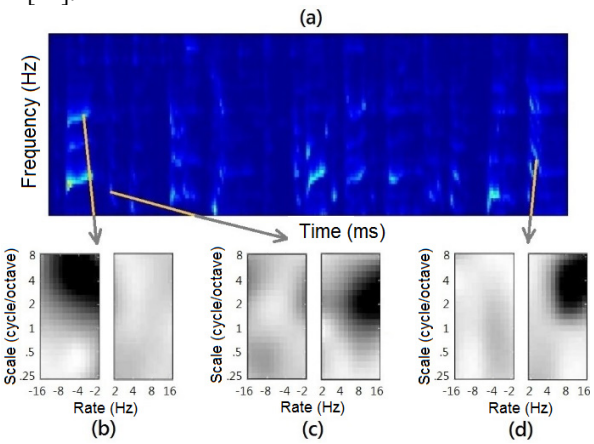


Figure 3. (a) A sample AUD spectrogram. (b)(c)(d) Rate-scale plots of the T-F units indicated by the arrows in (a).

In this paper, the local amplitude of the cortical output r is averaged in each subband and concatenated,

$$\bar{r}_i = \frac{1}{N_i}\sum_{j=1}^{Ni} r(f_j,t,\omega,\Omega) \qquad (9)$$

$$\bar{r} = [\bar{r}_1, \bar{r}_2, \cdots \bar{r}_O] \qquad (10)$$

where $o \in \{1,2,3,\ldots,O\}$ is the index of the subband, Ni is the number of bins in the i-th subband, O is the total number of subbands and $\bar{r}$ is our final RS feature. We extract RS features from the AUD spectrogram and the CQT spectrogram. For the cases of AUD spectrogram, the parameters were selected as $\omega \in \pm\{2,4,8,16\}$, $\Omega \in \{0.25,0.5,1,2,4,8\}$ and O=6(180Hz ~ 200Hz, 200Hz ~ 400Hz, 400Hz ~ 800Hz, 800Hz ~ 1600Hz, 1600Hz ~ 3200Hz, 3200Hz ~ 7246Hz), resulting in 288-dimensional feature vectors. For the cases of CQT spectrogram, the parameter were selected as $\omega \in \pm\{2,4,8\}$, $\Omega \in \{0.25,0.5,1,2,4,8,16\}$ and O=7(0Hz ~ 200Hz, 200Hz ~ 400Hz, 400Hz ~ 800Hz, 800Hz ~ 1600Hz, 1600Hz ~ 3200Hz, 3200Hz ~ 8000Hz, 8000Hz ~ half-sampling frequency), resulting in 294-dimensional feature vectors. These parameters were selected mainly to have comparable feature dimensions with the restriction posed by the 5.3-octave frequency coverage of the AUD spectrogram.

3. SPARSE CODING AND DICTIONARY LEARNING

Generally speaking, the sparse coding technique decomposes the original signal into a combination of a few codewords in a given codebook (or dictionary). The objective function of sparse coding can be formulated as:

$$\alpha^* = \arg\min_{\alpha} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda\|\alpha\|_1 \qquad (11)$$

where $x \in \mathbb{R}^d$ is the input signal (the feature vector in our case), $\alpha \in \mathbb{R}^k$ is the sparse code of x, $D \in \mathbb{R}^{d \times k}$ is a given dictionary and λ is a parameter which controls the sparsity of α. The d is the dimension of the feature vector and the k is the codebook size. Equation (11) is usually referred to as the Lasso problem and can be solved by the LARS-lasso algorithm [27].

Furthermore, the objective function of dictionary learning can be formulated as:

$$D^* = \arg\min_{D,\alpha} \frac{1}{n} \sum_{i=1}^{n} \left(\frac{1}{2} \|x_i - D\alpha_i\|_2^2 + \lambda \|\alpha_i\|_1 \right) \qquad (12)$$

where n is the total number of data to train the dictionary. Equation (12) represents a joint optimization problem in α and D. In this paper, the online dictionary learning (ODL) algorithm [28][29] was used to train the dictionary and the SPArse Modeling Software (SPAMS) [30] was used for implementation.

4. EXPERIMENTS

In this section, we describe the settings of the experiments and the classification results using various kinds of features.

4.1 Dataset

4.1.1 GTZAN Dataset

This public dataset is frequently used in literature for evaluation of automatic music genre classification. The dataset is composed of 100 30-second music clips in each of the ten genres (blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock). Each clip is sampled at 22050 Hz.

4.1.2 H-L Dataset

This dataset was collected by ourselves and used in this work for learning the dictionary of music. The dataset is composed of 100 30-second clips in each of the 21 genres (a cappella, A-pop, blues, bossa nova, classical, C-pop, electropop, funk, hip-hop, jazz, J-pop, latin, metal, musical, new age, opera, R&B, reggae, rock, romantical, and soul). All the clips in this dataset are different from those in the GTZAN dataset. Each clip is sampled at 44100 Hz.

4.2 System Overview

In our classification system, an input music clip is first transformed into the frame-level feature vectors. The feature vectors are then normalized to unit l_2-norm vectors. Second, the normalized feature vectors are encoded into frame-level sparse codes. Next, we summarize the frame-level sparse codes over the entire clip to obtain the song-level feature w. Finally, w is power normalized using Equation (13) to train/test the classifier. The block diagram of the classification system is shown in Figure 4.

$$w^* = sign(w)|w|^a \qquad (13)$$

In all of the experiments, we set the codebook size to 1024, the regularization parameter λ to $1/\sqrt{d}$, and the power normalization parameter a in Equation (13) to 0.5. The linear-SVM implemented in LIBSVM [31] was used as the classifier. Evaluation results were obtained by averaging results from 100 ten-fold cross-validation.

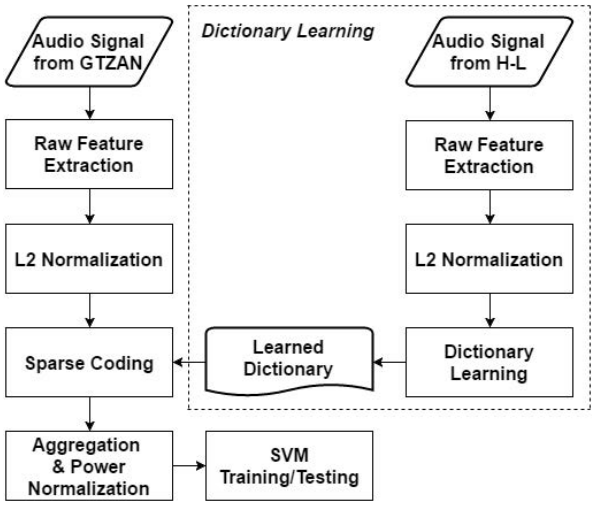


Figure 4. The block diagram of the sparse coding based classification system. The H-L dataset was mainly used to generate the dictionary of music.

4.3 Experiment Results

Experiment results are shown in this section. For simplicity, the name of the classification system is referred to as the name of the used raw features (e.g., the sparse coding based automatic genre classification system using STFT features is referred to as the STFT system).

4.3.1 Spectrogram Features versus Modulation Features

Recognition rates using the spectrogram features and the corresponding modulation features (STFT/Gabor, AUD/RS, CQT/RS) are shown in Figure 5. We can see that corresponding modulation features of the STFT spectrogram have a negative impact to system performance (75.8% to 73.8%) while they provide significant benefit to AUD spectrogram (71.0 to 79.7%) and CQT spectrogram (77.0% to 84.7%).

The main difference among the three spectrograms is the frequency scale, linear scale in STFT but logarithmic scale in AUD and CQT spectrograms. We postulate that modulation features extracted from the logarithmic frequency spectrogram are beneficial to genre classification. For validation, Gabor features were extracted from all three spectrograms and tested for system performance. Figure 6 shows the results of three Gabor systems (STFT/Gabor, AUD/Gabor, CQT/Gabor). Clearly, comparing with spectrogram features, Gabor features demonstrate a positive effect on the genre classification rate when extracted from the AUD and CQT spectrograms but a negative effect when extracted from the STFT spectrogram.

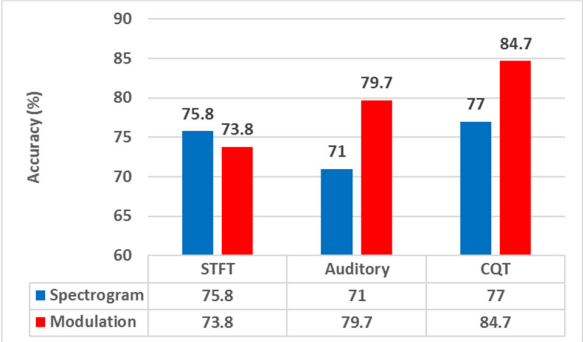


Figure 5. The recognition rates using the spectrogram features and corresponding modulation features (STFT/Gabor, AUD/RS, CQT/RS).

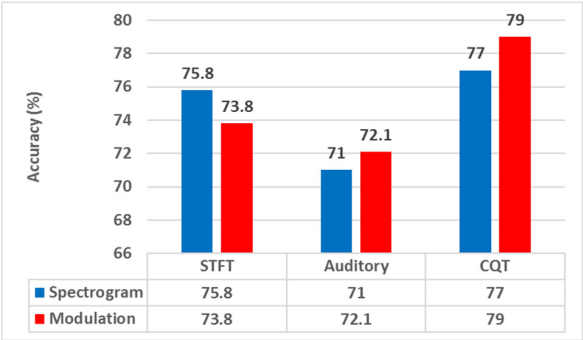


Figure 6. The recognition rates using the spectrogram features and Gabor modulation features (STFT/Gabor, AUD/Gabor, CQT/Gabor).

4.3.2 AUD Spectrogram versus CQT Spectrogram

Figure 5 and 6 show both Gabor and RS modulation features extracted from the CQT spectrogram perform better than those extracted from the AUD spectrogram. The main differences between CQT and AUD spectrograms are the filter shape, the frequency resolution (64 bins per octave on CQT and 24 bins per octave on AUD), and the covered frequency range (40Hz ~ 10700Hz on CQT and 180Hz ~ 7246Hz on AUD).

To investigate the effects from the frequency resolution and the frequency range, we tested RS modulation features extracted from CQT spectrograms with different settings listed in Table 1. The recognition rates are shown in Figure 7. We can observe that higher frequency resolution (64 bins/octave versus 24 bins/octave) does not necessarily produce higher recognition rate. Finding the optimal frequency resolution for recognition rate, however, is beyond the scope of this work. On the other hand, wider frequency coverage is more beneficial to system performance. In Exp.4, the CQT spectrogram was computed using the same frequency resolution and frequency coverage as the AUD spectrogram yet its RS features outperforms the RS features of AUD (82.4% versus 79.7% shown in Figure 5). It is probably because the CQT spectrogram possesses a higher Q value than the AUD spectrogram. A higher Q value generates a more sharpened spectrogram hence producing better performance.

Exp.	frequency range	bins/octave	ω	Ω
1	40Hz ~ 10700Hz	64	2~16	0.25~8
2	40Hz ~ 10700Hz	24	2~16	0.25~8
3	180Hz ~ 7246Hz	64	2~16	0.25~8
4	180Hz ~ 7246Hz	24	2~16	0.25~8

Table 1. Different frequency settings for generating CQT/RS modulation features

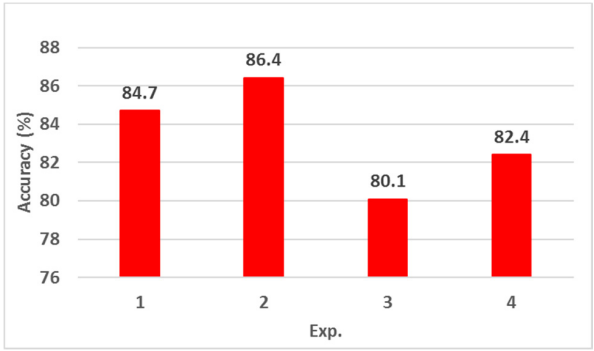


Figure 7. The recognition rates using CQT/RS features with different frequency range and frequency resolutions as listed in Table 1.

4.3.3 Gabor Features versus RS Features

From Figure 5 and 6, we can observe that the RS feature set performs better than the Gabor feature set on both of CQT and AUD spectrograms. RS features and Gabor features are produced using different sets of 2-D modulation filters. The parameters of the Gabor filter, θ and λ, and the parameters of the rate-scale filter, ω and Ω, affect the shape of the 2-D filter by changing its center frequency and bandwidth.

To demonstrate the effect of using different modulation filters, we tested RS features extracted from CQT spectrogram using a different set of ω and Ω. Experiment setting is listed as Exp.2 in Table 2 and the results are shown in Figure 8, where Exp.1 and Exp.3 are the RS/Gabor features using the original set of 2-D filters, respectively. We can observe that selecting different 2-D modulation filters significantly affect system performance. Therefore, selecting an appropriate set of 2-D filters for modulation feature extraction is important to system performance.

Exp.	2-D filter	ω	Ω	θ	λ
1	RS	2~8	0.25~16		
2	RS	0.25~16	2~8		
3	Gabor			0°, 30°, ..., 150°	2.5, 5, ..., 17.5

Table 2. Different modulation filters used to extract modulation features from the CQT spectrogram

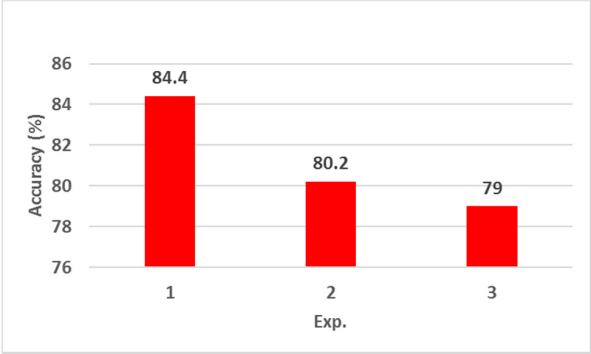


Figure 8. The recognition rates using different sets of 2-D modulation filters on the CQT spectrogram. Parameters of the modulation filters are listed in Table 2.

4.3.4 Linear Magnitude versus Log. Magnitude

It has been shown that logarithmic magnitude STFT produces better features than the linear magnitude STFT for genre classification [12]. In this sub-section, we demonstrate the effect of using log. magnitude on spectrograms. In addition to using spectrogram features, the system performance using modulation features extracted from the log. magnitude spectrograms were also examined.

Experiment results of using log. magnitude spectrograms versus using linear magnitude spectrograms are shown in Figure 9. Results of using their corresponding modulation features (Gabor from STFT, RS from AUD, and RS from CQT) are shown in Figure 10. We can see that both spectrogram features and modulation features extracted from log. magnitude spectrograms perform better than those extracted from linear magnitude spectrograms.

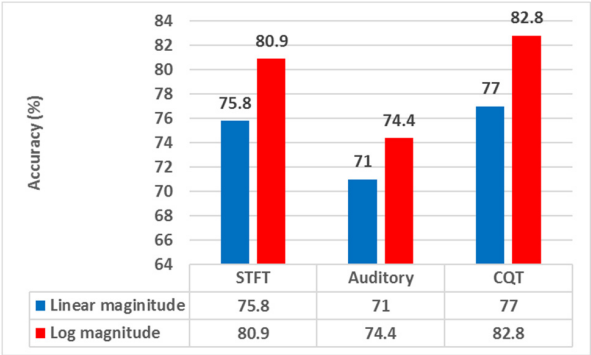


Figure 9. Recognition rates of using spectral profiles extracted from log. magnitude spectrograms and from linear magnitude spectrograms.

5. CONCLUSIONS

In this paper, we investigate the efficacy of features derived from joint spectro-temporal modulations, which intrinsically convey timbre and rhythm information of the sound, on music genre classification using a sparse coding based classification system. We extract two kinds of STM features, the Gabor and RS features, from three kinds of spectrograms, STFT, auditory, and CQT spectrograms, of the music signal and conduct several comparative experiments. The results show that modulation features do outperform spectral profiles in genre classification. In addition, several conclusions can be drawn from our results: 1) modulation features extracted from the logarithmic frequency scaled spectrogram perform better than those extracted from the linear frequency scaled spectrogram; 2) the spectrogram with wider frequency coverage produces more effective modulation features; 3) the selection of modulation filters could be task-dependent; 4) modulation features extracted from log. magnitude spectrograms produce higher genre recognition rates than those extracted from linear magnitude spectrograms.

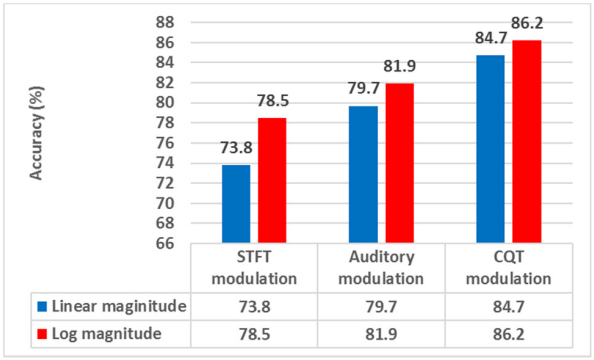


Figure 10. Recognition rates of using modulation features extracted from log. magnitude spectrograms and from linear magnitude spectrograms.

In this paper, the highest genre recognition rate on GTZAN dataset using modulation features is 86.2%, which is obtained by using RS features extracted from the log. magnitude CQT spectrogram. From experiment results shown in Section 4, we can assume the performance can probably be better by fine-tuning the parameters of the classification system, including the rate, scale parameters of the modulation filters and the frequency resolution of the CQT spectrogram.

6. ACKNOWLEDGEMENTS

This research is supported by the National Science Council, Taiwan under Grant No NSC 102-2220-E-009-049.

7. REFERENCES

[1] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[2] C. Xu, N. C. Maddage, X. Shao, F. Cao, and Q. Tian, "Musical genre classification using support vector machines," *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process.*, no. 5, pp. 429–432, 2003.

[3] A. Meng, P. Ahrendt, and J. Larsen, "Improving music genre classification by short-time feature intergration," *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process.*, no. 5, pp. 497–500, 2005.

[4] A. Meng and J. Shawe-Taylor, "An investigation of feature models for music genre classification using the support vector classifier," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp.604–609, 2005.

[5] A. Meng, P. Ahrendt, J. Larsen, and L. K. Hansen, "Temporal feature integration for music genre classification," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 15, no. 5, pp. 1654–1664, 2007.

[6] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. K´egl, "Aggregate features and adaboost for music classification," *Machine Learning*, vol. 65, no. 2-3, pp. 473–484, 2006.

[7] Y. Panagakis, C. Kotropoulos, and G. R. Arce, "Music genre classification using locality preserving non-negative tensor factorization and sparse representations," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 249–254, 2009.

[8] Y. Panagakis and C. Kotropoulos, "Music genre classification via topology preserving non-negative tensor factorization and sparse representations," *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process.*, pp. 249–252, 2010.

[9] H. Rump, S. Miyabe, E. Tsunoo, N. Ono, and S. Sagama, "Autoregressive mfcc models for genre classification improved by harmonic-percussion separation," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 87–92, 2010.

[10] N. Ono, K. Miyamoto, H. Kameoka, and S. Sagayama, "A real-time equalizer of harmonic and percussive componets in music signals," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 139–144, 2008.

[11] Cao, Chuan, and Ming Li. "Thinkit's submissions for MIREX2009 audio music classification and similarity tasks." *Music Information Retrieval Evaluation eXchange (MIREX)* (2009).

[12] C.-C. M. Yeh and Y.-H. Yang, "Supervised dictionary learning for music genre classification," *Proc. ACM Int. Conf. on Multimedia Retrieval*, no. 55, 2012.

[13] A. Ruppin and H. Yeshurun, "Midi genre classification by invariant features," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 397–399, 2006.

[14] T. Lidy, A. Rauber, A. Pertusa, and J. M. Inesta, "Improving genre classification by combination of audio and symbolic descriptors using a transcription system," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 61–66, 2007.

[15] R. Mayer, R. Neumayer, and A. Rauber, "Rhyme and style features for musical genre categorization by song lyrics," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 337–342, 2008.

[16] C. McKay, J. A. Burgoyne, J. Hockman, J. B. L. Smith, G. Vigliensoni, and I. Fujinaga, "Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 213–218, 2010.

[17] R. Mayer and A. Rauber, "Music genre classification by ensembles of audio and lyrics features," *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 675–680, 2011.

[18] C.-C. M. Yeh and Y.-H. Yang, "Towards a more efficient sparse coding based audio-word feature extraction system," *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conf.*, pp. 1–7, 2013.

[19] C.-C. M. Yeh, L. Su, and Y.-H. Yang, "Dual-layer bag-of-frames model for music genre classification," *Proc. IEEE Int. Conf. on Acoust, Speech and Signal Process*, pp. 246–250, 2013.

[20] M.-J. Wu, Z.-S. Chen, and J.-S. R. Jang, "Combining visual and acoustic features for music genre classification," *Proc. IEEE Int. Conf. on Machine Learning and Applications and Workshops*, pp. 124–129, 2011.

[21] Yen, Frederick, Yin-Jyun Luo, and Tai-Shih Chi. "Singing Voice Separation Using Spectro-Temporal Modulation Features." *Proc. of the Int. Soc. for Music Inform. Retrieval Conf.*, pp. 617–622, 2014.

[22] [Online]https://code.soundsoftware.ac.uk/projects/constant-q-toolbox.

[23] Schörkhuber, Christian, and Anssi Klapuri. "Constant-Q transform toolbox for music processing." *Proc. of Sound and Music Computing Conference,* pp. 3-64, 2010.

[24] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant q transform," *The Journal of the Acoustical Society of America*, vol. 92, no. 5, pp. 2698–2701, 1992.

[25] [Online]http://www.isr.umd.edu/Labs/NSL/Downloads.html

[26] T. Chi, P. Ru, and S. A. Shamma, "Multiresolution spectrotemporal analysis of complex sounds," *The Journal of the Acoustical Society of America*, vol. 118, no. 2, pp. 887–906, 2005.

[27] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[28] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," *Proc. of Int. Conf. on Machine Learning*, pp. 689–696, 2009.

[29] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, pp. 19–60, 2010.

[30] [Online]http://spams-devel.gforge.inria.fr/downloads.html

[31] Chang, Chih-Chung, and Chih-Jen Lin. "LIBSVM: a library for support vector machines." *ACM Transactions on Intelligent Systems and Technology,* vol. 2, no.3, Article 27, 2011.

TIME-DELAYED MELODY SURFACES FOR RĀGA RECOGNITION

Sankalp Gulati[1] **Joan Serrà**[2] **Kaustuv K Ganguli**[3] **Sertan Şentürk**[1] **Xavier Serra**[1]
[1] Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
[2] Telefonica Research, Barcelona, Spain
[3] Dept. of Electrical Engg., Indian Institute of Technology Bombay, Mumbai, India
sankalp.gulati@upf.edu

ABSTRACT

Rāga is the melodic framework of Indian art music. It is a core concept used in composition, performance, organization, and pedagogy. Automatic rāga recognition is thus a fundamental information retrieval task in Indian art music. In this paper, we propose the time-delayed melody surface (TDMS), a novel feature based on delay coordinates that captures the melodic outline of a rāga. A TDMS describes both the tonal and the temporal characteristics of a melody, using only an estimation of the predominant pitch. Considering a simple k-nearest neighbor classifier, TDMSs outperform the state-of-the-art for rāga recognition by a large margin. We obtain 98% accuracy on a Hindustani music dataset of 300 recordings and 30 rāgas, and 87% accuracy on a Carnatic music dataset of 480 recordings and 40 rāgas. TDMSs are simple to implement, fast to compute, and have a musically meaningful interpretation. Since the concepts and formulation behind the TDMS are generic and widely applicable, we envision its usage in other music traditions beyond Indian art music.

1. INTRODUCTION

Melodies in Hindustani and Carnatic music, two art music traditions of the Indian subcontinent, are constructed within the framework of rāga [3, 29]. The rāga acts as a grammar within the boundaries of which an artist composes a music piece or improvises during a performance. A rāga is characterized by various melodic attributes at different time scales such as a set of svaras (roughly speaking, notes), specific intonation of these svaras, ārōhana-avrōhana (the ascending and descending sequences of svaras), and by a set of characteristic melodic phrases or motifs (also referred to as 'catch phrases'). In addition to these melodic aspects, one of the most important characteristics of a rāga is its calan [23] (literally meaning movement or gait). The calan defines the melodic outline of a rāga, that is, how a melodic transition is made from one svara to another, the precise intonation to be followed during the transition, and the proportion of time spent on each svara. It can also be thought of as an abstraction of the characteristic melodic phrases mentioned above.

Rāga is a core musical concept used in the composition, performance, organization, and pedagogy of Indian art music (IAM). Numerous compositions in Indian folk and film music are also based on rāgas [9]. Despite its significance in IAM, there exists a large volume of audio content whose rāga is incorrectly labeled or, simply, unlabeled. This is partially because the vast majority of the tools and technologies that interact with the recordings' metadata fall short of fulfilling the specific needs of the Indian music tradition [26]. A computational approach to automatic rāga recognition can enable rāga-based music retrieval from large audio collections, semantically-meaningful music discovery, musicologically-informed navigation, as well as several applications around music pedagogy.

Rāga recognition is one of the most researched topics within music information retrieval (MIR) of IAM. As a consequence, there exist a considerable amount of approaches utilizing different characteristic aspects of rāgas. Many of such approaches use features derived from the pitch or pitch-class distribution (PCD) [2, 4, 5, 16]. This way, they capture the overall usage of the tonal material in an audio recording. In general, PCD-based approaches are robust to pitch octave errors, which is one of the most frequent errors in the estimation of predominant melody from polyphonic music signals. Currently, the PCD-based approach represents the state-of-the-art in rāga recognition. One of these approaches proposed by Chordia et al. [2] has shown promising results with an accuracy of 91.5% on a sizable dataset comprising 23 rāgas and close to 550 excerpts of 120 s duration, extracted from 121 audio recordings (note that the authors use monophonic recordings made under laboratory conditions).

One of the major shortcomings of PCD-based approaches is that they completely disregard the temporal aspects of the melody, which are essential to rāga characterization [23]. Temporal aspects are even more relevant in distinguishing phrase-based rāgas [17], as their aesthetics and identity is largely defined by the usage of meandering melodic movements, called gamakas. Several approaches address this shortcoming by modeling the temporal aspects of a melody in a variety of ways [18, 21, 27]. Such approaches typically use melodic progression templates [27], n-gram distributions [18], or hidden Markov models [21]

© Sankalp Gulati, Joan Serrà, Kaustuv K Ganguli, Sertan Şentürk and Xavier Serra. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Sankalp Gulati, Joan Serrà, Kaustuv K Ganguli, Sertan Şentürk and Xavier Serra. "Time-delayed melody surfaces for Rāga recognition", 17th International Society for Music Information Retrieval Conference, 2016.

to capture the sequential information in the melody. With that, they primarily utilize the ārōhana-avrōhana pattern of a rāga. In addition, most of them either transcribe the predominant melody in terms of a discrete svara sequence, or use only a single symbol/state per svara. Thus, they discard the characteristic melodic transitions between svaras, which are a representative and distinguishing aspect of a rāga [23]. Furthermore, they often rely on an accurate transcription of the melody, which is still a challenging and an ill-defined task given the nature of IAM [22, 24].

There are only a few approaches to rāga recognition that consider the continuous melody contour and exploit its raw melodic patterns [6, 11]. Their aim is to create dictionaries of characteristic melodic phrases and to exploit them in the recognition phase, as melodic phrases are prominent cues for the identification of a rāga [23]. Such phrases capture both the svara sequence and the transition characteristics within the elements of the sequence. However, the automatic extraction of characteristic melodic phrases is a challenging task. Some approaches show promising results [11], but they are still far from being perfect. In addition, the melodic phrases used by these approaches are typically very short and, therefore, more global melody characteristics are not fully considered.

In this paper, we propose a novel feature for rāga recognition, the time-delayed melody surface (TDMS). It is inspired by the concept of delay coordinates [28], as routinely employed in nonlinear time series analysis [15]. A TDMS captures several melodic aspects that are useful in characterizing and distinguishing rāgas and, at the same time, alleviates many of the critical shortcomings found in existing methods. The main strengths of a TDMS are:

- It is a compact representation that describes both the tonal and the temporal characteristics of a melody
- It simultaneously captures the melodic characteristics at different time-scales, the overall usage of the pitch-classes in the entire recording, and the short-time temporal relation between individual pitches.
- It is robust to pitch octave errors.
- It does not require the transcription of the melody nor a discrete representation of it.
- It is easy to implement, fast to compute, and has a musically-meaningful interpretation.
- As it will be shown, it obtains unprecedented accuracies in the raga recognition task, outperforming the state-of-the-art by a large margin, without the use of any elaborated classification schema.

In our experiments, we use TDMSs together with a k-nearest neighbor classifier and a set of well known distance measures. The reported results are obtained on two scalable, diverse, and representative data sets of Carnatic and Hindustani music, one of which is originally introduced in this study and made publicly available. To the best of our knowledge, these are the largest publicly available data sets for rāga recognition in terms of the number of recordings, number of rāgas, and total audio duration. The main contributions of the present study are:

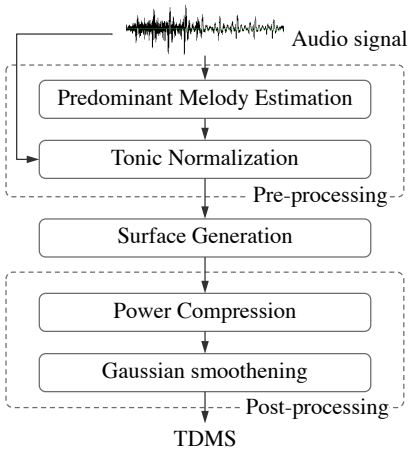


Figure 1. Block diagram for the computation of TDMSs.

- To perform a critical review of the existing methods for rāga recognition and identify some of their main constraints/limitations.
- To propose a novel feature based on delay coordinates, the TDMS, that has all the previously outlined strengths.
- To carry out a comparative evaluation with the best-performing state-of-the-art methods under the same experimental conditions.
- To publicly release a scalable Hindustani music dataset for rāga recognition that contains relevant metadata, annotations, and the computed features.
- To publicly release the code used for the computation of TMDSs and the performed evaluation.

2. RAGA RECOGNITION WITH TIME-DELAYED MELODY SURFACES

2.1 Time-delayed melody surface

The computation of a TDMS has three steps (Figure 1): pre-processing, surface generation, and post-processing. In pre-processing, we obtain a representation of the melody of an audio recording, which is normalized by the tonic or base frequency of the music piece. In surface generation, we compute a two dimensional surface based on the concept of delay coordinates. Finally, in post-processing, we apply power compression and Gaussian smoothing to the computed surface. We subsequently detail these steps.

2.1.1 Predominant melody estimation

We represent the melody of an audio excerpt by the pitch of the predominant melodic source. For predominant pitch estimation, we use the method proposed by Salamon and Gómez [25]. This method performed favorably in MIREX 2011 (an international MIR evaluation campaign) on a variety of music genres, including IAM, and has been used in several other studies for a similar task [7, 12, 13]. We use the implementation of this algorithm as available

in Essentia [1]. Essentia [1] is an open-source C++ library for audio analysis and content-based MIR. We use the default values of the parameters, except for the frame and hop sizes, which are set to 46 and 4.44 ms, respectively. In subsequent steps, we discard frames where a predominant pitch cannot be obtained.

2.1.2 Tonic normalization

The base frequency chosen for a melody in IAM is the tonic pitch of the lead artist [10], to which all other accompanying instruments are tuned. Therefore, for a musically meaningful feature for rāga recognition we normalize the predominant melody of every recording by considering its tonic pitch ω as the reference frequency during the Hertz-to-cent-scale conversion,

$$c_i = 1200 \log_2 \left(\frac{f_i}{\omega} \right),$$

for $0 \leq i < N$, where N is the total number of pitch samples, c_i is the normalized i^{th} sample of the predominant pitch (in cents), and f_i is the i^{th} sample of the predominant pitch (in Hz). The tonic pitch ω for every recording is identified using the multi-pitch approach proposed by Gulati et al. [10]. This approach is reported to obtain state-of-the-art results and has been successfully used elsewhere [8,11]. We use the implementation of this algorithm as available in Essentia with the default set of parameter values. The tonic values for different recordings of an artist are further majority voted to fix the *Pa* (fifth) type error [10].

2.1.3 Surface generation

The next step is to construct a two-dimensional surface based on the concept of delay coordinates (also termed phase space embedding) [15, 28]. In fact, such two-dimensional surface can be seen as a discretized histogram of the elements in a two-dimensional Poicaré map [15]. For a given recording, we generate a surface $\check{\mathbf{S}}$ of size $\eta \times \eta$ recursively, by computing

$$\check{s}_{ij} = \sum_{t=\tau}^{N-1} I\left(B\left(c_t\right), i\right) \; I\left(B\left(c_{t-\tau}\right), j\right)$$

for $0 \leq i, j < \eta$, where I is an indicator function such that $I(x, y) = 1$ iff $x = y$, $I(x, y) = 0$ otherwise, B is an octave-wrapping integer binning operator defined by

$$B(x) = \left\lfloor \left(\frac{\eta x}{1200} \right) \bmod \eta \right\rfloor, \qquad (1)$$

and τ is a time delay index (in frames) that is left as a parameter. Note that, as mentioned, the frames where a predominant pitch could not be obtained are excluded from any calculation. For the size of $\check{\mathbf{S}}$ we use $\eta = 120$. This value corresponds to 10 cents per bin, an optimal pitch resolution reported in [2].

An example of the generated surface $\check{\mathbf{S}}$ for a music piece [2] in rāga Yaman is shown in Figure 2 (a). We see that the prominent peaks in the surface correspond to the svaras of rāga Yaman. We notice that these peaks are steep and that the dynamic range of the surface is high. This can be attributed to the nature of the melodies in these music traditions, particularly in Hindustani music, where the melodies often contain long held svaras. In addition, the dynamic range is high because the pitches in the stable svara regions are within a small range around the svara frequency compared to the pitches in the transitory melodic regions. Because of this, the frequency values in the stable regions are mapped to a smaller set of bins, making the prominent peaks more steep.

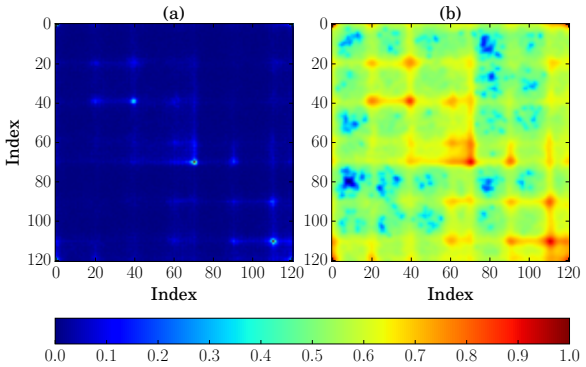


Figure 2. Generated surface for a music piece before (a) and after (b) applying post-processing ($\check{\mathbf{S}}$ and $\hat{\mathbf{S}}$, respectively). For ease of visualization, both matrices are normalized here between 0 and 1.

2.1.4 Post-processing

In order to accentuate the values corresponding to the transitory regions in the melody and reduce the dynamic range of the surface, we apply an element-wise power compression

$$\overline{\mathbf{S}} = \check{\mathbf{S}}^{\alpha},$$

where α is an exponent that is left as a parameter. Once a more compact (in terms of the dynamic range) surface is obtained, we apply Gaussian smoothing. With that, we attempt to attenuate the subtle differences in $\overline{\mathbf{S}}$ corresponding to the different melodies within the same rāga, while retaining the attributes that characterize that rāga.

We perform Gaussian smoothing by circularly convolving $\overline{\mathbf{S}}$ with a two-dimensional Gaussian kernel. We choose a circular convolution because of the cyclic (or octave-folded) nature of the TDMS (Eqn (1)), which mimics the cyclic nature of pitch classes. The standard deviation of this kernel is σ bins (samples). The length of the kernel is truncated to $8\sigma + 1$ bins in each dimension, after which the values are negligible (below 0.01% of the kernel's maximum amplitude). We experiment with different values of σ, and also with a method variant excluding the Gaussian smoothing (loosely denoted by $\sigma = -1$), so that we can quantify its influence on the accuracy of the system.

Once we have the smoothed surface $\hat{\mathbf{S}}$, there is only one step remaining to obtain the final TDMS. Since the overall duration of the recordings and of the voiced regions within

[1] https://github.com/MTG/essentia

[2] http://musicbrainz.org/recording/e59642ca-72bc-466b-bf4b-d82bfbc7b4af

them is different, the computed surface $\hat{\mathbf{S}}$ needs to be normalized. To do so, we divide $\hat{\mathbf{S}}$ by its L_1 matrix norm:

$$\mathbf{S} = \hat{\mathbf{S}} / \|\hat{\mathbf{S}}\|_1.$$

This also yields values of $\mathbf{S}$, the final TDMS, that are interpretable in terms of discrete probabilities.

The result after post-processing the surface of Figure 2 (a) with power compression and Gaussian smoothing is shown in Figure 2 (b). We see that the values corresponding to the non-diagonal elements are accentuated. A visual inspection of Figure 2 (b) provides several musical insights to the melodic aspects of the recording. For instance, the high salience indices along the diagonal, $(0, 0)$, $(20, 20)$, $(40, 40)$, $(60, 60)$, $(70, 70)$, $(90, 90)$, and $(110, 110)$, correspond to the 7 svaras used in rāga Yaman. Within which, the highest salience at indices (110,110) correspond to the Ni svara, which is the *Vadi* svara, i.e., musically the most salient svara of the rāga, in this case rāga Yaman [23]. The asymmetry in the matrix with respect to the diagonal indicates the asymmetric nature of the ascending and descending svara pattern of the rāga (compare, for example, the salience at indices $(70, 90)$ to indices $(90, 70)$, with the former being more salient than the latter). The similarity of the matrix between indices $(20, 20)$ and $(70, 70)$ with respect to the matrix between indices $(70, 70)$ and $(120, 120)$ delineates the tetra-chord structure of the rāga. Finally, it should be noted that an interesting property of TDMSs is that the mean of the sum across its row and columns yields a PCD representation (see Section 1).

2.2 Classification and distance measurement

In order to demonstrate the ability of the TDMSs in capturing rāga characteristics, we consider the task of classifying audio recordings according to their rāga label. To perform classification, we choose a k-nearest neighbor (kNN) classifier [20]. The reasons for our choice are manifold. Firstly, the kNN classifier is well understood, with well studied relations to other classifiers in terms of both performance and architecture. Secondly, it is fast, with practically no training and with known techniques to speed up testing or retrieval. Thirdly, it has only one parameter, k, which we can just blindly set to a relatively small value or can easily optimize in the training phase. Finally, it is a classifier that is simple to implement and whose results are both interpretable and easily reproducible.

The performance of a kNN classifier highly depends on the distance measure used to retrieve the k neighbors. We consider three different measures to compute the distance between two recordings n and m with TDMS features $\mathbf{S}^{(n)}$ and $\mathbf{S}^{(m)}$, respectively. We first consider the Frobenius norm of the difference between $\mathbf{S}^{(n)}$ and $\mathbf{S}^{(m)}$,

$$D_{\mathrm{F}}^{(n,m)} = \|\mathbf{S}_n - \mathbf{S}_m\|_2.$$

Next, we consider the symmetric Kullback-Leibler divergence

$$D_{\mathrm{KL}}^{(n,m)} = D_{\mathrm{KL}}\left(\mathbf{S}^{(n)}, \mathbf{S}^{(m)}\right) + D_{\mathrm{KL}}\left(\mathbf{S}^{(m)}, \mathbf{S}^{(n)}\right),$$

with

$$D_{\mathrm{KL}}\left(\mathbf{X}, \mathbf{Y}\right) = \sum \mathbf{X} \log\left(\frac{\mathbf{X}}{\mathbf{Y}}\right),$$

where we perform element-wise operations and sum over all the elements of the resultant matrix. Finally, we consider the Bhattacharyya distance, which is reported to outperform other distance measures with a PCD-based feature for the same task in [2],

$$D_{\mathrm{B}}^{(n,m)} = -\log\left(\sum \sqrt{\mathbf{S}^{(n)} \cdot \mathbf{S}^{(m)}}\right).$$

We again perform element-wise operations and sum over all the elements of the resultant matrix. Variants of our proposed method that use D_{F}, D_{KL} and D_{B} are denoted by $\mathcal{M}_{\mathrm{F}}$, $\mathcal{M}_{\mathrm{KL}}$, and $\mathcal{M}_{\mathrm{B}}$, respectively.

3. EVALUATION METHODOLOGY

3.1 Music collection

The music collection used in this study is compiled as a part of the CompMusic project [26]. It comprises two datasets: a Carnatic music data set (CMD) and a Hindustani music data set (HMD). Due to the differences in the melodic characteristics within these two music traditions, and for a better analysis of the results, we evaluate our method separately on each of these data sets. CMD and HMD comprise 124 and 130 hours of commercially available audio recordings, respectively, stored as 160 kbps mp3 stereo audio files. All the editorial metadata for each audio recording is publicly available in Musicbrainz [3], an open-source metadata repository. CMD contains full-length recordings of 480 performances belonging to 40 rāgas with 12 music pieces per rāga. HMD contains full-length recordings of 300 performances belonging to 30 rāgas with 10 music pieces per rāga. The selected music material is diverse in terms of the number of artists, the number of forms, and the number of compositions. In these terms, it can be regarded as a representative subset of real-world collections. The chosen rāgas contain diverse sets of svaras (notes), both in terms of the number of svaras and their pitch-classes (svarasthānās).

Note that CMD has already been introduced and made publicly available in [11]. With the same intentions to facilitate comparative studies and to promote reproducible research, we make HMD publicly available online [4]. Along with the rāga labels for each recording, we also make predominant melody, TDMSs, and the code used for our experiments openly available online.

3.2 Comparison with existing methods

In addition to our proposed method, we evaluate and compare two existing methods under the same experimental setup and evaluation data sets. The two selected methods are the ones proposed by Chordia & Şentürk [2], denoted by $\mathcal{E}_{\mathrm{PCD}}$, and by Gulati et al. [11], denoted by $\mathcal{E}_{\mathrm{VSM}}$. Both approaches have shown encouraging results on scalable

[3] https://musicbrainz.org/

[4] http://compmusic.upf.edu/node/300

datasets and can be regarded as the current, most competitive state-of-the-art in rāga recognition. The former, $\mathcal{E}_{\text{PCD}}$, employs PCD-based features computed from the entire audio recording. The latter, $\mathcal{E}_{\text{VSM}}$, uses automatically discovered melodic phrases and vector space modeling. Readers should note that the experimental setup used in [11] is slightly different from the one in the current study. Therefore, there exists a small difference in the reported accuracies, even when evaluated on the same dataset (CMD). For both $\mathcal{E}_{\text{PCD}}$ and $\mathcal{E}_{\text{VSM}}$, we use the original implementations obtained from the respective authors.

3.3 Validation strategy

To evaluate the performance of the considered methods we use the raw overall accuracy [20]. Since both CMD and HMD are balanced in the number of instances per class, we do not need to correct such raw accuracies to counteract for possible biases towards the majority class. We perform a leave-one-out cross validation [20], in which one recording from the evaluation data set forms the testing set and the remaining ones become the training set. To assess if the difference in the performance between any two methods is statistically significant, we use McNemar's test [19] with $p < 0.01$. To compensate for multiple comparisons, we apply the Holm-Bonferroni method [14]. Besides accuracy, and for a more detailed error analysis, we also compute the confusion matrix over the predicted classes.

In the case of $\mathcal{M}$, a test recording is assigned the majority class of its k-nearest neighbors obtained from the training set and, in case of a tie, one of the majority classes is selected randomly. Because we conjecture that none of the parameters we consider is critical to obtain a good performance, we initially make an educated guess and intuitively set our parameters to a specific combination. We later study the influence of every parameter starting from that combination. We initially use $\tau = 0.3$ s, $\alpha = 0.75$, $\sigma = 2$, and $k = 1$, and later consider $\tau \in \{0.2, 0.3, 0.5, 1, 1.5\}$ s, $\alpha \in \{0.1, 0.25, 0.5, 0.75, 1\}$, $\sigma \in \{-1, 1, 2, 3\}$, and $k \in \{1, 3, 5\}$ (recall that $\sigma = -1$ corresponds to no smoothing; Section 2.1.4).

4. RESULTS AND DISCUSSION

In Table 1, we show the results for all the variants of the proposed method $\mathcal{M}_{\text{F}}$, $\mathcal{M}_{\text{KL}}$ and $\mathcal{M}_{\text{B}}$, and the two state-of-the-art methods $\mathcal{E}_{\text{PCD}}$ and $\mathcal{E}_{\text{VSM}}$, using HMD and CMD data sets. We see that the highest accuracy obtained on HMD is 97.7% by $\mathcal{M}_{\text{KL}}$ and $\mathcal{M}_{\text{B}}$. This accuracy is considerably higher than the 91.7% obtained by $\mathcal{E}_{\text{PCD}}$, and the difference is found to be statistically significant. We also see that $\mathcal{E}_{\text{PCD}}$ performs significantly better than $\mathcal{E}_{\text{VSM}}$. Regarding the proposed variants, we see that, in HMD, $\mathcal{M}_{\text{KL}}$ and $\mathcal{M}_{\text{B}}$ perform better than $\mathcal{M}_{\text{F}}$, with a statistically significant difference.

In Table 1, we see that the trend in the performance for CMD across different methods is similar to that for HMD. The variants $\mathcal{M}_{\text{KL}}$ and $\mathcal{M}_{\text{B}}$ achieve the highest accuracy of 86.7%, followed by $\mathcal{E}_{\text{PCD}}$ with 73.1%. The difference between $\mathcal{M}_{\text{KL}}$ ($\mathcal{M}_{\text{B}}$) and $\mathcal{E}_{\text{PCD}}$ is found to be statistically significant. For CMD, also $\mathcal{M}_{\text{KL}}$ and $\mathcal{M}_{\text{B}}$ perform better than $\mathcal{M}_{\text{F}}$, with a statistically significant difference.

Data set	$\mathcal{M}_{\text{F}}$	$\mathcal{M}_{\text{KL}}$	$\mathcal{M}_{\text{B}}$	$\mathcal{E}_{\text{PCD}}$	$\mathcal{E}_{\text{VSM}}$
HMD	91.3	**97.7**	**97.7**	91.7	83.0
CMD	81.5	**86.7**	**86.7**	73.1	68.1

Table 1. Accuracy (%) of the three proposed variants, $\mathcal{M}_{\text{F}}$, $\mathcal{M}_{\text{KL}}$ and $\mathcal{M}_{\text{BC}}$, and the two existing state-of-the-art methods $\mathcal{E}_{\text{PCD}}$ and $\mathcal{E}_{\text{VSM}}$ (see text). The random baseline for this task is 3.3% for HMD and 2.5% for CMD.

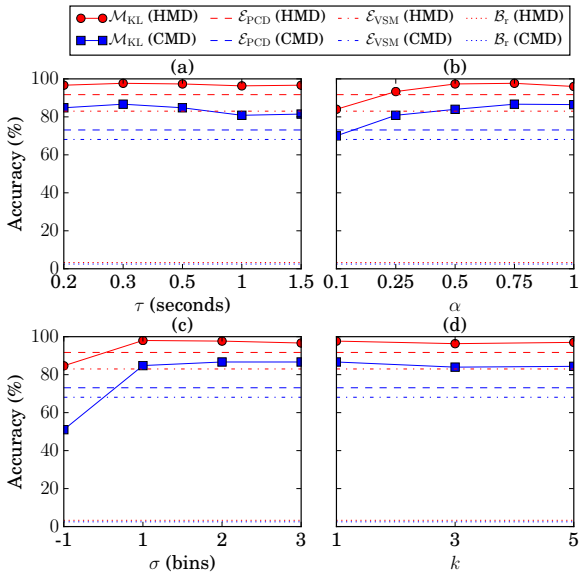


Figure 3. Accuracy of $\mathcal{M}_{\text{KL}}$ as a function of parameter values. State-of-the-art approaches $\mathcal{E}$ and random baselines $\mathcal{B}$ are also reported for comparison.

In general, we notice that, for every method, the accuracy is higher on HMD compared to CMD. This, as expected, can be largely attributed to the difference in the number of classes in HMD (30 rāgas) and CMD (40 rāgas). A higher number of classes makes the task of rāga recognition more challenging for CMD, compared to HMD. In addition to that, another factor that can cause this difference could be the length of the audio recordings, which for HMD are significantly longer than the ones in CMD.

As mentioned earlier, the system parameters corresponding to the results in Table 1 were set intuitively, without any parameter tuning. Since TDMSs are used here for the first time, we want to carefully analyze the influence that each of the parameters has on the final rāga recognition accuracy, and ultimately perform a quantitative assessment of their importance. In Figure 3, we show the accuracy of $\mathcal{M}_{\text{KL}}$ for different values of these parameters. In each case, only one parameter is varied and the rest are set to the initial values mentioned above.

In Figure 3 (a), we observe that the performance of the method is quite invariant to the choice of τ, except for the extreme delay values of 1 and 1.5 s for CMD. This

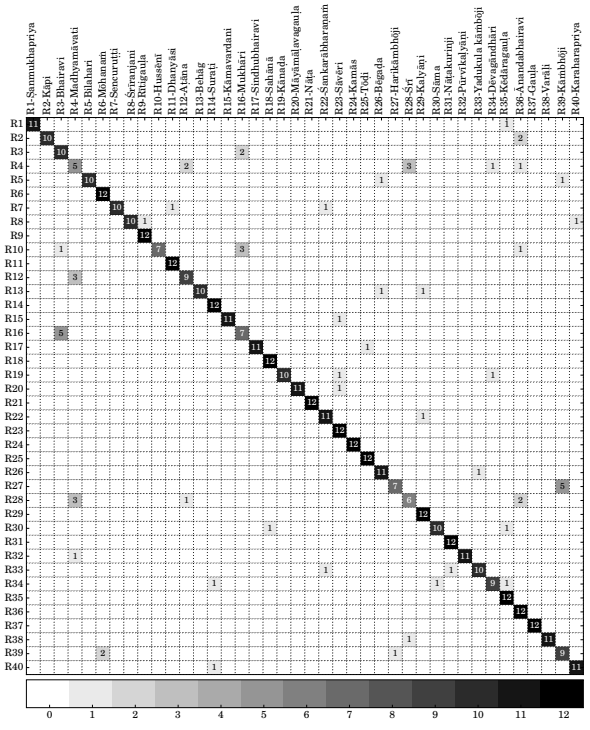


Figure 4. Confusion matrix of the predicted rāga labels obtained by $\mathcal{M}_{\mathrm{KL}}$ on CMD. Shades of grey are mapped to the number of audio recordings.

can be attributed to the melodic characteristics of Carnatic music, which presents a higher degree of oscillatory melody movements and shorter stationary svara regions, as compared to Hindustani music. In Figure 3 (b), we see that compression with $\alpha < 1$ slightly improves the performance of the method for both data sets. However, the performance degrades for $\alpha < 0.75$ for CMD and $\alpha < 0.25$ for HMD. This again appears to be correlated with the long steady nature of the svaras in Hindustani music melodies. Because the dynamic range of $\tilde{\mathbf{S}}$ is high, TDMS features require a lower value for the compression factor α to accentuate the surface values corresponding to the transitory regions in the melodies of Hindustani music. In Figure 3 (c), we observe that Gaussian smoothing significantly improves the performance of the method, and that such performance is invariant across the chosen values of σ. Finally, in Figure 3 (d), we notice that the accuracy decreases with increasing k. This is also expected due to the relatively small number of samples per class in our data sets [20]. Overall, the method appears to be invariant to different parameter values to a large extent, which implies that it is easier to extend and tune it to other data sets.

From the results reported in Figure 3, we see that there exist a number of parameter combinations that could potentially yield a better accuracy than the one reported in Table 1. For instance, using $\tau = 0.3$ s, $\alpha = 0.5$, $\sigma = 2$, and $k = 1$, we are able to reach 97.0% for $\mathcal{M}_{\mathrm{F}}$ and 98.0% for both $\mathcal{M}_{\mathrm{KL}}$ and $\mathcal{M}_{\mathrm{B}}$ on HMD. These accuracies are ad-hoc, optimizing the parameters on the testing set. However, and doing things more properly, we could learn the optimal parameters in training, through a standard grid search, cross-validated procedure over the training set [20]. As our primary goal here is not to obtain the best possible results, but to show the usefulness and superiority of TDMSs, we do not perform such an exhaustive parameter tuning and leave it for future research.

To conclude, we proceed to analyze the errors made by the best performing variant $\mathcal{M}_{\mathrm{KL}}$. For CMD, we show the confusion matrix of the predicted rāga labels in Figure 4. In general, we see that the confusions have a musical explanation. The majority of them are between the rāgas in the sets {Bhairavi, Mukhāri}, {Harikāmbhōji, Kāmbhōji}, {Madhyamavatī, Aṭāna, Śrī}, and {Kāpi, Ānandabhairavi}. Rāgas within each of these sets are allied rāgas [29], i.e., they share a common set of svaras and similar phrases. For HMD, there are only 7 incorrectly classified recordings (confusion matrix omitted for space reasons). Rāga Alhaiyā bilāwal and rāga Dēś is confused with rāga Gauḍ Malhār, which is musically explicable as these rāgas share exactly the same set of svaras. Rāga Rāgēśhrī is confused with Bāgēśhrī, which differ in only one svara. In all these cases, the rāgas which are confused also have similar melodic phrases. For two specific cases of confusions, that of rāga Khamāj with Bāgēśhrī, and rāga Darbārī with Bhūp, we find that the error lies in the estimation of the tonic pitch.

5. CONCLUSION

In this paper, we proposed a novel melody representation for rāga recognition, the TDMS, which is inspired by the concept of delay coordinates and Poicaré maps. A TDMS captures both the tonal and the short-time temporal characteristics of a melody. They are derived from the tonic-normalized pitch of the predominant melodic source in the audio. To demonstrate the capabilities of TDMSs in capturing rāga characteristics, we classified audio recordings according to their rāga labels. For this, we used sizable collections of Hindustani and Carnatic music with over 250 hours of duration. Using a k-nearest neighbor classifier, the proposed feature outperformed state-of-the-art systems in rāga recognition. We also studied the influence of different parameters on the accuracy obtained by TDMSs, and found that it is largely invariant to different parameter values. An analysis of the classification errors revealed that the confusions occur between musically similar rāgas that share a common set of svaras and have similar melodic phrases. In the future, we plan to investigate if PCD-based, phrase-based, and TDMSs can be successfully combined to improve rāga recognition. In addition, we would like to investigate the minimum duration of the audio recording needed to successfully recognize its rāga.

6. ACKNOWLEDGMENTS

This work is partly supported by the European Research Council under the European Unions Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583).

7. REFERENCES

[1] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X. Serra. Essentia: an audio analysis library for music information retrieval. In *Proc. of Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, pages 493–498, 2013.

[2] P. Chordia and S. Şentürk. Joint recognition of raag and tonic in north Indian music. *Computer Music Journal*, 37(3):82–98, 2013.

[3] A. Danielou. *The ragas of Northern Indian music*. Munshiram Manoharlal Publishers, New Delhi, 2010.

[4] P. Dighe, P. Agrawal, H. Karnick, S. Thota, and B. Raj. Scale independent raga identification using chromagram patterns and swara based features. In *IEEE Int. Conf. on Multimedia and Expo Workshops (ICMEW)*, pages 1–4, 2013.

[5] P. Dighe, H. Karnick, and B. Raj. Swara histogram based structural analysis and identification of Indian classical ragas. In *Proc. of Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, pages 35–40, 2013.

[6] S. Dutta, S. PV Krishnaraj, and H. A. Murthy. Raga verification in Carnatic music using longest common segment set. In *Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, pages 605–611, 2015.

[7] S. Dutta and H. A. Murthy. Discovering typical motifs of a raga from one-liners of songs in Carnatic music. In *Int. Soc. for Music Information Retrieval (ISMIR)*, pages 397–402, 2014.

[8] K. K. Ganguli, A. Rastogi, V. Pandit, P. Kantan, and P. Rao. Efficient melodic query based audio search for Hindustani vocal compositions. In *Proc. of Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, pages 591–597, 2015.

[9] T. Ganti. *Bollywood: a guidebook to popular Hindi cinema*. Routledge, 2013.

[10] S. Gulati, A. Bellur, J. Salamon, H. G. Ranjani, V. Ishwar, H. A. Murthy, and X. Serra. Automatic tonic identification in Indian art music: approaches and evaluation. *Journal of New Music Research*, 43(1):55–73, 2014.

[11] S. Gulati, J. Serrà, V. Ishwar, S. Şentürk, and X. Serra. Phrase-based rāga recognition using vector space modeling. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 66–70, 2016.

[12] S. Gulati, J. Serrà, V. Ishwar, and X. Serra. Mining melodic patterns in large audio collections of Indian art music. In *Int. Conf. on Signal Image Technology & Internet Based Systems (SITIS-MIRA)*, pages 264–271, 2014.

[13] S. Gulati, J. Serrà, and X. Serra. An evaluation of methodologies for melodic similarity in audio recordings of Indian art music. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 678–682, 2015.

[14] S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, 6(2):65–70, 1979.

[15] H. Kantz and T. Schreiber. *Nonlinear time series analysis*. Cambridge University Press, Cambridge, UK, 2004.

[16] G. K. Koduri, V. Ishwar, J. Serrà, and X. Serra. Intonation analysis of rāgas in Carnatic music. *Journal of New Music Research*, 43(1):72–93, 2014.

[17] T. M. Krishna and V. Ishwar. Karṇāṭic music: Svara, gamaka, motif and rāga identity. In *Proc. of the 2nd CompMusic Workshop*, pages 12–18, 2012.

[18] V. Kumar, H Pandya, and C. V. Jawahar. Identifying ragas in Indian music. In *22nd Int. Conf. on Pattern Recognition (ICPR)*, pages 767–772, 2014.

[19] Q. McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

[20] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, USA, 1997.

[21] P. V. Rajkumar, K. P. Saishankar, and M. John. Identification of Carnatic raagas using hidden markov models. In *IEEE 9th Int. Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 107–110, 2011.

[22] S. Rao. Culture specific music information processing: A perspective from Hindustani music. In *2nd CompMusic Workshop*, pages 5–11, 2012.

[23] S. Rao, J. Bor, W. van der Meer, and J. Harvey. *The raga guide: a survey of 74 Hindustani ragas*. Nimbus Records with Rotterdam Conservatory of Music, 1999.

[24] S. Rao and P. Rao. An overview of Hindustani music in the context of computational musicology. *Journal of New Music Research*, 43(1):24–33, 2014.

[25] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, 2012.

[26] X. Serra. A multicultural approach to music information research. In *Proc. of Int. Conf. on Music Information Retrieval (ISMIR)*, pages 151–156, 2011.

[27] S. Shetty and K. K. Achary. Raga mining of Indian music by extracting arohana-avarohana pattern. *Int. Journal of Recent Trends in Engineering*, 1(1):362–366, 2009.

[28] F. Takens. Detecting strange attractors in turbulence. *Dynamical Systems and Turbulence, Warwick 1980*, pages 366–381, 1981.

[29] T. Viswanathan and M. H. Allen. *Music in South India*. Oxford University Press, 2004.

TRANSCRIBING HUMAN PIANO PERFORMANCES INTO MUSIC NOTATION

Andrea Cogliati[*], David Temperley[+], Zhiyao Duan[*]
Electrical and Computer Engineering[*] and Eastman School of Music[+], University of Rochester
{andrea.cogliati,zhiyao.duan}@rochester.edu
dtemperley@esm.rochester.edu

ABSTRACT

Automatic music transcription aims to transcribe musical performances into music notation. However, existing transcription systems that have been described in research papers typically focus on multi-F0 estimation from audio and only output notes in absolute terms, showing frequency and absolute time (a piano-roll representation), but not in musical terms, with spelling distinctions (e.g., A♭ versus G♯) and quantized meter. To complete the transcription process, one would need to convert the piano-roll representation into a properly formatted and musically meaningful musical score. This process is non-trivial and largely unresearched. In this paper we present a system that generates music notation output from human-recorded MIDI performances of piano music. We show that the correct estimation of the meter, harmony and streams in a piano performance provides a solid foundation to produce a properly formatted score. In a blind evaluation by professional music theorists, the proposed method outperforms two commercial programs and an open source program in terms of pitch notation and rhythmic notation, and ties for the top in terms of overall voicing and staff placement.

1. INTRODUCTION

Automatic Music Transcription (AMT) is the process of inferring a symbolic music representation from a music performance, such as a live performance or a recording. The output of AMT can be a full musical score or an intermediate representation, such as a MIDI file [5]. AMT has several applications in music education (e.g., providing feedback to piano students), content-based music search (e.g., searching for songs with a similar chord progression or bassline), musicological analysis of non-notated music (e.g., jazz improvisations and most non-Western music), and music enjoyment (e.g., visual representation of the music content).

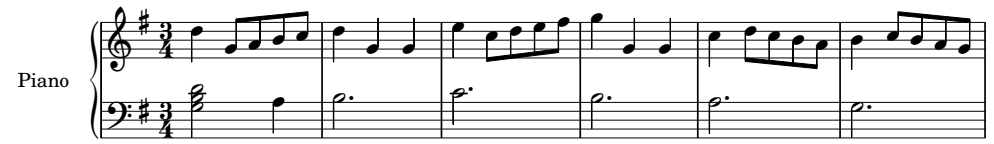
Intermediate representations are closer to the audio file even though they identify certain kinds of musical information that are not readily accessible, such as explicit pitches and note onsets. However, the encoding of this information is generally not done in abstract musical terms but still reflects some of the arbitrariness of a human performance; e.g., note onsets may be expressed in terms of absolute times instead of being quantized to a meter, and pitches may be expressed in terms of frequency or MIDI note numbers, instead of proper note spelling, e.g., C♯ has the same MIDI note number as D♭. Music notation provides further information such as key signature, time signature, rhythmic values, barlines, and voicing (e.g., the representation of multiple voices with upward and downward stems); this information is useful and indeed virtually necessary for further performance and analysis [13].

While AMT was initially formulated as a method to convert musical sounds into common music notation [15], most AMT systems so far have opted for lower level representations [5]; very few systems have attempted to estimate higher level musical information, such as beats or pattern repetitions, directly from the audio [9, 14]. Higher level musical information can also be estimated from an intermediate representation [6, 18]. In this paper we opt for the latter approach; this allows the conversion of MIDI to notation, and eventually (in combination with an audio-to-MIDI conversion system, such as [8]) could generate notation from audio as well.

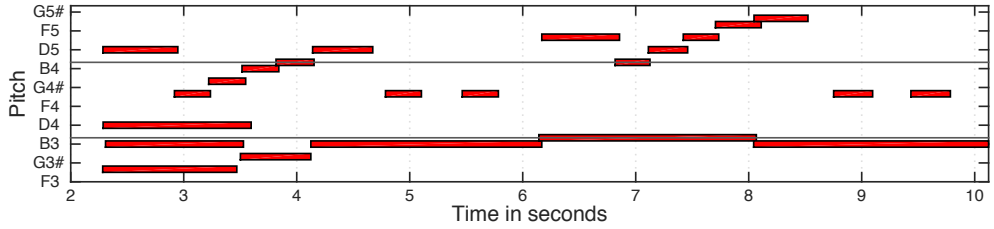
A MIDI file can represent a piano performance very accurately; in fact, the only variables involved are note onset, offset, velocity and pedal activation. Moreover, MIDI representations of piano performances can be recorded from a MIDI keyboard, or from a piano with key sensors. The MIDI standard is capable of encoding high-level musical information, such as key and time signatures, into MIDI files, but this information is not typically included in recorded performances, unless the performer manually inserts it. Furthermore, recorded MIDI performances are typically unquantized, as performers continuously change the speed of playing to obtain a more expressive performance, and may play certain notes slightly earlier or later than they should be to highlight certain musical lines.

The process of producing a correct full music notation from an unquantized and un-annotated MIDI file is non-trivial and, to the authors' knowledge, no system capable of producing full music notation has been implemented and documented in academic research papers thus far. Without a proper estimation of the meter and the har-

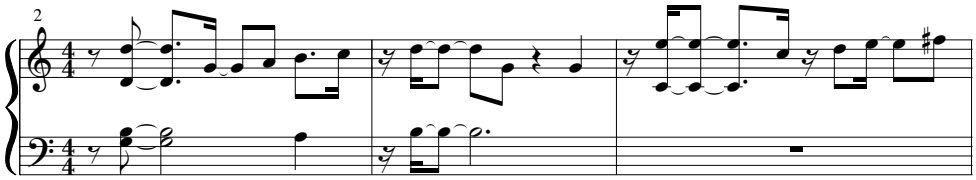
© Andrea Cogliati[*], David Temperley[+], Zhiyao Duan[*]. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Andrea Cogliati[*], David Temperley[+], Zhiyao Duan[*]. "Transcribing Human Piano Performances Into Music Notation", 17th International Society for Music Information Retrieval Conference, 2016.



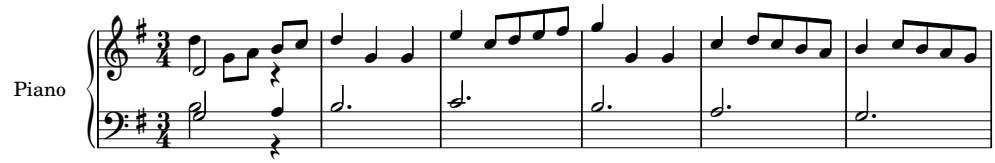
(a) Ground Truth



(b) Unquantized MIDI piano roll



(c) GarageBand



(d) Proposed method

Figure 1. Transcription of a performance of the Minuet in G from Bach's Notebook for Anna Magdalena Bach. (a) shows the original score (b) shows the unquantized pianoroll of a MIDI performance. (c) shows the output from GarageBand, which does not perform any analysis on the MIDI file. (d) shows the output of the proposed method after estimating the correct meter, key signature, beats and streams. The music excerpts are of different lengths for better formatting.

mony, the results are very poor – see Fig. 1 (c). The task can be divided into two main sub-tasks: musical structure analysis and note placement on the score. For the first sub-task, the MIDI file must be analyzed to estimate the key signature and the correct note spelling, as well as the beats and the correct time signature. For the second sub-task, once the notes have been correctly spelled and quantized to the correct meter, they must be properly positioned on the staff. Piano music is normally notated on two staves. The higher staff is usually notated in treble clef, and contains the notes generally played by the right hand. The lower staff is usually notated in bass clef, and contains the notes generally played by the left hand. Notes should be placed on staves to simplify the reading of the score, e.g., notes should be well spaced and typographical elements should not clash with each other. The placement of the notes and other typographical elements also convey musical meanings, e.g., notes pertaining to the same voice should have the stems pointing in the same direction and beaming should follow the rhythm of the musical passage. Finally, concurrent notes played by a single hand as chords should share the same stem. Exceptions to these basic rules are not uncommon, typically to simplify the reading by a performer, e.g., if a passage requires both hands to play in the higher range of the piano keyboard, both staves may be notated in the treble clef to avoid too many ledger lines and too many notes on the same staff.

In this paper we present a novel method to fully notate a piano performance recorded as an unquantized and un-annotated MIDI file, in which only the note pitches (MIDI number), onsets and offsets are considered. The initial analysis of the piece is done through a probabilistic model proposed by Temperley to jointly estimate meter, harmony and streams [18]. The engraving of the score is done through the free software LilyPond [1]. The evaluation dataset and the Python code are available on the first author's web site [1].

[1] http://www.ece.rochester.edu/~acogliat/

2. RELATED WORKS

There are several free and commercial programs, such as Finale, Sibelius and MuseScore, that can import MIDI files and translate them into full music notation, but they typically require user intervention to inform the process to a certain degree. For instance, Finale requires the user to manually select the time signature, while it can infer the key signature from the file itself. Certain sequencers and Digital Audio Workstations, such as GarageBand and Logic Pro, have various functions to facilitate the import of MIDI files; for example, Logic Pro has a function to align the time track to the beats in the MIDI files, but requires the user to input the time signature and estimate the initial tempo of the piece.

Among the programs used for the evaluation of the proposed method, MuseScore [2] has the most advanced MIDI file import feature. MuseScore has a specific option to import human performances, and is capable of estimating the meter and the key signature. During the experiment, MuseScore showed a sophisticated capability to position different voices on the piano staves, which resulted in high scores from the evaluators, especially in terms of overall voicing and staff placement. Unfortunately, details on how all these steps are performed are not documented in the website [2] and have not been published in research papers.

The task of identifying musical structures from a MIDI performance has been extensively researched, especially in the past two decades. Cambouropoulos [6] describes the key components necessary to convert a MIDI performance into musical notation: identification of elementary musical objects (i.e., chords, arpeggiated chords, and trills), beat identification and tracking, time quantization and pitch spelling. However, the article does not describe how to render a musical score from the modules presented. Takeda et al. [16] describe a Hidden Markov Model for the automatic transcription of monophonic MIDI performances. In his PhD thesis, Cemgil [7] presents a Bayesian framework for music transcription, identifying some issues related to automatic music typesetting (i.e., the automatic rendering of a musical score from a symbolic representation), in particular, tempo quantization, and chord and melody identification. Karydis et al. [12] proposes a perceptually motivated model for voice separation capable of grouping polyphonic groups of notes, such as chords or other forms of accompaniment figures, into a perceptual stream. A more recent paper by Grohganz et al. [11] introduces the concepts of score-informed MIDI file (S-MIDI), in which musical tempo and beats are properly represented, and performed MIDI file (P-MIDI), which records a performance in absolute time. The paper also presents a procedure to approximate an S-MIDI file from a P-MIDI file – that is, to detect the beats and the meter implied in the P-MIDI file, starting from a tempogram then analyzing the beat inconsistency with a salience function based on autocorrelation.

Musical structures can also be inferred directly from audio. Ochiai et al. [14] propose a model for the joint estimation of note pitches, onsets, offsets and beats based on Non-negative Matrix Factorization constrained with a rhythmic structure modeled with a Gaussian mixture model. Collins et al. [9] propose a model for multiple F0 estimation, beat tracking, quantization, and pattern discovery. The pitches are estimated with a neural network. A Hidden Markov Model (HMM) is separately used for beat tracking. The results are then combined to quantize the notes. Note spelling is performed by estimating the key of the piece and assigning to MIDI notes the most probable pitch class given the key.

3. PROPOSED METHOD

The proposed method takes an unquantized and unannotated MIDI file as input. The following subsections explain each step in the proposed method. The entire process is illustrated in Fig. 2. An example of the output is shown in Fig. 1 (d).

3.1 Fix spurious overlapping notes

The first step is to fix spurious overlapping notes. Piano players do not play notes with the correct length all the time. As we can see from Fig. 1 (b), certain notes are played shorter than they should be, resulting in gaps between notes, while other notes are played longer than they should be, resulting in overlapping notes. Gaps between notes in the same melodic line might result in extra rests in the score, while overlapping notes might result in extra streams being created by the probabilistic model [18] used in the next step, resulting in extra voices in the final score. In particular, the probabilistic model used in this paper always assigns overlapping notes to different streams, so it is critical to remove erroneous overlaps.

To estimate whether the overlap is correct or wrong we consider pairs of overlapping notes separately. For each pair, we calculate one overlapping ratio for each note. The ratio is defined as the length of the overlapping region over the length of the note. The overlap is considered spurious if the sum of the two ratios is below a certain threshold. For the experiment we set a threshold of 30%. The output of the first step is a note list, i.e., a list of note events, each including an onset, a duration (both in milliseconds), and a MIDI note number. An example is shown in Fig. 3. Notice the small overlaps in the top figure between the three low notes in the initial chord and the second bass note, as well as the short overlaps in the scale in the soprano line; these are removed in the second figure. Also notice that correct overlapping notes, such as a melody line moving over the same bass note, are preserved.

3.2 Estimate meter, harmony and streams

In the second step, we apply the probabilistic model [18] to the note list. The probabilistic model estimates the meter, the harmony, and the streams. The meter and harmony are estimated in a single joint process. This process is modeled as an HMM and is based on the concept of tactus-root combination (TRC), a combination of two adjacent tactus

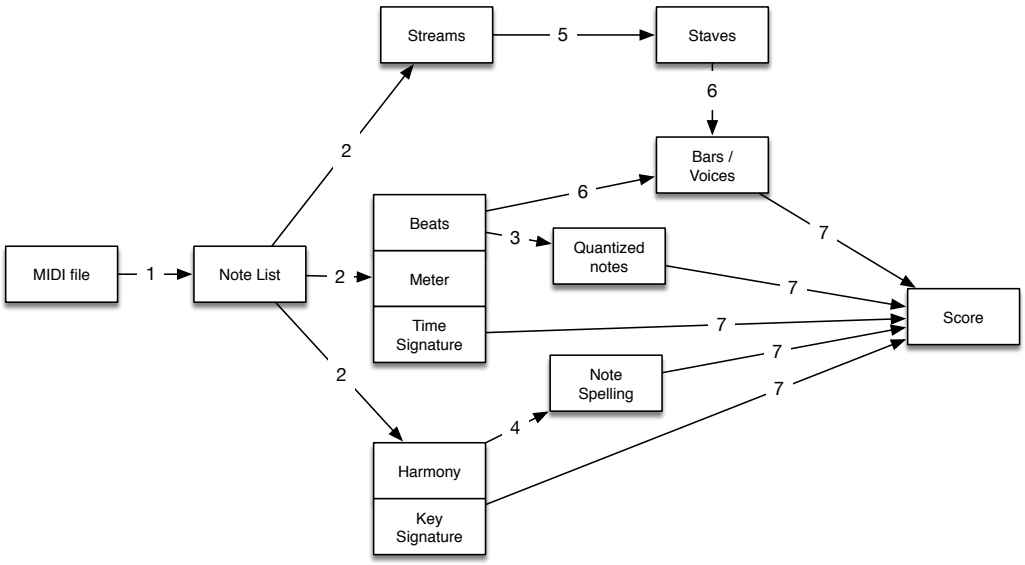
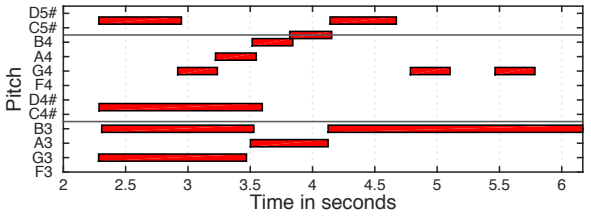
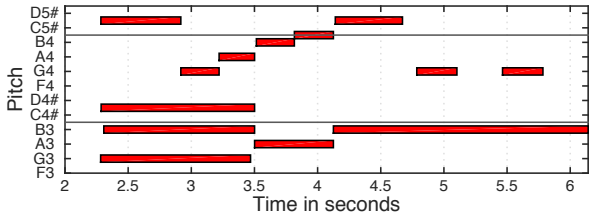


Figure 2. Illustration of the proposed method. The arrows indicate dependencies between entities. The numbers refer to the steps (subsection numbers) in Section 3.



(a) Original pianoroll of a MIDI performance



(b) Pianoroll after fixing spurious overlapping notes

Figure 3. An example of the step of fixing spurious overlapping notes.

beats and a chord root. The probability of a TRC only depends on the previous TRC, and the probability of beats and notes within a TRC only depends on the TRC. The musical intuition behind this is that the "goodness" of a tactus interval depends only on its relationship to the previous tactus interval (with a preference to minimize changes in length from one interval to the next), the goodness of a root depends only on the previous root (with a preference to maintain the same root if possible, or to move to another root that is a fifth away), and the goodness of a particular pattern of notes within a short time interval depends only on the current root and the placement of beats within that interval (with a preference for note onsets on tactus beats or at plausible points–e.g., roughly halfway–in between them, and a preference for notes that are chord-tones of the root). The process also considers different divisions of the tactus interval (representing simple or compound meter) and placements of strong beats (duple versus triple meter). In the current context, the metrical analysis is useful for the placement of barlines and for rhythmic notation; the harmonic analysis is useful for pitch spelling, and also influences the metrical analysis, since there is a preference for strong beats at changes of harmony (this is the reason for estimating the meter and harmony jointly). The stream segregation problem is solved with dynamic programming by grouping notes into streams such that the number of streams, the number and length of rests within streams, and pitch intervals within streams are all minimized [18].

The output of the probabilistic model is a list of beats, notes, and chord roots. Each beat includes an onset in milliseconds, and a level in a metrical hierarchy [17]. The probabilistic model considers the tactus and two subdivisions in the metrical structure; e.g., in a 3/4 meter, the tactus will be the quarter note, the first subdivision will be the 8th note, and the lowest subdivision the 16th note. The metrical structure also indicates the downbeats. Each note has an onset and a duration in milliseconds, a midi note number, and a stream number. The chord roots are quantized to the beats. An example of the output of this stage is shown in Fig. 4.

3.3 Quantize notes

The third step quantizes the note onsets to the closest beat subdivision. The offset of each note is also set to coincide with the onset of the next note in a stream; i.e., gaps within each stream are discarded. This avoids extra rests in the final scores, which could stem from notes played

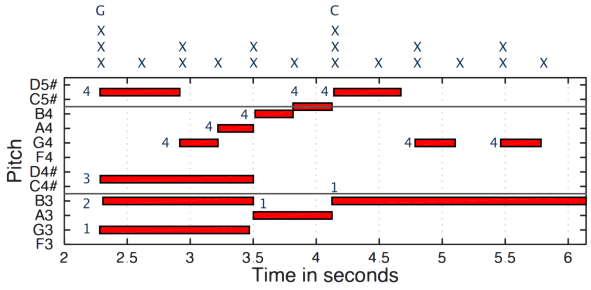


Figure 4. Sample output of the probabilistic model for estimating the metrical, harmonic, and stream structures. The Xs above the pianoroll illustrate the meter analysis (only 3 levels displayed). The letters above show the chord root (only roots on the downbeats are shown). The numbers next to the notes indicate the stream.

shorter than they should be. See, for instance, the two quarter notes in stream 4 in the second bar of the pianoroll in Fig. 4; they were played slightly shorter than 8th notes.

3.4 Determine note spelling

The correct note spelling is determined from the harmony generated by the probabilistic model and is based on the proximity in the line of fifths (the circle of fifths stretched out into a line) to the chord root. For example, the MIDI note 66 (F♯/G♭) would be spelled F♯ on a root of D, but spelled as G♭ on a root of E♭.

3.5 Assign streams to staves

The staves of the final score are set to be notated in treble clef for the upper staff and bass clef for the lower staff. Streams are assigned to the staff that accommodates all the notes with the fewest number of ledger lines.

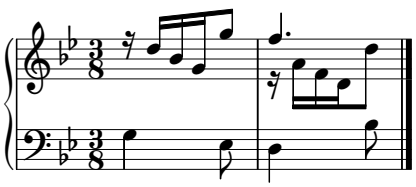


Figure 5. First two measures of Bach's Sinfonia in G minor, BWV 797. In the second bar, two streams are assigned to the same staff, so two separate monophonic voices must be created for proper rendering.

3.6 Detect concurrent voices

Once streams have been assigned to staves, we determine bars and voices. Bars are easily determined by the metrical structure, but note adjustments might be necessary if a note starts in one bar and continues to the next bar. In that case, the note has to be split into two or more tied notes. Concurrent notes in the same bar and staff must be detected and encoded appropriately for the next step. If a staff contains streams that overlap in time, we create monophonic voices consisting of sequences of notes. A sequence is defined as a gapless succession of notes and rests without overlaps. For example, as shown in Fig. 5, concurrent streams in measure 1 can be treated as monophonic inputs as they are assigned to separate staves, but in measure 2, two concurrent streams are assigned to the same staff, so we have to create two monophonic sequences of notes as input for the next step, one containing the F dotted quarter, the other containing the 16th notes and the D 8th note.

3.7 Generate the score

Finally, a Lilypond input file is generated. Lilypond is a free, command-line oriented music engraving program, which takes a text file as input and, thus, is suitable for the automatic generation of music notation. A possible alternative to Lilypond, which was considered during our research, is MusicXML [10]. Lilypond has the advantage of a simpler and more concise syntax. For instance, the music example from [10], which requires 130 lines of MusicXML, only requires 12 lines in Lilypond.

4. EVALUATION

To evaluate the proposed method, we asked five doctoral students in the Music Theory department of the Eastman School of Music, at various stages of advancement in their program, to blindly rate the output of the proposed method, two commercial programs (Finale 2015 [3] and GarageBand 10 [4]) and a free engraving program (MuseScore 2) applied to the Kostka-Payne dataset used to evaluate the probabilistic model [18]. The commercial programs have been chosen due to their popularity: GarageBand is freely available to all Mac users, Finale is one of the two major commercial music notation programs, the other being Sibelius. We also tested the import functionality of Sibelius but the results were very similar to the ones obtained by Finale, so we dropped this dataset to save time during the human evaluation. The dataset comprises 19 music excerpts, all of them piano pieces by well-known composers, for a total of 76 music scores to evaluate. The pieces were performed on a MIDI keyboard by a semi-professional piano player. For each piece we provided the original score, i.e., the ground truth. All the scores had been anonymized, so that the source program name was unknown, and the order of the evaluation was randomized. The evaluators were asked the following questions: 1) Rate the pitch notation with regard to the key signature and the spelling of notes. 2) Rate the rhythmic notation with regard to the time signature, bar lines, and rhythmic values. 3) Rate the notation with regard to stems, voicing, and placement of notes on staves. These three questions summarize the most important features that determine the formatting and the readability of a musical score. The three features are also fairly independent of each other.

The ratings were on a scale from 1 to 10 – 10 being the best. We instructed the evaluators to rate the scores to reflect how close each output was to the ground truth.

Finally, we told the evaluators that, since each rating may reflect multiple aspects of the notation, it was entirely up to their judgment to decide how to balance them (e.g., the relative importance of time signature, barline placement, and rhythmic values for the second question).

The results are shown in Figs. 6, 7 and 8. The ratings from each evaluator have been normalized (z-scores) by subtracting the mean and dividing by the standard deviation, and the results have been rescaled to the original range by setting their mean to 5 and their standard deviation to 2. The proposed method outperforms all the other methods in the first two ratings – pitch notation and rhythm notation – and ties for the top in median for the third rating – voicing and staff placement. Paired sign tests show that the ratings of the proposed method are significantly better than all the three baselines for the first two aspects, at a significance level of $p = 0.0001$. For the third aspect, the proposed method is superior to Finale and equivalent to MuseScore at a significance level of $p = 0.0001$, while the comparison with GarageBand is statistically inconclusive.

More work is needed in the note placement. One common aspect of music notation that has not been addressed in the proposed method is how to group concurrent notes into chords; we can see how that affects the output in Fig. 1 (d). In the downbeat of the first bar, the lowest three notes are not grouped into a chord, as in the ground truth (Fig. 1 (a)). This makes the notation less readable, and also introduces an unnecessary rest in the upper staff. A possible solution to this problem consists in grouping into chords notes that have the same onset and duration, and that are not too far apart, i.e., so that they could be played by one hand. A possible drawback of this approach is that it may group notes belonging to different voices.

Another limitation of the proposed method is the positioning of concurrent voices in polyphonic passages. Currently, the proposed method relies on the streams detected in step 2 to determine the order in which the voices are positioned in step 6. In polyphonic music, voices can cross so the relative positioning of voices might be appropriate for certain bars but not for others. A possible solution is to introduce another step between 6 and 7 to analyze each single measure and determine whether the relative positions of the voice is optimal or not. These two limitations affect the note positioning, reflected in the scores shown in Fig. 8. Finally, the probabilistic model does not always produce the correct results, especially with respect to beats and streams. A more sophisticated model may improve the rhythm notation and the note positioning, reflected in the scores shown in Fig. 7 and Fig. 8.

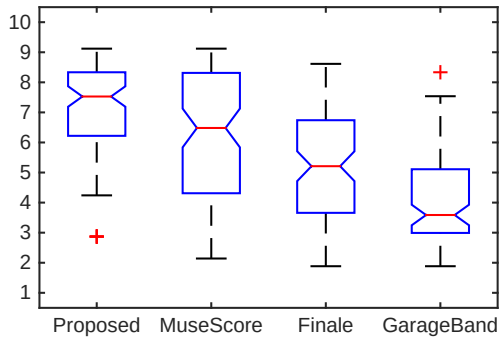


Figure 6. Normalized pitch notation ratings. Each box contains 76 scores from each of the 5 evaluators.

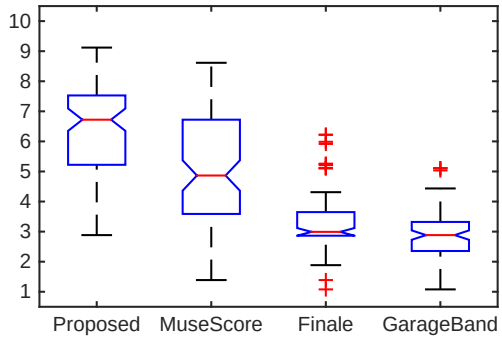


Figure 7. Normalized rhythm notation ratings. Each box contains 76 scores from each of the 5 evaluators.

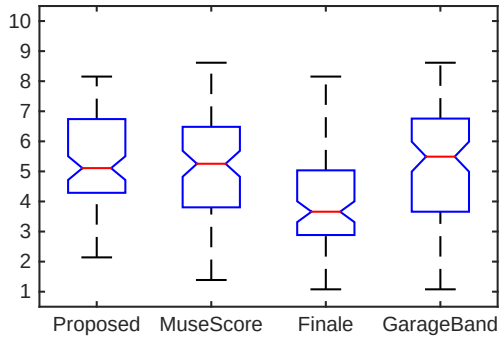


Figure 8. Normalized note positioning ratings. Each box contains 76 scores from each of the 5 evaluators.

5. CONCLUSIONS

In this paper we presented a novel method to generate a music notation output from a piano-roll input of a piano performance. We showed that the correct estimation of the meter, harmony and streams is fundamental in producing a properly formatted score. In a blind evaluation by professional music theorists, the proposed method consistently outperforms two commercial programs and an open source program in terms of pitch notation and rhythmic notation, and ties for the top in voicing and staff placement on 19 human performances on a MIDI keyboard. The proposed method can also be combined with any note-level automatic music transcription method to complete the audio to music notation conversion process, but more experiments are needed to assess the performance. For future work, we also plan to design a transcription metric for objective evaluation on a larger dataset, which should include complete piano pieces.

6. REFERENCES

[1] http://lilypond.org.

[2] https://musescore.org.

[3] http://www.finalemusic.com.

[4] http://www.apple.com/mac/garageband/.

[5] Emmanouil Benetos, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434, 2013.

[6] Emilios Cambouropoulos. From MIDI to traditional musical notation. In *Proceedings of the AAAI Workshop on Artificial Intelligence and Music: Towards Formal Models for Composition, Performance and Analysis*, volume 30, 2000.

[7] Ali Taylan Cemgil. *Bayesian music transcription*. PhD thesis, Radboud University Nijmegen, 2004.

[8] Andrea Cogliati, Zhiyao Duan, and Brendt Wohlberg. Piano music transcription with fast convolutional sparse coding. In *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*, pages 1–6, Sept 2015.

[9] Tom Collins, Sebastian Böck, Florian Krebs, and Gerhard Widmer. Bridging the audio-symbolic gap: The discovery of repeated note content directly from polyphonic music audio. In *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. Audio Engineering Society, 2014.

[10] Michael Good. MusicXML for notation and analysis. *The virtual score: representation, retrieval, restoration*, 12:113–124, 2001.

[11] Harald Grohganz, Michael Clausen, and Meinard Müller. Estimating musical time information from performed midi files. In *ISMIR*, pages 35–40, 2014.

[12] Ioannis Karydis, Alexandros Nanopoulos, Apostolos Papadopoulos, Emilios Cambouropoulos, and Yannis Manolopoulos. Horizontal and vertical integration/segregation in auditory streaming: a voice separation algorithm for symbolic musical data. In *Proceedings 4th Sound and Music Computing Conference (SMC'2007)*, 2007.

[13] Meinard Müller. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer, 2015.

[14] Kazuki Ochiai, Hirokazu Kameoka, and Shigeki Sagayama. Explicit beat structure modeling for non-negative matrix factorization-based multipitch analysis. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 133–136. IEEE, 2012.

[15] Martin Piszczalski and Bernard A. Galler. Automatic music transcription. *Computer Music Journal*, 1(4):24–31, 1977.

[16] Haruto Takeda, Naoki Saito, Tomoshi Otsuki, Mitsuru Nakai, Hiroshi Shimodaira, and Shigeki Sagayama. Hidden markov model for automatic transcription of midi signals. In *Multimedia Signal Processing, 2002 IEEE Workshop on*, pages 428–431. IEEE, 2002.

[17] David Temperley. *Music and probability*. The MIT Press, 2007.

[18] David Temperley. A unified probabilistic model for polyphonic music analysis. *Journal of New Music Research*, 38(1):3–18, 2009.

WIMIR: AN INFORMETRIC STUDY ON WOMEN AUTHORS IN ISMIR

Xiao Hu[1] **Kahyun Choi**[2] **Jin Ha Lee**[3] **Audrey Laplante**[4] **Yun Hao**[2]
Sally Jo Cunningham[5] **J. Stephen Downie**[2]

[1]University of Hong Kong
xiaoxhu@hku.hk

[2]University of Illinois
{ckahyu2,yunhao2,
jdownie}@illinois.edu

[3]Univeristy of Washington
jinhalee@uw.edu

[4] Université de Montréal
audrey.laplante@umontreal.ca

[5]University of Waikato
sallyjo@waikato.ac.nz

ABSTRACT

The Music Information Retrieval (MIR) community is becoming increasingly aware of a gender imbalance evident in ISMIR participation and publication. This paper reports upon a comprehensive informetric study of the publication, authorship and citation characteristics of female researchers in the context of the ISMIR conferences. All 1,610 papers in the ISMIR proceedings written by 1,910 unique authors from 2000 to 2015 were collected and analyzed. Only 14.1% of all papers were led by female researchers. Temporal analysis shows that the percentage of lead female authors has not improved over the years, but more papers have appeared with female co-authors in very recent years. Topics and citation numbers are also analyzed and compared between female and male authors to identify research emphasis and to measure impact. The results show that the most prolific authors of both genders published similar numbers of ISMIR papers and the citation counts of lead authors in both genders had no significant difference. We also analyzed the collaboration patterns to discover whether gender is related to the number of collaborators. Implications of these findings are discussed and suggestions are proposed on how to continue encouraging and supporting female participation in the MIR field.

1. INTRODUCTION

Music Information Retrieval (MIR) is a highly interdisciplinary field with researchers from multiple domains including Electrical Engineering, Computer Science, Information Science, Musicology and Music Theory, Psychology, and so on. Perhaps due to the strong representation from the technical domains, the MIR field has been dominated by male researchers [12]. Responding to this trend, supporting female researchers has emerged as an important agenda for the MIR community. Since 2011, "Women in MIR" (WiMIR) sessions have been organized in order to identify current issues and challenges female MIR researchers face, and to brainstorm ideas for providing more support to female MIR researchers. The sessions have been well attended by both female and male participants every year, and a number of initiatives have been started for ensuring the inclusion of female researchers in various leadership roles such as session chairs, conference and program chairs, reviewers and meta-reviewers, as well as ISMIR board members. In addition, a mentorship program targeted for junior female mentees has recently been established.[1]

While we continue encouraging young female students to enter the field, we lack a solid understanding of where our current female researchers come from, what their research strengths are, who they collaborate with and what their impact has been in the field. This makes it difficult to establish a mentoring relationship between these young researchers and established scholars, which has been identified as being critical for increasing the representation of female scholars and retaining them in the field. As an effort to provide useful empirical data to support such initiatives, this paper reports an informetric study analyzing the publication, authorship and citation patterns of female researchers in the context of the ISMIR conferences.

2. RELATED WORK

2.1 Informetric Studies in MIR

A few studies in MIR have used citation analysis (examining publication and citation counts, and co-citation patterns) and co-authorship analysis to measure the impact of individual papers or authors and understand the patterns of publication. Lee, Jones, and Downie [12] conducted a citation analysis of ISMIR proceedings from 2000 to 2008, aiming to discover how the publication patterns have changed over time. They were able to identify the top 22 authors with the largest number of distinct co-authors, distinguish the commonly used title terms reflecting the research foci in the ISMIR community, and reveal the increasing co-authorship among the MIR scholars. Lee and Cunningham [13] specifically examined 198 user studies in MIR and analyzed the overall growth, publication and citation patterns, popular topics, and methods employed. They found that overall the number of user studies increased, but not the ratio of user studies published in ISMIR proceedings over time. Additionally, they were able to identify a few strong networks

© Xiao Hu, Kahyun Choi, Jin Ha Lee, Audrey Laplante, Yun Hao, Sally Jo Cunningham and J. Stephen Downie. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Xiao Hu, Kahyun Choi, Jin Ha Lee, Audrey Laplante, Yun Hao, Sally Jo Cunningham and J. Stephen Downie. "WiMIR: An Informetric Study on Women Authors in ISMIR", 17th International Society for Music Information Retrieval Conference, 2016.

[1] http://www.ismir.net/wimir.html

of co-authorship based on universities and labs, and also found that many of the studies heavily focused on experiment and usability testing. Another study by these authors [4] applied informetric methods to investigate the influence of ISMIR and MIREX research on patents, through citation and topic analysis. The results showed evidence of strong links between academic and commercial MIR research. Very recently, Sordo et al. [18] analyzed the evolution of topics and co-authorship networks in the ISMIR conference, and found larger groups with more variability of topics made more impact to the field.

Notwithstanding the significance of these studies in measuring the status and impact of the field, there has not been any study focusing on the gender disparities in MIR research. The role of gender in scholarly research and academic career has been a long standing topic in many fields, as briefly summarized in the next subsection.

2.2 Female Authors and Scholarly Research

Although there is abundant research showing that female researchers are as devoted as male researchers in the goal of discovery [17][19], women researchers are underrepresented in almost all disciplines, especially in science, technology, engineering, and mathematics (STEM) [15]. In a recent study, Sugimoto et al. [11] analyzed 5,483,841 articles published over the period 2008-2012, from the Web of Science database with over 27 million authorships. They found that only 30% of the authors were female, but surprisingly female authors dominated in some countries, such as Latvia and Ukraine. Kosmulski [10] studied publication patterns of scholars in Poland and found that female scientists in Poland published less than their male counterparts. However, an examination of yearly statistics reveals a trend moving towards gender equalization in recent years. Another study by Aksnes et al. [1] analyzed the publications of 8,500 Norwegian researchers from all disciplines. Findings showed that female researchers published significantly fewer papers than their male counterparts, but the difference in citation rate was not as salient. They also found that among the most productive researchers, women perform as well as men do. Female researchers were even found to be more highly cited than male researchers in physical sciences, including computer science, informatics, and engineering. Conversely, a study of gender-based citation patterns in the field of International Relations [14] found that women were cited significantly less than men, even after controlling for variables including tenure status, institution, and year of publication. This discrepancy was identified as partly due to gender-based self-citation patterns (where men tend to self-cite more than women) and to a tendency for men to cite other men proportionately more than women—perhaps indicating that social networks can have an impact on citation practices.

3. DATA COLLECTION AND PREPROCESSING

Two sources were used to collect the titles of ISMIR papers and their authors: the ISMIR online proceedings and the ISMIR conference web pages. First, bibliographic records of papers published between 2000 and 2011 were downloaded from the Cumulative ISMIR Proceedings database[1], which supports export in CSV format. Records for papers published between 2012 and 2015 were collected by crawling the program webpages of the conferences since they were only included in dblp[2], which does not provide a function for exporting multiple records. The crawled raw HTML pages were then parsed with regular expressions, to extract titles and author names.

3.1 Standardization and Deduplication of Names

The downloaded author names needed to be standardized in several aspects. First, some names were inverted with the last name first. Second, some authors varied the form of their name across multiple papers (e.g., including or omitting middle name initials). Third, diacritic letters in names were occasionally replaced by English letters. Besides manual inspection of these cases, we also made use of OpenRefine[3], a tool for data cleansing and exploration, to help identify similar forms of names. Once different forms of a same name were identified, we kept the most frequently used version and removed others as duplicates.

3.2 Author Gender Identification

We manually determined and labelled the gender of each author based on their names. Some first names are exclusively or almost exclusively used for one gender (e.g., *Susan*, *Marie*, and *Yumi* are female names by convention). Some names are almost exclusively attributed to males in one language but to females in another language (e.g., *Rene* is a male name in French but a female name in English). In these cases, we tried to determine the gender of authors, taking into account their cultural origin. However, many first names are androgynous, especially Chinese names whose English written forms represent the pronunciations rather than the meanings, which makes determining the gender of those names difficult. To address that, we relied on our collective knowledge of ISMIR authors and we used the affiliation information to search for these authors on the Web. Nevertheless, this did not allow us to assign genders to all authors. The last step was to send a call through the ISMIR mailing list to ask the community to help determine the gender of the authors we could not identify. We also directly contacted a few authors and labs when possible. In the end, we were able to determine the gender of 1,863 (97.54%) out of a total of the 1,910 unique authors on the list.

3.3 Citation Counts

Google Scholar (GS)[4] was used as the source of citation data for this study, since ISMIR proceedings are not indexed in Web of Science (WoS) or Scopus, the two other main sources of citation data for scholarly works. Studies have shown that GS coverage had grown substantially since its launch [21] and now even surpasses WoS coverage in certain disciplines, including Computer Science

[1] http://www.ismir.net/proceedings/
[2] http://dblp.uni-trier.de/db/conf/ismir/index.html
[3] http://openrefine.org
[4] https://scholar.google.com/

[7]. It also indexes a wider variety of academic sources, including more books, conference papers, and working papers than WoS [8]. As a result, GS has been considered as a reliable source of citation data and an adequate alternative to WoS for research evaluation [7][8]. Since GS does not offer a function for exporting multiple records at once, we used Publish or Perish[1], an open source software tool, to retrieve the citation data for ISMIR papers.

3.4 Limitations

As mentioned previously, we relied on name convention to determine the gender of a large proportion of the authors. It is possible that some of the authors from one gender had a first name more traditionally attributed to the other gender and have thus been mislabelled. Moreover, a high proportion of the 2.56% (48) of authors whose gender could not be determined are of Chinese origin. Therefore, Chinese authors are underrepresented in our dataset. Moreover, our work only focuses on two genders (i.e., male and female) based on name convention and no other gender identity. It is possible that some of these authors identify as neither male nor female and we are not able to represent that information in our analysis.

The use of GS brings additional limitations. Although GS is considered by researchers an adequate source of citation data for research evaluation, it still has some weaknesses. Research shows that the database contains many errors such as duplicates and false positive citations [21] which can potentially inflate the number of citations, but we have no reason to believe that this would affect male- and female-led papers differently. Finally, ISMIR workshops were not consistently indexed in GS, and thus we had no citation data for 35 papers.

4. RESULTS

4.1 Number of Authors and Publications

There are 1,910 unique authors who published at least one paper in ISMIR proceedings from 2000 to 2015. The gender information of 1,863 (97.54%) authors was identified. Among the identified authors, 274 (14.71%) were female and 1,589 (85.29%) were male. There were 1,610 papers published over the years. Among them, 389 papers (24.2%) had female co-authors, 227 (14.1%) had female first authors, compared to 1,188 (73.8%) papers without any female authors and 1,362 (84.6%) led by male authors. While the number of female authors did increase over time, the total number of ISMIR papers and male authors also significantly increased [12]. Figure 1 shows the percentage of papers with male and female first authors over the years as well as those with and without female authors. There is virtually no improvement over the years in terms of the proportion of papers led by female authors, but more papers with female co-authors appeared in recent years (2014 and 2015).

Figure 2 compares the number of papers led by female versus male researchers in histograms. The most proliferate female and male researchers had led almost equal number of papers, with 13 papers by Jin Ha Lee (female) and 12 by Xiao Hu (female) compared to 14 by Meinard Müller (male). This demonstrates a similar pattern to the finding in [1] that the most productive women and men researchers perform equally well.

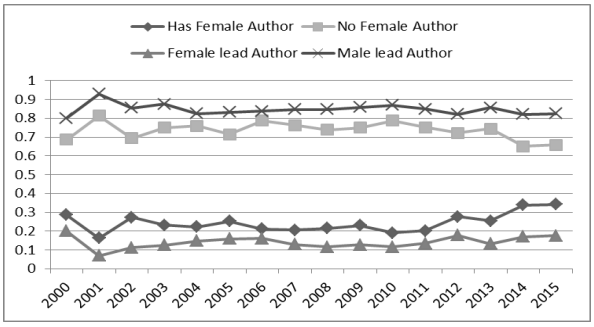


Figure 1. Proportion of ISMIR papers by each gender.

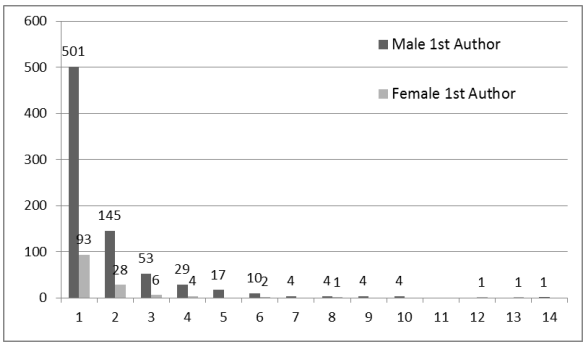


Figure 2. Number of ISMIR papers led by each gender.

4.2 Institutions and Disciplines of Female Authors

The 227 papers with female first authors (including single authored papers) were analyzed to identify the institutions and disciplines of the first authors at the time of publication. Table 1 shows the institutions with the largest number of such papers. The ranks of the top three institutions were in fact earned by their female students, as no female researchers with permanent positions in these institutions has led an ISMIR paper. This is evidence of a strong contribution that female students made to the field, supporting the importance of fostering the growth of junior female researchers through mentorship programs.

Institutions	Number of papers
University of Illinois	12
Queen Mary University of London	10
McGill University	9
University of Washington	9
Indiana University	8
University of Waikato	8
University of Southern California	7
Fraunhofer IDMT	6
Goldsmiths, University of London	5
Pompeu Fabra University	5
Stanford University	5
Utrecht University	5

Table 1. Institutions with the most papers of first female authors.

[1] http://www.harzing.com/resources/publish-or-perish

Table 2 lists the most frequent disciplines of female authors who led ISMIR papers. The discipline information was obtained from the departments of the female first authors' affiliations as written in the papers. The disciplines were cleaned up such that closely related ones were combined. For example, "Computer Science and Informatics" was combined with "Computer Science", and "Audio and Speech Processing" was combined with "Electronical Engineering". The most popular discipline is Computer Science, following by Library and Information Science and Music Technology. The latter two were interdisciplinary fields which historically had stronger female representations [16]. When looking at Tables 1 and 2 together, papers from some top ranked institutions were contributed from authors in Library and Information Science (University of Illinois, University of Washington) or Music Technology (Queen Mary University of London, McGill University), rather than the Engineering disciplines predominant in the field. The results indicate that it can be promising to try to foster more female contributors to ISMIR from these disciplines.

Discipline	Number of papers
Computer Science	87
Library and Information Science	44
Music Technology	40
Electrical Engineering	18
Musicology and Music Theory	12

Table 2. Top disciplines of first female authors.

From the affiliations of the female authors, we identified the geographic locations of the authors, as shown in Table 3. Unsurprisingly, most of them were in North America and Europe, in accordance with the fact that most labs in the MIR field are located in these areas. These are followed by Asia and Pacific region with 39 papers led by female authors. Promoting international collaborations between regions with more established and reputable research facilities and other emerging but less developed regions can be a fruitful approach for fostering female researchers in the field. Sugimoto et al. [11] also advocated international collaboration as "it might help to level the playing field" (p.213). This observation may also be related to a study by Ferreira [6] which reported that a steady growth of PhD dissertations written by female in the U.S. was observed, but the increase was attributed to international female research students who came from other parts of the world including Asia. Although further studies are warranted to verify whether this trend also holds for ISMIR authors, in our dataset we did observe circumstantial evidence in that many female author names with Asian origins were based at institutions in Europe or North America.

4.3 Co-authorship

Among all the papers led by female and male authors, the average number of co-authors is 2.69 and 2.86, respectively. A two-sample unequal variance t-test reported a non-significant difference between the two (p = 0.289). Figure 3 illustrates the co-authorship trend over the years. In general, papers led by authors in either gender tend to have an increasing number of co-authors.

Continent	Number of papers
North America	96
Europe	90
Asia	28
Oceania	11
South America	2
Total	227

Table 3. Continents of female leading authors.

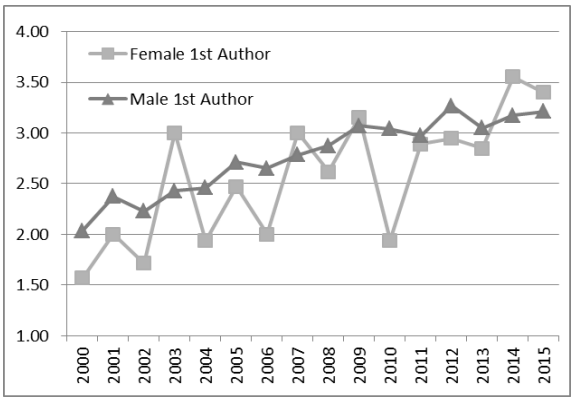


Figure 3. Number of co-authors per paper (2000-2015).

There were 214 single authored papers: 35 (16.3%) of them written by female authors and 179 (83.7%) by males. This percentage of 16.3% is lower than what was reported in [20] in which they found that 26% of single-authored papers published in the JSTOR network databases since 1990 were contributed by female authors. In our dataset, 22 (8.0%) of female authors had single-authored one or more ISMIR papers, whereas 129 (8.1%) of male authors had done so. The results indicate that both female and male authors reach out for collaborations, perhaps due to the interdisciplinary nature of the MIR field. In addition, similar percentages of female and male authors opted to write single authored papers.

We also conducted social network analysis (SNA) on the co-authorship networks of female researchers and their collaborators, to find out with which authors the female researchers most frequently collaborated. Figure 4 shows the network graphs (generated by using the NodeXL SNA tool [8]). The graphs' nodes represent researchers who were grouped into clusters by using the Clauset-Newman-Moore clustering algorithm [3], such that the authors who often collaborated with each other were grouped into a single cluster. The size of a node is proportional to the number of papers written by the researcher. Figure 4 contains nine clusters, each of which has at least five female authors. Each female author in the graphs is represented by a node of diamond shape and the name is marked with an asterisk. In each graph, the names of authors with the most co-authors are labelled.

As shown in Figure 4, some clusters contain multiple female authors with a relatively high number of publications, such as the one with Jin Ha Lee, Xiao Hu and Sally Jo Cunningham, as well as the one with Rebecca

Fiebrink, Catherine Lai, etc. This can probably be attributed to the research groups these authors were affiliated with, as this result corresponds to the pattern observed in Table 1: the two clusters match to the research groups in University of Illinois and McGill University, respectively. Other clusters shown in this figure also reflect research groups such as the Music Technology Group in Pompeu Fabra University (the cluster with Emilia Gomez) and Utrecht University (the cluster with Anja Volk). This pattern once again verifies the importance of having research labs or groups that can foster the growth of female researchers.

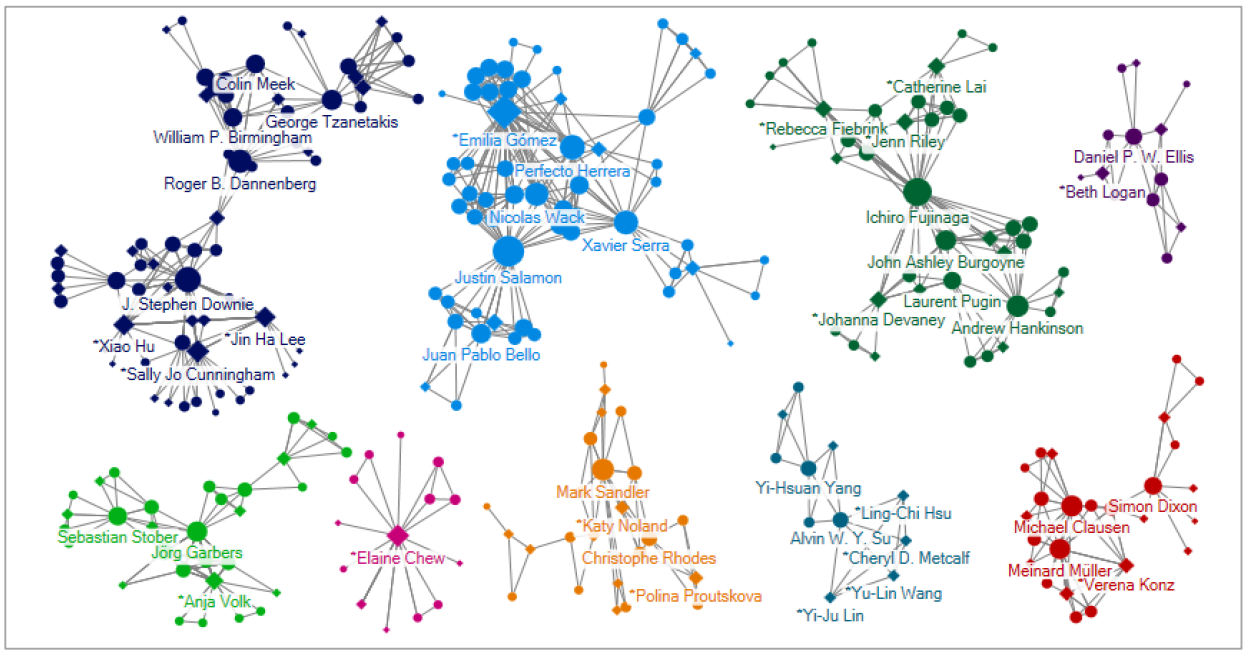


Figure 4. Co-authorship networks of ISMIR female authors (groups with at least five female authors are presented)

4.4 Citation analysis

The average citation count of all papers with female researcher as the leading authors is 34.30. Although this number is lower than that of papers with male leading authors (43.26), the difference was not significant (p = 0.259). When considering citation counts of single-authored papers, the difference is even smaller: 22.25 versus 25.27 for female and male authors, respectively.

Figure 5 shows the comparative distribution of papers led by female and male authors by number of citations. A chi-square independence test shows that the two distributions are very similar (χ^2 = 11.124, df = 8, p = 0.195). The proportion of papers with no citation is the same for both groups (9%). The proportion of highly cited papers (papers cited more than 100 times) is also very similar, representing 10% of female first-authored papers and 9% of male first-authored papers.

These results indicate that the scholarly impact of authors in both genders is similar. Although previous studies found that the difference between citation rates for male- and female-led papers was smaller than that between publication rates [1], it is unusual to see no significant difference on citation rate between genders. The large scale study by Sugimoto et al. [11] found there were fewer citations for papers with female being sole author, first author or last author than in cases where a man was in one of these roles. As Sugimoto and her colleagues worked with more than 5 million of papers across all disciplines, it is an encouraging result that such gender disparity in scholarly impact as measured by citation counts is not significant in MIR.

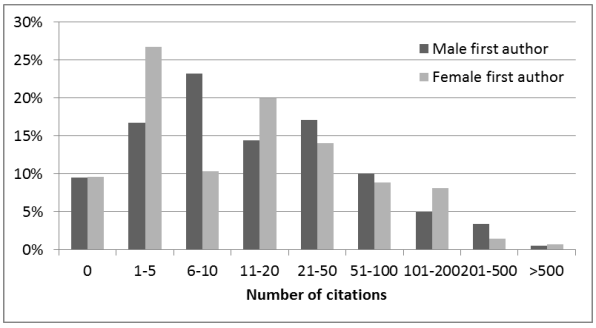


Figure 5. Distribution of female and male-led ISMIR papers by number of citations.

4.5 Topics

Topic analysis was conducted with the titles of the papers, to identify the topics female authors tended to pursue. Both single terms (unigrams) in the titles and combinations of two consecutive terms (bigrams) were extracted. To combine different forms of the same word prefix, the Porter stemmer was used. Stop words were also eliminated, as was the word "music" as it is related to all papers in ISMIR. The most frequently used title words (unigrams) are presented in Table 4. For comparison purposes, the table includes the top title words for six paper

categories: all papers, papers with female lead authors, papers with at least one female author (but not the lead), papers with no female authors, papers with male lead authors, as well as papers written by teams of all female authors. As such, there are overlapping papers between the "Female 1st" and "All Female" categories, and between the "Female non-1st" and "Male 1st" categories.

The first five columns in Table 4 show similar words such as "audio", "retriev" (retrieval"), "classif" (classify). One exception is the female non-1st author category that contains terms such as "detect", "evalu" (evaluation), and "record", which suggests that female researchers collaborated with male to work on key detection and evaluation. Several terms from all female teams are quite different from those in other categories (e.g., "digit", "user", "kei" (key) and "evalu"), which suggests that areas in which female authors worked together include user studies, key detection and evaluation.

As single words may bear limited semantics, we also extracted bigrams from the paper titles of the aforementioned six categories of gender authorship. Bigrams are two consecutive words which are often phrases, and thus may provide more meanings than unigrams. As listed in Table 5, the differences of bigrams across paper categories are even more obvious than those of unigrams. Papers with all female authors were most likely to be on audio key finding, digital libraries, melody extraction, and user studies. Papers led by female authors were more likely to focus on melody similarity, mood classification, retrieval systems, corpus and data sources, as well as cross-cultural issues. In contrast, male researchers were more likely to write about Markov model, audio signals, audio features, and Web-based approaches. These differences in focus may reflect the distribution of representation of women in Computer Science and Engineering, where proportionately more women in those fields focus on Human-Computer Interaction (e.g., user studies, cross-cultural issues, digital libraries) rather than signal processing (e.g., audio signals, audio features) [2].

All	Female 1st	Female non-1st	All male	Male 1st	All Female
audio	audio	retriev	audio	audio	inform
retriev	retriev	audio	retriev	retriev	retriev
inform	inform	classif	model	similar	**digit**
automat	classif	inform	featur	featur	similar
classif	similar	analysi	similar	classif	**user**
similar	model	**detect**	analysi	automat	audio
featur	automat	**evalu**	automat	inform	**kei**
analysi	polyphon	**record**	inform	analysi	**evalu**
recognit	**song**	system	classif	system	extract
polyphon	featur	feature	system	recognit	**find**

Table 4. Most frequent words in paper titles (terms unique to female authors are bolded).

All	Female 1st	Female non-1st	All male	Male 1st	All Female
content-bas	inform_retriev	inform_retriev	content-bas	content-bas	inform_retriev
polyphonic_audio	**genr_classif**	polyphon_audio	audio_signal	non-negativ	**audio_kei**
real-tim	**melod_similar**	**genr_classif**	markov_model	polyphon_audio	**digit_librari**
non-negativ	**classif_us**	audio_record	web-bas	audio_signal	**kei_find**
markov_model	content-bas	**auditori_model**	audio_feature	markov_model	**melodi_extract**
audio_feature	**mood_classif**	**base_transcrib**	audio_us	audio_feature	**understand_user**
audio_signal	**retriev_system**	**classif_us**	audio-bas	web-bas	
audio_fingerprint	**comput_model**	**corpus-bas**	polyphonic_audio	audio_record	
audio_record	**cross-cultur**	**data_sourc**	audio_record	audio_us	
automati_chord	**machin_learn**	**digit_imag**	score_inform	audio-bas	

Table 5. Most frequent bigrams in titles of papers (terms unique to female authors are bolded).

5. CONCLUSION AND FUTURE WORK

Overall our findings show both positive and negative aspects related to gender balance issues in MIR. While it is discouraging that the participation of female authors has hovered around 10-20% throughout the history of ISMIR without much improvement over time, we also see that the most prolific authors of both genders are similarly productive and papers led by both genders are cited at similar rates. Our analysis highlights the importance of the role of mentorship through co-authoring papers and also being part of the same labs or research groups for increasing the number of female scholars in the field. International collaborations connecting female researchers in less represented regions with more established groups can be a promising approach. In addition, we may encourage and attract female contributors from interdisciplinary disciplines historically with better female representations such as Information Science and Music Technology. Promoting research in these areas (whether by male or female authors) has also been identified as an important step forward for the field of MIR [5][12]—it is crucial to the development of usable, effective music systems that we understand our users and their needs, and work to create new systems that integrate with both technological and social infrastructures.

In order to conduct a more accurate informetric study in the future, it would be useful for the ISMIR program committee to collect gender information during the paper submission process directly from the authors. This will not only allow us to obtain a more accurate representation of the ISMIR community, but also enable the analysis on paper rejection rates in terms of gender. We also recommend the gathering of gender and research focus data for program committee members, to examine the possible effect of gender in the gatekeeping aspect of entry to the ISMIR community.

6. REFERENCES

[1] D. W. Aksnes, K. Rorstad, F. Piro, and G. Sivertsen: "Are female researchers less cited? A large - scale study of Norwegian scientists," *Journal of the American Society for Information Science and Technology*, Vol. 62, No. 4, pp. 628–636, 2011.

[2] J. M. Cavero, B. Vela, P. Cáceres, C. Cuesta, and A. Sierra-Alonso: "The evolution of female authorship in computing research," *Scientometrics*, *103*(1), 85-100, 2015.

[3] A. Clauset, M. E. J. Newman, and C. Moore: "Finding community structure in very large networks," *Physical Review E* 70.6 (2004): 066111.

[4] S. J. Cunningham and J. H. Lee: "Influences of ISMIR and MIREX research on technology patents." *ISMIR*, pp. 137-142. 2013.

[5] J. S. Downie, X. Hu, J. H. Lee, K. Choi, S. J. Cunningham, and Y. Hao: "Ten years of MIREX: Reflections, challenges and opportunities." *ISMIR*, pp. 657 – 662, 2014.

[6] M. M. Ferreira: "Trends in women's representation in science and engineering." *Journal of Women and Minorities in Science and Engineering*, Vol. 15, No. 3, 2009.

[7] M. Franceschet: "A comparison of bibliometric indicators for computer science scholars and journals on Web of Science and Google Scholar," *Scientometrics*, vol. 83, no. 1, pp. 243-258, 2009.

[8] D. L. Hansen, B. Schneiderman, and M. A. Smith: *Analyzing Social Media Networks with NodeXL: Insights from A Connected World*, Burlington, MA: Morgan Kaufmann, 2011.

[9] A.-W. Harzing: "A preliminary test of Google Scholar as a source for citation data: a longitudinal study of Nobel prize winners," Scientometrics, vol. 94, no. 3, pp. 1057-1075, 2012.

[10] K. Kosmulski: "Gender disparity in Polish science by year (1975–2014) and by discipline." *Journal of Informetrics*, Vol. 9, No. 3, pp. 658–666, 2015.

[11] V. Larivière, C. Q. Ni, Y. Gingras, B. Cronin and C. R. Sugimoto: "Global gender disparities in science." *Natur*e, Vol. 504, No. 7479, pp. 211–213, 2013.

[12] J. H. Lee, M. C. Jones and J. S. Downie: "An Analysis of ISMIR Proceedings: Patterns of Authorship, Topic, and Citation." *ISMIR*, pp. 57–62, 2009.

[13] J. H. Lee and S. J. Cunningham: "Toward an understanding of the history and impact of user studies in music information retrieval." *Journal of Intelligent Information Systems*, Vol 41, No. 3, pp. 499–511, 2013.

[14] D. Maliniak, R. Powers, and B. F. Walter: "The gender citation gap in international relations," *International Organization*, *67*(04), 889-922, 2013.

[15] National Center for Education Statistics: *Postsecondary Institutions in the United States: Fall 2000 and degrees and other awards conferred 1999–2000*, Washington, DC, National Center for Education Statistics, 2001.

[16] D, Rhoten. and S, Pfirman.: "Women in interdisciplinary science: Exploring preferences and consequences." 2010 *Research policy*, Vol. 36, No. 1, pp. 56–75, 2007.

[17] G. Sonnert and G. Holton: *Gender Differences in Science Careers: The Project Access Study.* Rutgers University Press, New Brunswick, NJ, 1995.

[18] M. Sordo, M. Ogihara, and S. Wuchty: "Analysis of the evolution of research groups and topics in the ISMIR conference." *ISMIR*, pp. 204 – 210, 2015.

[19] R. Subotnik and K. Arnold: "Passing through the gates: career establishment of talented women scientists," *Roeper Review*, Vol. 18, No. 1, pp. 55–61, 1995.

[20] J. D. West, J. Jacquet, M. M. King, S. J. Correll and C. T. Bergstrom: "The role of gender in scholarly authorship," *PloS one*, Vol. 8, No. 7, e66212, 2013.

[21] J. C. F. Winter, A. A. Zadpoor, and D. Dodou: "The expansion of Google Scholar versus Web of Science: a longitudinal study," *Scientometrics*, vol. 98, no. 2, pp. 1547-1565, 2013.

Oral Session 6

Symbolic

AUTOMATIC MELODIC REDUCTION USING A SUPERVISED PROBABILISTIC CONTEXT-FREE GRAMMAR

Ryan Groves
groves.ryan@gmail.com

ABSTRACT

This research explores a Natural Language Processing technique utilized for the automatic reduction of melodies: the Probabilistic Context-Free Grammar (PCFG). Automatic melodic reduction was previously explored by means of a probabilistic grammar [11] [1]. However, each of these methods used unsupervised learning to estimate the probabilities for the grammar rules, and thus a corpus-based evaluation was not performed. A dataset of analyses using the Generative Theory of Tonal Music (GTTM) exists [13], which contains 300 Western tonal melodies and their corresponding melodic reductions in tree format. In this work, supervised learning is used to train a PCFG for the task of melodic reduction, using the tree analyses provided by the GTTM dataset. The resulting model is evaluated on its ability to create accurate reduction trees, based on a node-by-node comparison with ground-truth trees. Multiple data representations are explored, and example output reductions are shown. Motivations for performing melodic reduction include melodic identification and similarity, efficient storage of melodies, automatic composition, variation matching, and automatic harmonic analysis.

1. INTRODUCTION

Melodic reduction is the process of finding the more structural notes in a melody. Through this process, notes that are deemed less structurally important are systematically removed from the melody. The reasons for removing a particular note are, among others, pitch placement, metrical strength, and relationship to the underlying harmony. Because of its complexity, formal theories on melodic reduction that comprehensively define each step required to reduce a piece in its entirety are relatively few.

Composers have long used the rules of ornamentation to elaborate certain notes. In the early 1900s, the music theorist Heinrich Schenker developed a hierarchical theory of music reduction (a comprehensive list of Schenker's publications was assembled by David Beach [7]). Schenker ascribed each note in the musical surface as an elaboration of a representative musical object found in the deeper levels of reduction. The particular categories of ornamentation that were used in his reductive analysis were *neighbor tones*, *passing tones*, *repetitions*, *consonant skips*, and *arpeggiations*. Given a sequence of notes that can be identified as a particular ornamentation, an analyst can remove certain notes in that sequence so that only the more important notes remain.

In the 1980s, another theory of musical reduction was detailed in the GTTM [16]. The authors' goal was to create a formally-defined generative grammar for reducing a musical piece. In GTTM, every musical object in a piece is *subsumed* by another musical object, which means that the subsumed musical object is directly subordinate to the other. This differs from Schenkerian analysis, in that every event is related to another single musical event. In detailing this process, Lerdahl and Jackendoff begin by breaking down metrical hierarchy, then move on to identifying a grouping hierarchy (separate from the metrical hierarchy). Finally, they create two forms of musical reductions using the information from the metrical and grouping hierarchies—the time-span reduction, and the prolongational reduction. The former details the large-scale grouping of a piece, while the latter notates the ebb and flow of musical tension in a piece.

Many researchers have taken the idea—inspired by GTTM or otherwise—of utilizing formal grammars as a technique for reducing or even generating music (see Section 2.0.0.0.2). However, most of these approaches were not data-driven, and those that were data-driven often utilized unsupervised learning rather than supervised learning. A dataset for the music-theoretical analysis of melodies using GTTM has been created in the pursuit of implementing GTTM as a software system [13]. This dataset contains 300 Western classical melodies with their corresponding reductions, as notated by music theorists educated in the principles of GTTM. Each analysis is notated using tree structures, which are directly compatible with computational grammars, and their corresponding parse trees. The GTTM dataset is the corpus used for the supervised PCFG detailed in this paper.

This work was inspired by previous research on a PCFG for melodic reduction [11], in which a grammar was designed by hand to reflect the common melodic movements found in Western classical music, based on the compositional rules of ornamentation. Using that hand-made grammar, the researchers used a dataset of melodies to calculate the probabilities of the PCFG using unsupervised learning. This research aims to simulate and perform the pro-

© Ryan Groves. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Ryan Groves. "Automatic Melodic Reduction Using a Supervised Probabilistic Context-Free Grammar", 17th International Society for Music Information Retrieval Conference, 2016.

cess of melodic reduction, using a supervised Probabilisitic Context-Free Grammar (PCFG). By utilizing a ground-truth dataset, it is possible to directly induce a grammar from the solution trees, creating the set of production rules for the grammar and modelling the probabilities for each rule expansion. In fact, this is the first research of its type that seeks to directly induce a grammar for the purpose of melodic reduction. Different data representations will be explored and evaluated based on the accuracy of their resulting parse trees. A standard metric for tree comparison is used, and example melodic reductions will be displayed.

The structure of this paper is as follows: The next section provides a brief history of implementations of GTTM, as well as an overview of formal grammars used for musical purposes. Section 3 presents the theoretical foundations of inducing a probabilistic grammar. Section 4 describes the data set that will be used, giving a more detailed description of the data structure available, and the different types of melodic reductions that were notated. Section 5 describes the framework built for converting the input data type to an equivalent type that is compatible with a PCFG, and also details the different data representations used. Section 6 presents the experiment, including the comparison and evaluation method, and the results of the different tests performed. Section 7 provides some closing remarks.

2. LITERATURE REVIEW

In order to reduce a melody, a hierarchy of musical events must be established in which more important events are at a higher level in the hierarchy. Methods that create such a structure can be considered to be in the same space as melodic reduction, although some of these methods may apply to polyphonic music as well. The current section details research regarding hierarchical models for symbolic musical analysis.

2.1 Implementing GTTM

While much research has been inspired by GTTM, some research has been done to implement GTTM directly. Fred Lerdahl built upon his own work by implementing a system for assisted composition [17]. Hamanaka et al. [13] presented a system for implementing GTTM. The framework identifies time-span trees automatically from monophonic melodic input, and attained an f-measure of 0.60. Frankland and Cohen isolated the grouping structure theory in GTTM, and tested against the task of melodic segmentation [10].

2.2 Grammars in Music

In 1979, utilizing grammars for music was already of much interest, such that a survey of the different approaches was in order [20]. Ruwet [21] suggested that a generative grammar would be an excellent model for the creation of a top-down theory of music. Smoliar [22] attempted to decompose musical structure (including melodies) from audio signals with a grammar-based system.

Baroni et al. [4] also created a grammatical system for analyzing and generating melodies in the style of Lutheran chorales and French chansons. The computer program would create a completed, embellished melody from an input that consisted of a so-called "primitive phrase" (Baroni et al. 1982, 208).

Baroni and Jacoboni designed a grammar to analyze and generate melodies in the style of major-mode chorales by Bach [5, 6]. The output of the system would generate the soprano part of the first two phrases of the chorale.

2.3 Probabilistic Grammars

Gilbert and Conklin [11] designed a PCFG for melodic reduction and utilized unsupervised learning on 185 of Bach's chorales from the Essen Folksong Collection. This grammar was also explored by Abdallah and Gold [1], who implemented a system in the logical probabilistic framework PRISM for the comparison of probabilistic systems applied to automatic melodic analysis. The authors implemented the melodic reduction grammar provided by Gilbert and Conklin using two separate parameterizations and compared the results against four different variations of Markov models. The evaluation method was based on data compression, given in bits per note (bpn). The authors found that the grammar designed by Gilbert and Conklin was the best performer with 2.68 bpn over all the datasets, but one of the Markov model methods had a very similar performance. The same authors also collaborated with Marsden [2] to detail an overview of probabilistic systems used for the analysis of symbolic music, including melodies.

Hamanaka et al. also used a PCFG for melodic reduction [12]. The authors used the dataset of treebanks that they had previously created [13] to run supervised learning on a custom-made grammar that he designed, in order to automatically generate time-span reduction trees. This work is very similar to the work presented here, with two exceptions. First, the grammar was not learned from the data. Secondly, Hamanaka used a series of processes on the test melodies using previous systems he had built. These systems notated the metrical and grouping structure of the input melody, before inputting that data into the PCFG. Hamanaka achieves a performance of 76% tree accuracy.

2.4 Similar Methods for Musical Reduction

Creating a system that can perform a musical reduction according to the theory of Heinrich Schenker has also been the topic of much research. Marsden explored the use of Schenkerian reductions for identifying variations of melodies [19]. PCFGs have not yet been utilized for this particular task. One notable caveat is the probabilistic modelling of Schenkerian reductions, using a tree-based structure [15]. Kirlin did not explicitly use a PCFG, however his model was quite similar, and also was a supervised learning method.

3. SUPERVISED LEARNING OF A PCFG

To understand the theoretical framework of the PCFG, it is first useful to give a brief background of formal grammars. Grammars were formalized by Chomsky [8] and extended by himself [9] and Backus et al. [3]. The definition of a formal grammar consists of four parameters, $G = \{N, \Sigma, R, S\}$, which are defined as follows [14]:

N	a set of non-terminal symbols
Σ	a set of terminals (disjoint from N)
R	a set of production rules, each of the form $\alpha \rightarrow \beta$
S	a designated start symbol

Each production rule has a right-hand side, β, that represents the expansion of the term found on the left-hand side, α. In a Context-Free Grammar (CFG), the left-hand side consists of a single non-terminal, and the right-hand side consists of a sequence of non-terminals and terminals. Non-terminals are variables that can be expanded (by other rules), while terminals are specific strings, representing elements that are found directly in the sequence (for example, the 'dog' terminal could be one expansion for the *Noun* non-terminal). Given a CFG and an input sequence of terminals, the CFG can *parse* the sequence, creating a hierarchical structure by iteratively finding all applicable rules. Grammars can be ambiguous; there can be multiple valid tree structures for one input sequence.

PCFGs extend the CFG by modelling the probabilities of each right-hand side expansion for every production rule. The sum of probabilities for all of the right-hand side expansions of each rule must sum to 1. Once a PCFG is calculated, it is possible to find the *most probable* parse tree, by cumulatively multiplying each production rule's probability throughout the tree, for every possible parse tree. The parse tree with the maximum probability is the most likely. This process is called disambiguation.

3.1 Inducing a PCFG

When a set of parse tree solutions (called a *treebank*) exists for a particular set of input sequences, it is possible to construct the grammar directly from the data. In this process, each parse tree from the treebank will be broken apart, so that the production rule at every branch is isolated. A grammar will be formed by accumulating every rule that is found at each branch in each tree, throughout the entire treebank. When a rule and its corresponding expansions occurs multiple times, the probabilities of the right-hand side expansion possibilities are modelled. Inducing a PCFG is a form of supervised learning.

4. GTTM DATASET

The GTTM dataset contains the hierarchical reductions (trees) of melodies in an Extensible Markup Language (XML) representation.

There are two different types of reduction trees that are created with the theories in GTTM: time-span reduction trees, and prolongational reduction trees. The time-span reduction is built upon the grouping structure analysis provided in GTTM, which in turn uses the metrical structure analysis to influence its decision-making. Time-span reduction trees are generally more reliant on the metrical information of a piece, since it utilizes the grouping structure directly. The prolongational reductions are designed to notate the ebb and flow of tension and progression in a piece. In fact, in GTTM, the prolongational reductions use time-span reduction trees as a starting point, but then build the branching system from the top, down, based on pitch and harmonic content in addition to the time-span information.

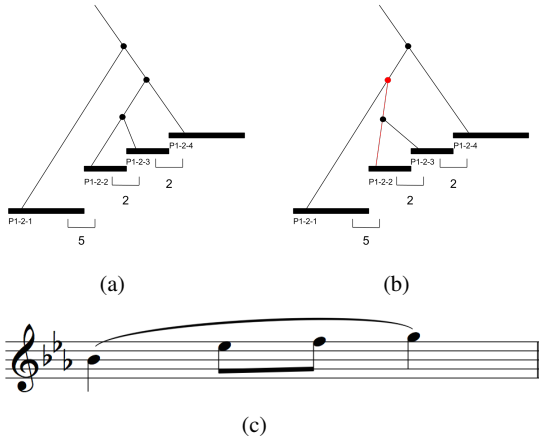


Figure 1: The prolongational tree (a) and the time-span tree (b) for the second four notes in Frédéric Chopin's "Grande Valse Brillante", as well as the score (c). The intervals between notes are notated in number of semitones.

An example helps to detail their differences. Figure 1 shows a particular phrase from one of the melodies in the GTTM dataset: Frédéric Chopin's "Grande Valse Brillante" [13]. The note labelled P1-2-2 is attached to the last note of the melody in the prolongational reduction, because of the passing tone figure in the last 3 notes, whereas the time-span tree connects note P1-2-2 to the first note of the melody, due to its metrical strength and proximity.

The entire dataset consists of 300 melodies, with analyses for each. However, the prolongational reduction trees are only provided for 100 of the 300 melodies, while the time-span trees are provided for all 300 melodies. The prolongational reductions require the annotations of the underlying harmony. Likewise, there are only 100 harmonic analyses in the dataset.

5. FORMING THE PCFG

Gilbert and Conklin decided to model the relevant characteristics of the data by hand, by manually creating grammar rules that represented the music composition rules of ornamentation [11]. The melodic embellishment rules included in their grammar were the following: passing tone, neighbor tone, repeated tone, and the escape tone. Additionally, they created a "New" rule which was a kind of catch-all for any interval sequence that could not be described by the other rules. In order for the rules to be applicable at

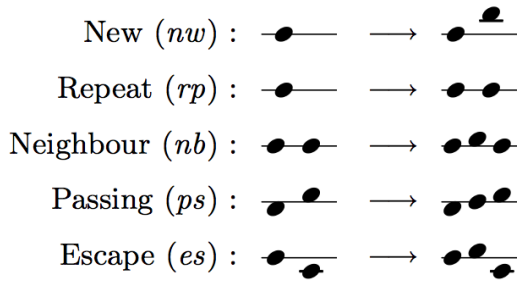


Figure 2: A visualization of a set of melodic embellishment rules, encoded manually into the production rules of a formal grammar [11, 3].

any pitch location, the fundamental unit of data was the interval between two notes, rather than two separate values for each note. The complete ruleset is shown in Figure 2.

When learning a grammar directly from a dataset of annotations, the most important decision to make is the data representation. The representation chosen should be able to capture the most relevant characteristics of the data. Similar to Gilbert and Conklin, each rule modelled two consecutive intervals in a sequence of three notes, and had the following form (notes labelled as *n1* through *n3*):

$$interval_{n1,n3} \rightarrow interval_{n1,n2} \;\; interval_{n2,n3} \quad (1)$$

The motivation was that melodic rules often involve a sequence of 3 notes. This is true for the passing tone, neighbor tone, and the escape tone. The repetition rule would normally require only two notes, however to keep a consistent format, repetitions were only reduced when three consecutive notes of the same pitch were found, which were then reduced to two notes of the same pitch (creating one interval). The "New" rule was no longer needed, since the model learns the rules directly from the training data. This form of one interval expanding into two consecutive intervals for the grammatical rules was adopted for this research.

5.1 A Framework for Converting Trees

Utilizing a representation that required a sequence of two intervals in every right-hand expansion presented a problem, because the GTTM reduction trees were in a format that associated pairs of notes at each branch intersection—not the three consecutive notes required for the two consecutive intervals. Given this challenge, a framework was developed to convert the note representation of the GTTM data into the interval notation desired, and to build the corresponding tree structure using the interval representation.

An example GTTM tree is shown in Figure 3. Note that at the end of every branch is a single note. An algorithm was developed to allow the conversion of these note-based trees to any interval representation desired, based on a sequence of 3 notes. The algorithm traverses the tree from

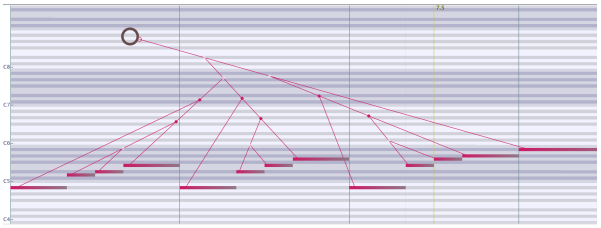


Figure 3: The prolongational reduction tree for half of the first melody in the GTTM dataset, Frédéric Chopin's "Grande Valse Brillante", as displayed in the GTTM visualizer provided by Hamanaka, Hirata, and Tojo [13].

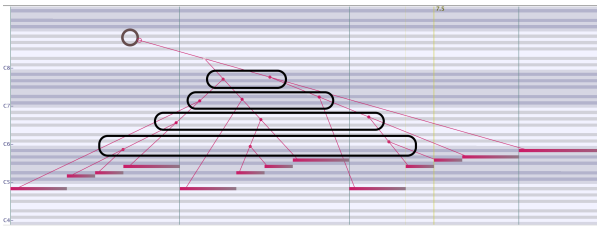


Figure 4: A depiction of the process for converting a tree that uses a note representation to a tree that uses an interval representation, by traversing the tree breadth-wise and relating sets of 3 notes.

the top, down, in a breadth-wise fashion. At each level of depth, the sequence of notes at that depth are broken into sets of 3 consecutive notes, and their intervals are computed. The framework allows for any interval-based representation to be applied. For example, it could be regular pitch intervals, inter-onset interval (IOI), difference in metric prominence, or even representations that consider the notes' relationships to scale and harmony. Figure 4 highlights the breadth-wise traversal process.

The framework was built in Python. It takes a function as input, which allows the user to define unique interval representations. When the function is called during the tree conversion process, the information available for defining the representation consists of the two notes (which contain duration, onset and pitch information), the current key, and the current underlying harmony (if available). The interval encoding that is returned by the function is then used as a node in the resulting tree.

5.2 Training/Induction

The Python-based Natural Language Toolkit (NLTK) was used for the process of PCFG induction [18]. Given a treebank of solutions, the process for inducing a PCFG is described as follows. For every tree in the treebank, traverse through the tree to identify each branching location. For every branching location, create a rule with the node label as the left-hand side, and the children as the right-hand side. Collect the set of rules found at every branch of every tree in the treebank, and pass that list of production rule instances into NLTK's induce_pcfg function. The induce_pcfg function will catalogue every rule, and build up a grammar based on those rules. It will also model the

probability of each rule's unique expansions.

5.3 Data Representations

For the representation of intervals between two consecutive notes, this research focused on a few certain musical attributes. These attributes were tested first in isolation, and then in combination. The following descriptions relate to the attributes labelled in the results table (the key for each attribute is given in parentheses following the name).

Pitch The difference in pitch between two notes was a part of every data representation tested. However, the encodings for these pitch values varied. Initially, a simple pitch-class representation was used. This allowed pitch intervals at different points in the musical scale to be grouped into the same production rules. It was assumed that direction of pitch would also be an important factor, so the **Pitch-Class (PC)** attribute allowed the following range of intervals: [-11, 11]. Melodic embellishment rules often apply to the same movements of intervals within a musical scale. For this reason, the **Key-Relative Pitch-Class (KPC)** was also used, which allowed a range of intervals from [-7, 7], measuring the distance in diatonic steps between two consecutive notes.

Metrical Onset For encoding the metrical relationships between two notes, the *metric delta* representation was borrowed from previous research [11]. This metric delta assigns every onset to a level in a metrical hierarchy. The metrical hierarchy is composed of levels of descending importance, based on their onset location within a metrical grid. The onsets were assigned a level based on their closest onset location in the metrical hierarchy. This metrical hierarchy was also used in GTTM for the metrical structure theory [16].

Because the GTTM dataset contains either 100 or 300 solutions (for prolongational reduction trees and time-span trees, respectively), the data representations had to be designed to limit the number of unique production rules created in the PCFG. With too many production rules, there is an increased chance of production rules that have a zero probability (due to the rule not existing in the training set), which results in the failure to parse certain test melodies. Therefore, two separate metrical onset attributes were created. One which represented the full metrical hierarchy, named **Metric Delta Full (Met1)**, and one which represented only the change in metric delta (whether the metric level of the subsequent note was higher, the same, or lower than the previous note), named **Metric Delta Reduced (Met0)**.

Harmonic Relationship This research was also designed to test whether or not the information of a note's relationship to the underlying harmony was useful in the melodic reduction process. A **Chord Tone Change (CT)** attribute was therefore created, which labelled whether or not each note in the interval was a chord tone. This created four possibilities: a chord tone followed by a chord tone, a chord tone followed by a non-chord tone, a non-chord tone followed by a chord tone and a non-chord tone followed by a non-chord tone. This rule was designed to test whether harmonic relationships affected the reduction process.

6. THE EXPERIMENT

Given a method for creating a treebank with any interval-based data representation from the GTTM dataset and inducing the corresponding PCFG, an experiment was designed to test the efficacy of different data representations when applied to the process of melodic reduction. This section details the experiment that was performed. First, different representations that were tested are presented. Then, the comparison and evaluation method are described. Finally, the results of cross-fold evaluation for the PCFG created with each different data representation are shown.

6.1 Comparison

The comparison method chosen was identical to the methods used in other experiments of the same type, in which the output of the system is a tree structure, and the tree solutions are available [13, 15]. First, for a given test, the input melody is parsed, which yields the most probable parse tree as an output. The output trees are then compared with the solution trees. To do so, the tree is simply traversed, and each node from the output tree is compared for equivalence to the corresponding node in the solution tree. This method is somewhat strict, in that mistakes towards the bottom of the tree will be propagated upwards, so incorrect rule applications will be counted as incorrect in multiple places.

6.2 Evaluation

Cross-fold evaluation was used to perform the evaluation. The entire treebank of solutions were first partitioned into 5 subsets, and 1 subset was used for the test set in 5 iterations of the training and comparison process. The results were then averaged. In order to keep consistency across data representations, the same test and training sets were used for each cross-validation process.

6.3 Results

Each data representation that was selected was performed on both the set of time-span reduction trees and the set of prolongational reduction trees, when possible. As mentioned previously, the set of prolongational reduction trees amounted to only 100 samples, while the time-span trees amounted to 300. In some situations, the data representation would create too many unique production rules, and not all the test melodies could be parsed. All of the data representations in the results table had at least a 90% coverage of the test melodies, meaning that at least 90% of the tests could be parsed and compared. There are also two data representations that use time-span trees with the harmonic representation. For these tests, the solution set contained only 100 samples as opposed to the usual 300 for

time-span trees, since there is only harmonic information for 100 of the 300 melodies.

Tree-type	PC	KPC	Met1	Met0	CT	% nodes correct
TS	X					35.33
PR	X					38.57
TS		X			X	40.40
PR		X			X	38.50
TS		X	X			44.12
PR		X		X		46.55
TS		X		X	X	44.80
PR		X		X	X	46.74

These results mostly progress as one might expect. Looking at only the tests done with time-span trees, the results improve initially when using the **Key-Relative Pitch-Class** encoding for pitch intervals paired with the **Chord Tone Change** feature; it received a 5% increase as compared with the PCFG that only used the **Pitch-Class** feature (which could be considered a baseline). It gained an even bigger increase when using the **Metric Delta Full** feature, an almost 9% increase in efficacy compared with the **Pitch-Class** test. Combining metric and chord features with the **Key-Relative Pitch-Class** encoding did not provide much further gain that with the metric feature alone. The prolongational reduction also improved when given the metric delta information, however the harmonic relationship feature affected the outcome very little.

The best performing PCFG was induced from the prolongational reduction trees, and used a data representation that included the **Key-Relative Pitch-Class** encoding combined with both the simplified metric delta and the chord tone information.

It is possible that the lack of data and the subsequent limitation on the complexity of the data representation could be avoided by the use of probabilistic smoothing techniques (to estimate the distributions of those rules that did not exist in the training set) [14, 97]. Indeed, the use of the **Key-Relative Pitch-class** feature as the basis for most of the representations was an attempt to limit the number of resulting rules, and therefore the number of zero-probability rules. This would be an appropriate topic for future experimentation.

A specific example helps to illustrate both the effectiveness and the drawbacks of using the induced PCFG for melodic reduction. Figure 5 displays the iterative reductions applied by pruning a PCFG tree, level by level. The grammar used to create this reduction was trained on prolongational reduction trees, and included the **Key-Relative Pitch-class** intervals, with notations for the **Metric Delta Reduced** feature, and the **Chord Tone Change** feature. This PCFG was the best performing, according to the evaluation metric. From a musicological perspective, the PCFG initially makes relatively sound decisions when reducing notes from the music surface. It is only when it begins to make decisions at the deeper levels of reduction that it chooses incorrect notes as the more important tones.

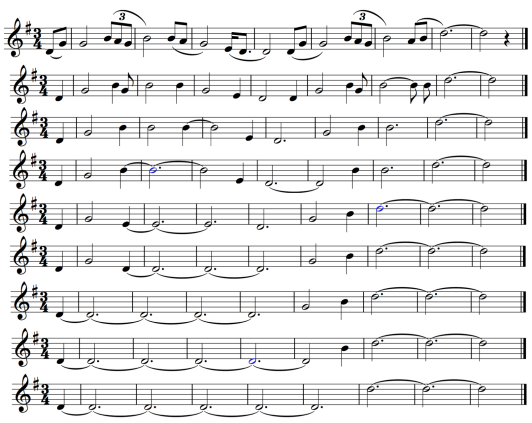


Figure 5: A set of melodies that show the progressive reductions, using the data representation that includes key-relative pitch-class, metric delta and chord tone features.

7. CONCLUSION

This research has performed for the first time the induction of a PCFG from a treebank of solutions for the process of melodic reduction. It was shown that, for the most part, adding metric or harmonic information in the data representation improves the efficacy of the resulting probabilistic model, when analyzing the results for the model's ability to reduce melodies in a musically sound way. A specific example reduction was generated by the best-performing model. There is still much room for improvement, because it seems that the model is more effective at identifying melodic embellishments on the musical surface, and is not able to identify the most important structural notes at deeper layers of the melodic reductions. The source code for this work also allows any researcher to create their own interval representations, and convert the GTTM dataset into a PCFG treebank.

There are some specific areas of improvement that might benefit this method. Currently there is no way to identify which chord a note belongs to with the grammar—the harmonic data is simply a boolean that describes whether or not the note is a chord tone. If there were a way to identify *which* chord the note belonged to, it would likely help with the grouping of larger phrases in the reduction hierarchy. For example, if a group of consecutive notes belong to the same underlying harmony, they could be grouped together, which might allow the PCFG to better identify the more important notes (assuming they fall at the beginning or end of phrases/groups). Beyond that, it would be greatly helpful if the sequences of chords could be considered as well. Furthermore, there is no way to explicitly identify repetition in the melodies with this model. That, too, might be able to assist the model, because if it can identify similar phrases, it could potentially identify the structural notes on which those phrases rely.

The source code for this research is available to the public, and can be found on the author's github account [1] .

[1] http://www.github.com/bigpianist/SupervisedPCFG_MelodicReduction

8. REFERENCES

[1] Samer A. Abdallah and Nicolas E. Gold. Comparing models of symbolic music using probabilistic grammars and probabilistic programming. In *Proceedings of the International Computer Music Conference*, pages 1524–31, Athens, Greece, 2014.

[2] Samer A. Abdallah, Nicolas E. Gold, and Alan Marsden. Analysing symbolic music with probabilistic grammars. In David Meredith, editor, *Computational Music Analysis*, pages 157–89. Springer International, Cham, Switzerland, 2016.

[3] John W. Backus. The syntax and semantics of the proposed international algebraic language of the Zurich ACM-GAMM conference. In *Proceedings of the International Conference for Information Processing*, pages 125–31, Paris, France, 1959.

[4] Mario Baroni, R. Brunetti, L. Callegari, and C. Jacoboni. A grammar for melody: Relationships between melody and harmony. In Mario Baroni and L Callegari, editors, *Musical Grammars and Computer Analysis*, pages 201–18, Florence, Italy, 1982.

[5] Mario Baroni and C. Jacobini. Analysis and generation of Bach's chorale melodies. In *Proceedings of the International Congress on the Semiotics of Music*, pages 125–34, Belgrade, Yugoslavia, 1975.

[6] Mario Baroni and C. Jacoboni. *Proposal for a grammar of melody: The Bach Chorales*. Les Presses de l'Université de Montréal, Montreal, Canada, 1978.

[7] David Beach. A Schenker bibliography. *Journal of Music Theory*, 13(1):2–37, 1969.

[8] Noam Chomsky. Three models for the description of language. *Institute of Radio Engineers Transactions on Information Theory*, 2:113–24, 1956.

[9] Noam Chomsky. On certain formal properties of grammars. *Information and Control*, 2(2):137–67, 1959.

[10] B. Frankland and Annabel J. Cohen. Parsing of melody: Quantification and testing of the local grouping rules of Lerdahl and Jackendoff's "A generative theory of tonal music". *Music Perception*, 21(4):499–543, 2004.

[11] Édouard. Gilbert and Darrell Conklin. A probabilistic context-free grammar for melodic reduction. In *Proceedings for the International Workshop on Artificial Intelligence and Music, International Joint Conference on Artificial Intelligence*, pages 83–94, Hyderabad, India, 2007.

[12] Masatoshi Hamanaka, K. Hirata, and Satoshi Tojo. σGTTM III: Learning based time-span tree generator based on PCFG. In *Proceedings of the Symposium on Computer Music Multidisciplinary Research*, Plymouth, UK, 2015.

[13] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. Implementing "A generative theory of tonal music". *Journal of New Music Research*, 35(4):249–77, 2007.

[14] Daniel Jurafsky and James H. Martin. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall, Upper Saddle River, NJ, 1st edition, 2000.

[15] Phillip B. Kirlin. *A probabilistic model of hierarchical music analysis*. Ph.D. thesis, University of Massachusetts Amherst, Amherst, MA, 2014.

[16] Fred Lerdahl and Ray Jackendoff. *A generative theory of tonal music*. The MIT Press, Cambridge, MA, 1983.

[17] Fred Lerdahl and Yves Potard. *La composition assistée par ordinateur*. Rapports de recherche. Institut de Recherche et Coordination Acoustique/Musique, Centre Georges Pompidou, Paris, France, 1986.

[18] Edward Loper and Steven Bird. NLTK: The natural language toolkit. In *Proceedings of the Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, volume 1, pages 63–70, Stroudsburg, PA, 2002.

[19] Alan Marsden. Recognition of variations using automatic Schenkerian reduction. In *Proceedings of the International Conference on Music Information Retrieval*, pages 501–6, Utrecht, Netherlands, August 9-13 2010.

[20] Christopher Roads and Paul Wieneke. Grammars as representations for music. *Computer Music Journal*, 3(1):48–55, March 1979.

[21] Nicolas Ruwet. Theorie et methodes dans les etudes musicales. *Musique en Jeu*, 17:11–36, 1975.

[22] Stephen W. Smoliar. Music programs: An approach to music theory through computational linguistics. *Journal of Music Theory*, 20(1):105–31, 1976.

A NEURAL GREEDY MODEL FOR VOICE SEPARATION IN SYMBOLIC MUSIC

Patrick Gray
School of EECS
Ohio University, Athens, OH
pg219709@ohio.edu

Razvan Bunescu
School of EECS
Ohio University, Athens, OH
bunescu@ohio.edu

ABSTRACT

Music is often experienced as a simultaneous progression of multiple streams of notes, or voices. The automatic separation of music into voices is complicated by the fact that music spans a voice-leading continuum ranging from monophonic, to homophonic, to polyphonic, often within the same work. We address this diversity by defining voice separation as the task of partitioning music into streams that exhibit both a high degree of external perceptual separation from the other streams and a high degree of internal perceptual consistency, to the maximum degree that is possible in the given musical input. Equipped with this task definition, we manually annotated a corpus of popular music and used it to train a neural network with one hidden layer that is connected to a diverse set of perceptually informed input features. The trained neural model greedily assigns notes to voices in a left to right traversal of the input chord sequence. When evaluated on the extraction of consecutive within voice note pairs, the model obtains over 91% F-measure, surpassing a strong baseline based on an iterative application of an envelope extraction function.

1. INTRODUCTION AND MOTIVATION

The separation of symbolic music into perceptually independent streams of notes, i.e. voices or lines, is generally considered to be an important pre-processing step for a number of applications in music information retrieval, such as query by humming (matching monophonic queries against databases of polyphonic or homophonic music) [13] or theme identification [12]. Voice separation adds structure to music and thus enables the implementation of more sophisticated music analysis tasks [17]. Depending on their definition of voice, existing approaches to voice separation in symbolic music can be organized in two main categories: 1) approaches that extract voices as monophonic sequences of successive non-overlapping musical notes [5, 6, 8, 11, 14, 16, 17]; and 2) approaches that allow voices to contain simultaneous note events, such as chords [4, 9, 10, 15]. Approaches in the first category typically use the musicological notion of voice that is referenced in the voice-leading rules of the Western musical tradition, rules that govern the horizontal motion of individual voices from note to note in successive chords [1, 4]. Starting with [4], approaches in the second category break with the musicological notion of voice and emphasize a perceptual view of musical voice that corresponds more closely to the notion of independent auditory streams [2, 3]. Orthogonal to this categorization, a limited number of voice separation approaches are formulated as parametric models, with parameters that are trained on music already labeled with voice information [6, 8, 11].

In this paper, we propose a data-driven approach to voice separation that preserves the musicological notion of voice. Our aim is to obtain a segregation of music into voices that would enable a downstream system to determine whether an arbitrary musical input satisfies the known set of voice-leading rules, or conversely identify places where the input violates voice-leading rules.

2. TASK DEFINITION

According to Huron [7], "the principal purpose of voice-leading is to create perceptually independent musical lines". However, if a voice is taken to be a monophonic sequence of notes, as implied by traditional voice-leading rules [1], then not all music is composed of independent musical lines. In homophonic accompaniment, for example, multiple musical lines (are meant to) fuse together into one perceptual stream. As Cambouropoulos [4] observes for homophonic accompaniment, "traditional voice-leading results in perceivable musical *texture*, not independent musical lines". In contrast with the traditional notion of voice used in previous voice separation approaches, Cambouropoulos redefines in [4] the task of 'voice' separation as that of separating music into perceptually independent musical *streams*, where a stream may contain two or more synchronous notes that are perceived as fusing in the same auditory stream. This definition is used in [9, 15] to build automatic approaches for splitting symbolic music into perceptually independent musical streams.

Since a major aim of our approach is to enable building "musical critics" that automatically determine whether an arbitrary musical input obeys traditional voice-leading rules, we adopt the musicological notion of voice as a

© Patrick Gray, Razvan Bunescu. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Patrick Gray, Razvan Bunescu. "A Neural Greedy Model for Voice Separation in Symbolic Music", 17th International Society for Music Information Retrieval Conference, 2016.

monophonic sequence of non-overlapping notes. This definition however leads to an underspecified voice separation task: for any non-trivial musical input, there usually is a large number of possible separations into voices that satisfy the constraints that they are monophonic and contain notes in chronological order that do not overlap. Further constraining the voices to be perceptually independent would mean the definition could no longer apply to music with homophonic textures, as Cambouropoulos correctly noticed in [4]. Since we intend the voice separation approach to be applicable to arbitrary musical input, we instead define voice separation as follows:

Definition 1. *Voice separation is the task of partitioning music into monophonic sequences (voices) of non-overlapping notes that exhibit both a high degree of external perceptual separation from the other voices and a high degree of internal perceptual consistency, to the maximum degree that is possible in the given musical input.*

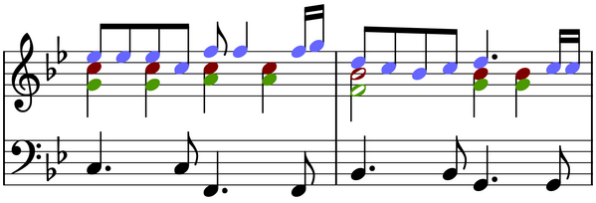


Figure 1. Example voice separation from "Earth Song".

Figure 1 shows a simple example of voice separation obtained using the definition above. While the soprano and bass lines can be heard as perceptually distinct voices, we cannot say the same for the tenor and alto lines shown in green and red, respectively. However, clear perceptual independence is not needed under the new task definition. The two intermediate voices exhibit a high degree of perceptual consistency: their consecutive notes satisfy to a large extent the pitch proximity and temporal continuity principles needed to evoke strong auditory streams [7]. Indeed, when heard in isolation, both the tenor and the alto are heard as continuous auditory streams, the same streams that are also heard when the two are played together. The two streams do not overlap, which helps with perceptual tracking [7]. Furthermore, out of all the streaming possibilities, they also exhibit the largest possible degree of external perceptual separation from each other and from the other voices in the given musical input.

3. ANNOTATION GUIDELINES

According to the definition in Section 2, voice separation requires partitioning music into monophonic sequences of non-overlapping notes that exhibit a high degree of perceptual salience, to the maximum extent that is possible in the given musical input. As such, an overriding principle that we followed during the manual annotation process was to always give precedence to what was heard in the music, even when this appeared to contradict formal perceptual principles, such as pitch proximity. Furthermore, whenever formal principles seemed to be violated by perceptual streams, an attempt was made to explain the apparent conflict. Providing justifications for non-trivial annotation decisions enabled refining existing formal perceptual principles and also informed the feature engineering effort.

Listening to the original music is often not sufficient on its own for voice separation, as not all the voices contained in a given musical input can be distinctly heard. Because we give precedence to perception, we first annotated those voices that could be distinguished clearly in the music, which often meant annotating first the melodic lines in the soprano and the bass. When the intermediate voices were difficult to hear because of being masked by more salient voices, one simple test was to remove the already annotated most prominent voice (often the soprano [1]) and listen to the result. Alternatively, when multiple conflicting voice separations were plausible, we annotated the voice that, after listening to it in isolation, was easiest to distinguish perceptually in the original music.

Figure 2 shows two examples where the perceptual principle of pitch proximity appears to conflict with what is heard as the most salient voice. In the first measure, the first D_4 note can continue with any of the 3 notes in the following I^6 chord. However, although the bass note in the chord has the same pitch, we hear the first D_4 most saliently as part of the melody in the soprano. The D_4 can also be heard as creating a musical line with the next D_4 notes in the bass, although less prominently. The least salient voice assignment would be between the D_4 and the intermediate line that starts on the following G_4. While we annotate all these streaming possibilities (as shown in Figure 7), we mark the soprano line assignment as the most salient for the D_4. Similarly, in the last chord from the second measure from Figure 2, although E_4 is closer to the previous F_4, it is the G_4 that is most prominently heard as continuing the soprano line. This was likely reinforced by the fact that the G_4 in the last chord was "prepared" by the G_4 preceding F_4.

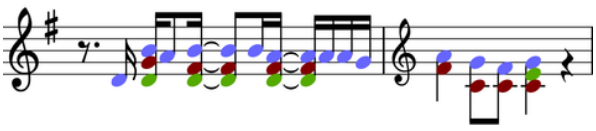


Figure 2. Voice separation annotations, for measures 5 in "Knockin' on Heaven's Door" and 12 in "Let It Be".

Other non-trivial annotation decisions, especially in the beginning of the annotation effort, involved whether two streams should be connected or not. Overall, we adopted the guideline that we should break the music into fewer and consequently longer voices, especially if validated perceptually. Figure 3, for example, shows the A_4 in the third measure connected to the following C_5. Even though the two notes are separated by a quarter rest, they are heard as belonging to the same stream, which may also be helped by the relatively long duration of A_4 and by the fact that the same pattern is repeated in the piece. We have also dis-

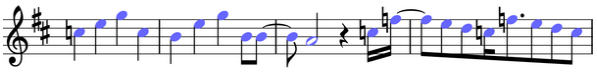


Figure 3. Voice separation annotation in the treble for measures 38-41 in "Count on Me".

covered that "preparation" through previous occurrences of the same note or notes one octave above or below can significantly attenuate the effect of a large pitch distance and thus help with connecting the note to an active stream. This effect is shown in Figure 4, where the voice in the first measure is most prominently heard as continuing with the B_4 in the second measure.

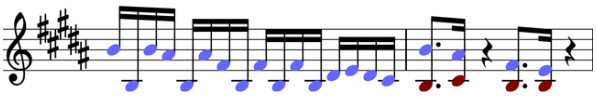


Figure 4. Voice separation annotation in the treble for measures 26-27 in "A Thousand Miles".

Sometimes, the assignment of a note to one of the available active voices is hard to make due to inherent musical ambiguity. An example is shown in Figure 5, where it is hard to determine if the A_4 in the second measure connects to the top C_6 or the C_5 one octave below. After being played separately, each voice assignment can be distinguished perceptually in the original music. The C_5 is closer in pitch to the A_4 and it is also in a range with better defined pitch sensations than the C_6. On the other hand, the pitch distance between the upper C_6 and the A_4 is attenuated by the synchronous C_5. Eventually we annotated A_4 as connecting to the slightly more salient C_5, but also marked it as ambiguous between the two C notes.

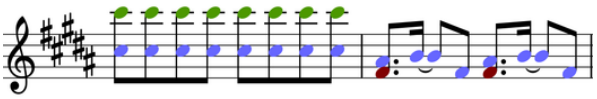


Figure 5. Voice separation annotation in the treble for measures 62-63 in "A Thousand Miles".

Other examples of harmony influencing voice assignment involve the seventh scale degree notes appearing in VII and VII^6 chords. As shown in Figure 6, when such a chord is first used, the $\hat{7}$ note does connect to any of the previous streams, despite the closer pitch proximity.

4. VOICE SEPARATION DATASET

We compiled a corpus of piano versions of 20 popular compositions of varying complexity that are representative of many genres of music. Each song was downloaded from *www.musescore.com* and converted to MusicXML. In selecting music, we followed a few basic criteria. First, we avoided collecting piano accompaniments and gave preference to piano renditions that sounded as much as possible like the original song. Among other things, this ensured that each score contained at least one clearly defined

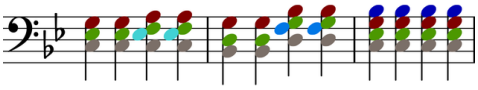


Figure 6. Voice separation annotation in the bass for measures 26-28 in "Earth Song".

melody. Second, we collected only tonal music. Atonal music is often comprised of unusual melodic structures, which were observed to lead to a poor perception of voices by the annotators. Following the annotation guidelines, we manually labeled the voice for each note in the dataset. The annotations will be made publicly available. The names of the 20 musical pieces are shown in Table 1, together with statistics such as the total number of notes, number of voices, average number of notes per voice, number of within-voice note pairs, number of unique note onsets, and average number of notes per chord. The 20 songs were manually annotated by the first author; additionally, the 10 songs marked with a star were also annotated by the second author. In terms of F-measure, the inter-annotator agreement (ITA) on the 10 songs is 96.08% (more detailed ITA numbers are shown in Table 2). The last column shows the (macro-averaged) F-measure of our neural greedy model, to be discussed in Section 6. As can be seen in Table 1, the number of voices varies widely, ranging between 4 for Greensleeves to 123 for 21 Guns, the longest musical composition, with a variable musical texture and frequent breaks in the harmonic accompaniment of the melody. The last line shows the same total/average statistics for the first 50 four-part Bach Chorales available in Music21, for which we use the original partition into voices, without the duplication of unisons.

5. THE VOICE SEPARATION MODEL

To separate a musical input into its constituent voices, we first order all the notes based on their onsets into a sequence of *chords* $\mathcal{C} = \{c_1, c_2, ..., c_T\}$, where a chord is defined to be a maximal group of notes that have the same onset. Assignment of notes to voices is then performed in chronological order, from left to right, starting with the first chord c_1. Because voices are by definition monophonic, each note in the first chord is considered to start a separate, new voice. These first voices, together with an empty voice ϵ, constitute the initial set of *active voices* $\mathcal{V}$. At each onset t, the algorithm greedily assigns a note n from the current chord c_t to one of the voices in the active set by selecting the active voice v that maximizes a trained assignment probability $p(n, v)$, i.e. $v(n) = \arg\max_{v \in \hat{\mathcal{V}}} p(n, v)$. Notes from the current chord are assigned to voices in the order of their maximal score $p(n, v(n))$. If a note is assigned to the empty voice, then a new voice is added to the active set. The set of candidate active voices $\hat{\mathcal{V}}$ available for any given note n is a subset of active voices $\mathcal{V}$ constrained such that assigning n to any of the voices in $\hat{\mathcal{V}}$ would not lead to crossing voices or to multiple synchronous notes being assigned to the same voice.

Popular Music dataset	# Notes	# Voices	# N / V	# Pairs	# Onsets	Synchronicity	F-measure
21 Guns (Green Day)	1969	123	16.01	1801	666	2.96	86.24
Apples to the Core (Daniel Ingram)	923	29	31.83	892	397	2.32	77.67
Count on Me (Bruno Mars)	775	11	70.45	764	473	1.64	97.22
Dreams (Rogue)*	615	12	51.25	603	474	1.30	98.32
Earth Song (Michael Jackson)*	431	15	28.73	416	216	2.00	93.27
Endless Love (Lionel Richie)	909	23	39.52	886	481	1.89	96.52
Forest (Twenty One Pilots)	1784	89	20.04	1695	1090	1.64	91.93
Fur Elise (Ludwig van Beethoven)*	900	77	11.69	823	653	1.38	91.98
Greensleeves*	231	4	57.75	213	72	3.21	92.88
How to Save a Life (The Fray)*	440	13	33.85	427	291	1.51	98.11
Hymn for the Weekend (Coldplay)	1269	50	25.38	1218	706	1.80	92.30
Knockin' on Heaven's Door (Bob Dylan)*	355	41	8.66	312	180	1.97	90.92
Let It Be (The Beatles)*	563	22	25.59	540	251	2.24	87.29
One Call Away (Charlie Puth)	993	56	17.73	937	505	1.97	91.33
See You Again (Wiz Khalifa)*	704	66	10.67	638	359	1.96	81.16
Teenagers (My Chemical Romance)	315	18	17.50	297	145	2.17	91.39
A Thousand Miles (Vanessa Carlton)*	1001	61	16.41	937	458	2.19	96.61
To a Wild Rose (Edward Macdowell)	307	20	15.35	287	132	2.33	88.72
Uptown Girl (Billy Joel)	606	46	13.17	560	297	2.04	93.41
When I Look at You (Miley Cyrus)*	1152	82	14.05	1067	683	1.69	92.92
Totals & Averages	16242	42.90	26.28	15313	8529	2.01	91.51
Bach Chorales dataset	12282	4	61.41	11874	4519	2.73	95.47

Table 1. Statistics for the Popular Music dataset and the Bach Chorales dataset.

The assignment probability $p(n, v)$ captures the compatibility between a note n and an active voice v. To compute it, we first define a vector $\Phi(n, v)$ of perceptually informed compatibility features (Section 5.2). The probability is then computed as $p(n, v) = \sigma(\mathbf{w}^T h_W(n, v))$, where σ is the sigmoid function and $h_W(n, v)$ is the vector of activations of the neurons on the last (hidden) layer in a neural network with input $\Phi(n, v)$.

To train the network parameters $\theta = [\mathbf{w}, W]$, we maximize the likelihood of the training data:

$$\hat{\theta} = \arg\max_{\theta} \prod_{t=1}^{T} \prod_{n \in c_t} \prod_{v \in \hat{\mathcal{V}}} p(n,v|\theta)^{l(n,v)} (1-p(n,v|\theta))^{1-l(n,v)} \quad (1)$$

where $l(n, v)$ is a binary label that indicates whether or not note n was annotated to belong to voice v in the training data. This formulation of the objective function is flexible enough to be used in 2 types of voice separation scenarios:

1. **Ranking**: Assign a note to the top-ranked candidate active voice, i.e. $v(n) = \arg\max_{v \in \hat{\mathcal{V}}} p(n, v)$.
2. **Multi-label classification**: Assign a note to all candidate active voices whose assignment probability is large enough, i.e. $V(n) = \{v \in \hat{\mathcal{V}} | p(n, v) > 0.5\}$.

The first scenario is the simplest one and rests on the working assumption that a note can belong to a single voice. The second scenario is more general and allows a note to belong to more than one voice. Such capability would be useful in cases where a note is heard simultaneously as part of two musical streams. Figure 7, for example, shows the voice separation performed under the two scenarios for the same measure. In the ranking approach shown on the left, we label the second F_4 as belonging to the soprano voice. Since in this scenario we can assign a note to just one voice, we select the voice assignment that is heard as the most salient, which in this case is the soprano. In the multi-label approach shown on the right, we label the second F_4 as belonging to both active voices, since the note is heard as belonging to both. In the experiments that we report in this paper (Section 6), we used the simpler ranking approach, leaving the more general multi-label approach for future work.

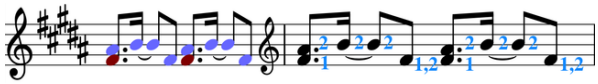


Figure 7. Two voice separation scenarios, for measure 16 from "A Thousand Miles".

5.1 Iterative Envelope Extraction

We also propose a baseline system for voice-separation that iteratively extracts the upper *envelope* i.e. the topmost monophonic sequence of non-overlapping notes. Figure 8 shows how the iterative envelope extraction process works on the second measure from Figure 2, copied here for readability. The top left measure is the original measure from Figure 2 and we use it as the current input. Its upper envelope is shown in the bottom left measure, which will become the first voice. After extracting the first voice from the input, we obtain the second measure in the top staff, which is now set to be the current input. We again apply

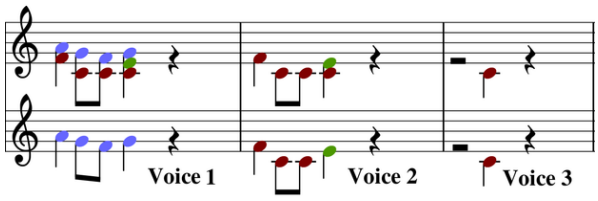


Figure 8. Voice separation as iterative envelope extraction.

the same envelope extraction process to obtain the second voice, shown in the second measure on the bottom staff. After extracting the second voice from the current input, we obtain a new current input, shown in the third measure on the top staff. Extracting the third voice from the current input results in an empty set and correspondingly the baseline algorithm stops. For this input, the baseline extracted voice 1 without errors, however it made a mistake in the last note assignment for voice 2.

5.2 Voice Separation Features

The assignment probability $p(n, v)$ is computed by the neural model based on a vector of input features $\Phi(n, v) = [\phi_0, \phi_1, ..., \phi_K]$ that will be described in this section, using $v.last$ to denote the last note in the active voice v.

5.2.1 Empty Voice Feature

The empty voice feature ϕ_0 is set to 1 only for the empty voice, i.e. $\phi_0(n, \epsilon) = 1$ and $\phi_0(n, v) = 0, \forall v \neq \epsilon$. All the remaining features in any feature vector for an empty voice $\Phi(n, \epsilon)$ are set to zero. This allows the empty voice to activate a bias parameter w_0, which is equivalent to learning a threshold $-w_0$ that the weighted combination of the remaining features must exceed in order for the note to be assigned to an existing, non-empty, active voice. Otherwise, the note n will be assigned to the empty voice, meaning it will start a new voice.

5.2.2 Pitch and Pitch Proximity Features

According to Huron's formulation of the pitch proximity principle, *the coherence of an auditory stream is maintained by close pitch proximity in successive tones within the stream* [7]. Correspondingly, we define a pitch proximity feature $\phi_1(n, v) = pd(n, v.last) = |ps(n) - ps(v.last)|$ to be the absolute distance in half steps between the pitch space representations of notes n and $v.last$. The pitch proximity feature enables our system to quickly learn that notes rarely pair with voices lying at intervals beyond an octave. We also add two features $\phi_2(n, v) = ps(n)$ and $\phi_3(n, v) = ps(v.last)$ that capture the absolute pitch of the note n and $v.last$. Pitch values are taken from a pitch space in which C_4 has value 60 and a difference of 1 corresponds to one half step, e.g. C_5 has value 72. Using absolute pitches as separate input features will enable neurons on the hidden layer to discover possibly unknown pitch-based rules for perceptual streaming.

5.2.3 Temporal and Temporal Continuity Features

We define an inter-onset feature $\phi_4(n, v)$ as the temporal distance between the note onsets of n and $v.last$. An additional feature $\phi_5(n, v)$ is computed as the temporal distance between the note onset of n and the note offset (the time when a note ends) of $v.last$. These complementary features help our system model both acceptable rest lengths between notes and the gradual dissipation of note salience throughout the duration of a note.

Notes that lie between the onsets of $v.last$ and n may influence the voice assignment. Thus, we appropriately define a feature $\phi_6(n, v)$ as the number of unique onsets between the onsets of $v.last$ and n. We also define two features $\phi_7(n, v) = qd(n)$ and $\phi_8(n, v) = qd(v.last)$ for the durations of n and $v.last$, respectively, where note durations are measured relative to the quarter note. These features, when combined in the hidden layer, enable the system to learn to pair notes that appear in common duration patterns, such as dotted quarter followed by an eighth.

5.2.4 Chordal Features

Notes that reside in the soprano either alone or at the top of a chord tend to be heard as the most salient. As a result, the most prominent melodic line of a score often navigates through the topmost notes, even in situations where a candidate active voice lies closer in pitch to the alto or tenor notes of the current chord. Notes in a low bass range that stand alone or at the bottom of a chord exhibit a similar behavior. To enable the learning model to capture this perceptual effect, we define two features $\phi_9(n, v) = cp(n)$ and $\phi_{10}(n, v) = cp(v.last)$ to mark the relative positions of n and $v.last$ in their respective chords, where the chord position number (cp) starts at 0 from the top of a chord. To place chord positions into the appropriate context, we define $\phi_{11}(n, v)$ as the number of notes in n's chord and $\phi_{12}(n, v)$ as the number of notes in $v.last$'s chord. For more direct comparisons between notes in n's chord and the active voice, we calculate pitch proximities (*pd*) between $v.last$ and n's upper and lower neighbors $n.above$ and $n.below$. Thus, we define the features $\phi_{13}(n, v) = pd(v.last, n.above)$ and $\phi_{14}(n, v) = pd(v.last, n.below)$. We also add the features $\phi_{15}(n, v) = pd(n, n.above)$ and $\phi_{16}(n, v) = pd(n, n.below)$ to encode the intervals between n and its chordal neighbors.

5.2.5 Tonal Features

We use scale degrees $\phi_{17}(n, v) = sd(n)$ and $\phi_{18}(n, v) = sd(v.last)$ of the notes n and $v.last$ as features in order to enable the model to learn melodic intervals that are most appropriate in a given key. For example, if a candidate active voice ends on a leading tone, then it is likely to resolve to the tonic. We also define a feature $\phi_{19}(n, v)$ for the interval between the note n and the root of its chord, and similarly, a feature $\phi_{20}(n, v)$ for the interval between the note $v.last$ and the root of its chord.

The last tonal feature $\phi_{21}(n, v)$ is a Boolean feature that is set to 1 if the note $v.last$ in the active voice v appears in a tonic chord at a cadence. Tonic chords at cadences induce a sense of finality [1], which could potentially break the voice from the notes that follow.

5.2.6 Pseudo-polyphony Features

In pseudo-polyphony, two perceptually independent streams are heard within a rapidly alternating, monophonic sequence of notes separated by relatively large pitch intervals. Figure 9 presents an example of pseudo-polyphony.

Dataset	Model	All within-voice pairs of notes				Exclude pairs of notes separated by rests			
		Jaccard	Precision	Recall	F-measure	Jaccard	Precision	Recall	F-measure
Popular Music	Baseline	59.07	74.51	74.03	74.27	67.55	80.48	80.79	80.64
	NGModel	83.55	92.08	90.01	91.03	85.73	92.74	91.89	92.31
	ITA	92.45	94.96	97.21	96.08	93.04	95.12	97.70	96.41
Bach Chorales	Baseline	87.25	93.34	93.04	93.18	87.62	93.22	93.58	93.39
	NGModel	91.36	95.59	95.37	95.47	91.66	95.91	95.39	95.64

Table 2. Comparative results of Neural Greedy (NG) Model vs. Baseline on Popular Music and Bach Chorales; Inter-Annotator (ITA) results on the subset of 10 popular songs shown in Table 1.

Although the offset of each D_4 note is immediately followed by the onset of the next note, the often large intervals and the fast tempo break the upper and lower notes into two perceptually independent streams.

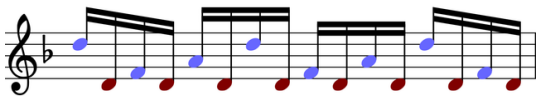


Figure 9. Example pseudo-polyphony from "Forest".

We model this phenomenon by introducing three features to the neural system. In designing these features, we first employ the envelope extraction method described in Section 5.1 to gather monophonic sequences of non-overlapping notes. We next find the maximal contiguous subsequences with an alternating up-down pattern of direction changes, like the one shown in Figure 9. The first feature $\phi_{22}(n, v) = apv(n)$ is set to be the alternating path value (*apv*) of the note n, which is 0 if n is not on an alternating path, 1 if it is in the lower part of an alternating path, and 2 if it is in the upper part of an alternating path. Similarly, we define $\phi_{23}(n, v) = apv(v.last)$ to be the alternating path value of the note $v.last$. The third feature is set to 1 if both n and $v.last$ have the same alternating path value, i.e. $\phi_{24}(n, v) = 1[apv(n) = apv(v.last)]$.

6. EXPERIMENTAL EVALUATION

We implemented the neural greedy model as a neural network with one hidden layer, an input layer consisting of the feature vector $\Phi(n, v)$, and an output sigmoid unit that computes the assignment probability $p(n, v|\theta)$. The network was trained to optimize a regularized version of the likelihood objective shown in Equation 1 using gradient descent and backpropagation. The model was trained and tested using 10-fold cross-validation. For evaluation, we considered pairs of consecutive notes from the voices extracted by the system and compared them with pairs of consecutive notes from the manually annotated voices. Table 2 shows results on the two datasets in terms of the Jaccard similarity between the system pairs and the true pairs, precision, recall, and micro-averaged F-measure. Precision and recall are equivalent to the soundness and completeness measures used in [6, 11]. We also report results for which pairs of notes separated by rests are ignored.

The results show that the newly proposed neural model performs significantly better than the envelope baseline, especially on popular music. When pairs of notes separated by rests are excluded from evaluation, the baseline performance increases considerably, likely due to the exclusion of pseudo-polyphonic passages.

Dataset	Model	Precision	Recall	F-measure
10 Fugues	[6]	94.07	93.42	93.74
	NGModel	95.56	92.24	93.87
30 Inv. 48 F.	[14]	95.94	70.11	81.01
	NGModel	95.91	93.83	94.87

Table 3. Comparative results on Bach datasets.

Close to our model is the data-driven approach from [6] for voice separation in lute tablature. Whereas we adopt a ranking approach and use as input both the note and the candidate active voice, [6] use only the note as input and associate voices with the output nodes. Therefore, while our ranking approach can label music with a variable number of voices, the classification model from [6] can extract only a fixed number of voices. Table 3 shows that our neural ranking model, although not specifically designed for music with a fixed number of voices, performs competitively with [6] when evaluated on the same datasets of 10 Fugues by Bach. We also compare the neural ranking model with the the approach from [14] on a different dataset containing 30 inventions and 48 fugues [1] .

7. CONCLUSION AND FUTURE WORK

We presented a neural model for voice separation in symbolic music that assigns notes to active voices using a greedy ranking approach. The neural network is trained on a manually annotated dataset, using a perceptually-informed definition of voice that also conforms to the musicological notion of voice as a monophonic sequence of notes. When used with a rich set of note-voice features, the neural greedy model outperforms a newly introduced strong baseline using iterative envelope extraction. In future work we plan to evaluate the model in the more general multi-label classification setting that allows notes to belong to multiple voices.

We would like to thank the anonymous reviewers for their helpful remarks and Mohamed Behairy for insightful discussions on music cognition.

[1] In [14] it is stated that soundness and completeness "as suggested by Kirlin [11]" were used for evaluation; however, the textual definitions given in [14] are not consistent with [11]. As was done in [6], for lack of an answer to this inconsistency, we present the metrics exactly as in [14].

8. REFERENCES

[1] E. Aldwell, C. Schachter, and A. Cadwallader. *Harmony and Voice Leading*. Schirmer, 4 edition, 2011.

[2] A. S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. The MIT Press, Cambridge, MA, 1990.

[3] A. S. Bregman and J. Campbell. Primary Auditory Stream Segregation and Perception of Order in Rapid Sequences of Tones. *Journal of Experimental Psychology*, 89(2):244–249, 1971.

[4] E. Cambouropoulos. 'Voice' Separation: Theoretical, Perceptual, and Computational Perspectives. In *Proceedings of the* 9^{th} *International Conference on Music Perception and Cognition*, pages 987–997, Bologna, Italy, 2006.

[5] E. Chew and X. Wu. Separating Voices in Polyphonic Music: A Contig Mapping Approach. In *Computer Music Modeling and Retrieval:* 2^{nd} *International Symposium*, pages 1–20, 2004.

[6] R. de Valk, T. Weyde, and E. Benetos. A Machine Learning Approach to Voice Separation in Lute Tablature. In *Proceedings of the* 14^{th} *International Society for Music Information Retrieval Conference*, pages 555–560, Curitiba, Brazil, 2013.

[7] D. Huron. Tone and Voice: A Derivation of the Rules of Voice-Leading from Perceptual Principles. *Music Perception*, 19(1):1–64, 2001.

[8] A. Jordanous. Voice Separation in Polyphonic Music: A Data-Driven Approach. In *Proceedings of the International Computer Music Conference*, Belfast, Ireland, 2008.

[9] I. Karydis, A. Nanopoulos, A. N. Papadopoulos, and E. Cambouropoulos. VISA: The Voice Integration/Segregation Algorithm. In *Proceedings of the* 8^{th} *International Society for Music Information Retrieval Conference*, pages 445–448, Vienna, Austria, 2007.

[10] J. Kilian and H. Hoos. Voice Separation: A Local Optimization Approach. In *Proceedings of the* 3^{rd} *International Society for Music Information Retrieval Conference*, pages 39–46, Paris, France, 2002.

[11] P. B. Kirlin and P. E. Utgoff. VoiSe: Learning to Segregate Voices in Explicit and Implicit Polyphony. In *Proceedings of the* 6^{th} *International Society for Music Information Retrieval Conference*, pages 552–557, London, England, 2005.

[12] O. Lartillot. Discovering Musical Patterns Through Perceptive Heuristics. In *Proceedings of the* 4^{th} *International Society for Music Information Retrieval Conference*, pages 89–96, Washington D.C., USA, 2003.

[13] K. Lemstrom and J. Tarhio. Searching Monophonic Patterns within Polyphonic Sources. In *Proceedings of the* 6^{th} *Conference on Content-Based Multimedia Information Access*, pages 1261–1279, Paris, France, 2000.

[14] S. T. Madsen and G. Widmer. Separating Voices in MIDI. In *Proceedings of the* 7^{th} *International Society for Music Information Retrieval Conference*, pages 57–60, Victoria, Canada, 2006.

[15] D. Rafailidis, E. Cambouropoulos, and Y. Manolopoulos. Musical Voice Integration/Segregation: VISA Revisited. In *Proceedings of the* 6^{th} *Sound and Music Computing Conference*, pages 42–47, Porto, Portugal, 2009.

[16] D. Rafailidis, A. Nanopoulos, E. Cambouropoulos, and Y. Manolopoulos. Detection of Stream Segments in Symbolic Musical Data. In *Proceedings of the* 9^{th} *International Society for Music Information Retrieval Conference*, pages 83–88, Philadelphia, PA, 2008.

[17] D. Temperley. *The Cognition of Basic Musical Structures*. The MIT Press, Cambridge, MA, 2001.

TOWARDS SCORE FOLLOWING IN SHEET MUSIC IMAGES

Matthias Dorfer **Andreas Arzt** **Gerhard Widmer**
Department of Computational Perception, Johannes Kepler University Linz, Austria
matthias.dorfer@jku.at

ABSTRACT

This paper addresses the matching of short music audio snippets to the corresponding pixel location in images of sheet music. A system is presented that simultaneously learns to read notes, listens to music and matches the currently played music to its corresponding notes in the sheet. It consists of an end-to-end multi-modal convolutional neural network that takes as input images of sheet music and spectrograms of the respective audio snippets. It learns to predict, for a given unseen audio snippet (covering approximately one bar of music), the corresponding position in the respective score line. Our results suggest that with the use of (deep) neural networks – which have proven to be powerful image processing models – working with sheet music becomes feasible and a promising future research direction.

1. INTRODUCTION

Precisely linking a performance to its respective sheet music – commonly referred to as audio-to-score alignment – is an important topic in MIR and the basis for many applications [20]. For instance, the combination of score and audio supports algorithms and tools that help musicologists in in-depth performance analysis (see e.g. [6]), allows for new ways to browse and listen to classical music (e.g. [9, 13]), and can generally be helpful in the creation of training data for tasks like beat tracking or chord recognition. When done on-line, the alignment task is known as score following, and enables a range of applications like the synchronization of visualisations to the live music during concerts (e.g. [1, 17]), and automatic accompaniment and interaction live on stage (e.g. [5, 18]).

So far all approaches to this task depend on a symbolic, computer-readable representation of the sheet music, such as MusicXML or MIDI (see e.g. [1, 5, 8, 12, 14–18]). This representation is created either manually (e.g. via the time-consuming process of (re-)setting the score in a music notation program), or automatically via optical music recognition software. Unfortunately automatic methods are still highly unreliable and thus of limited use, especially for more complex music like orchestral scores [20].

© Matthias Dorfer, Andreas Arzt, Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Matthias Dorfer, Andreas Arzt, Gerhard Widmer. "Towards Score Following in Sheet Music Images", 17th International Society for Music Information Retrieval Conference, 2016.

The central idea of this paper is to develop a method that links the audio and the image of the sheet music *directly*, by *learning* correspondences between these two modalities, and thus making the complicated step of creating an in-between representation obsolete. We aim for an algorithm that simultaneously learns to *read notes*, *listens* to music and *matches* the currently played music with the correct notes in the sheet music. We will tackle the problem in an end-to-end neural network fashion, meaning that the entire behaviour of the algorithm is learned purely from data and no further manual feature engineering is required.

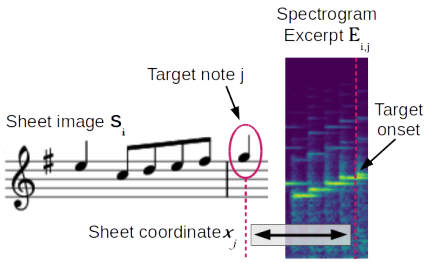
2. METHODS

This section describes the audio-to-sheet matching model and the input data required, and shows how the model is used at test time to predict the expected location of a new unseen audio snippets in the respective sheet image.

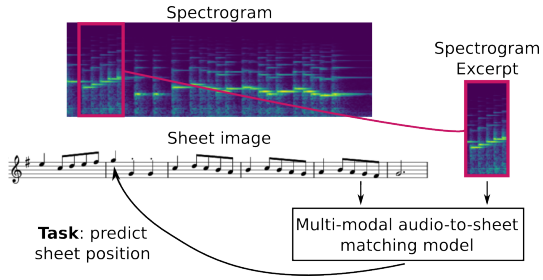
2.1 Data, Notation and Task Description

The model takes two different input modalities at the same time: images of scores, and short excerpts from spectrograms of audio renditions of the score (we will call these *query snippets* as the task is to predict the position in the score that corresponds to such an audio snippet). For this first proof-of-concept paper, we make a number of simplifying assumptions: for the time being, the system is fed only a *single staff line* at a time (not a full page of score). We restrict ourselves to *monophonic music*, and to the *piano*. To generate training examples, we produce a fixed-length query snippet for each note (onset) in the audio. The snippet covers the target note onset plus a few additional frames, at the end of the snippet, and a fixed-size context of 1.2 seconds into the past, to give some temporal context. The same procedure is followed when producing example queries for off-line testing.

A training/testing example is thus composed of two inputs: Input 1 is an image $\mathbf{S}_i$ (in our case of size 40×390 pixels) showing one staff of sheet music. Input 2 is an audio snippet – specifically, a spectrogram excerpt $\mathbf{E}_{i,j}$ (40 frames $\times$ 136 frequency bins) – cut from a recording of the piece, of fixed length (1.2 seconds). The rightmost onset in spectrogram excerpt $\mathbf{E}_{i,j}$ is interpreted as the target note j whose position we want to predict in staff image $\mathbf{S}_i$. For the music used in our experiments (Section 3) this context is a bit less than one bar. For each note j (represented by its corresponding spectrogram excerpt $\mathbf{E}_{i,j}$) we annotated its *ground truth* sheet location x_j in sheet image $\mathbf{S}_i$. Coor-



(a) Spectrogram-to-sheet correspondence. In this example the rightmost onset in spectrogram excerpt $\mathbf{E}_{i,j}$ corresponds to the rightmost note (target note j) in sheet image $\mathbf{S}_i$. For the present case the temporal context of about 1.2 seconds (into the past) covers five additional notes in the spectrogram. The staff image and spectrogram excerpt are exactly the multi-modal input presented to the proposed audio-to-sheet matching network. At train time the target pixel location x_j in the sheet image is available; at test time $\hat{x}_j$ has to be predicted by the model (see figure below).



(b) Schematic sketch of the audio-to-sheet matching task targeted in this work. Given a sheet image $\mathbf{S}_i$ and a short snippet of audio (spectrogram excerpt $\mathbf{E}_{i,j}$) the model has to predict the audio snippet's corresponding pixel location x_j in the image.

Figure 1: Input data and audio-to-sheet matching task.

dinate x_j is the distance of the note head (in pixels) from the left border of the image. As we work with single staffs of sheet music we only need the x-coordinate of the note at this point. Figure 1a relates all components involved.

Summary and Task Description: For training we present triples of (1) staff image $\mathbf{S}_i$, (2) spectrogram excerpt $\mathbf{E}_{i,j}$ and (3) ground truth pixel x-coordinate x_j to our audio-to-sheet matching model. At test time only the staff image and spectrogram excerpt are available and the task of the model is to predict the estimated pixel location $\hat{x}_j$ in the image. Figure 1b shows a sketch summarizing this task.

2.2 Audio-Sheet Matching as Bucket Classification

We now propose a multi-modal convolutional neural network architecture that learns to match unseen audio snippets (spectrogram excerpts) to their corresponding pixel location in the sheet image.

2.2.1 Network Structure

Figure 2 provides a general overview of the deep network and the proposed solution to the matching problem. As mentioned above, the model operates jointly on a staff image $\mathbf{S}_i$ and the audio (spectrogram) excerpt $\mathbf{E}_{i,j}$ related to a note j. The rightmost onset in the spectrogram excerpt is the one related to target note j. The multi-modal model consists of two specialized convolutional networks: one dealing with the sheet image and one dealing with the audio (spectrogram) input. In the subsequent layers we fuse the specialized sub-networks by concatenation of the latent image- and audio representations and additional processing by a sequence of dense layers. For a detailed description of the individual layers we refer to Table 1 in Section 3.4. The output layer of the network and the corresponding localization principle are explained in the following.

2.2.2 Audio-to-Sheet Bucket Classification

The objective for an unseen spectrogram excerpt and a corresponding staff of sheet music is to predict the excerpt's location x_j in the staff image. For this purpose we start with horizontally quantizing the sheet image into B non-overlapping buckets. This discretisation step is indicated as the short vertical lines in the staff image above the score in Figure 2. In a second step we create for each note j in the train set a target vector $\mathbf{t}_j = \{t_{j,b}\}$ where each vector element $t_{j,b}$ holds the probability that bucket b covers the current target note j. In particular, we use soft targets, meaning that the probability for one note is shared between the two buckets closest to the note's true pixel location x_j. We linearly interpolate the shared probabilities based on the two pixel distances (normalized to sum up to one) of the note's location x_j to the respective (closest) bucket centers. Bucket centers are denoted by c_b in the following where subscript b is the index of the respective bucket. Figure 3 shows an example sketch of the components described above. Based on the soft target vectors we design the output layer of our audio-to-sheet matching network as a B-way soft-max with activations defined as:

$$\phi(y_{j,b}) = \frac{e^{y_{j,b}}}{\sum_{k=1}^{B} e^{y_{j,k}}} \tag{1}$$

$\phi(y_{j,b})$ is the soft-max activation of the output neuron representing bucket b and hence also representing the region in the sheet image covered by this bucket. By applying the soft-max activation the network output gets normalized to range $(0, 1)$ and further sums up to 1.0 over all B output neurons. The network output can now also be interpreted as a vector of probabilities $\mathbf{p}_j = \{\phi(y_{j,b})\}$ and shares the same value range and properties as the soft target vectors.

In training, we optimize the network parameters Θ by minimizing the Categorical Cross Entropy (CCE) loss l_j between target vectors $\mathbf{t}_j$ and network output $\mathbf{p}_j$:

$$l_j(\Theta) = -\sum_{k=1}^{B} t_{j,k}\, log(p_{j,k}) \tag{2}$$

The CCE loss function becomes minimal when the network output $\mathbf{p}_j$ exactly matches the respective soft target vector $\mathbf{t}_j$. In Section 3.4 we provide further information on the exact optimization strategy used. [1]

[1] *For the sake of completeness*: In our initial experiments we started to predict the sheet location of audio snippets by minimizing the Mean-Squared-Error (MSE) between the predicted and the true pixel coordinate (MSE regression). However, we observed that training these networks is much harder and further performs worse than the bucket classification approach proposed in this paper.

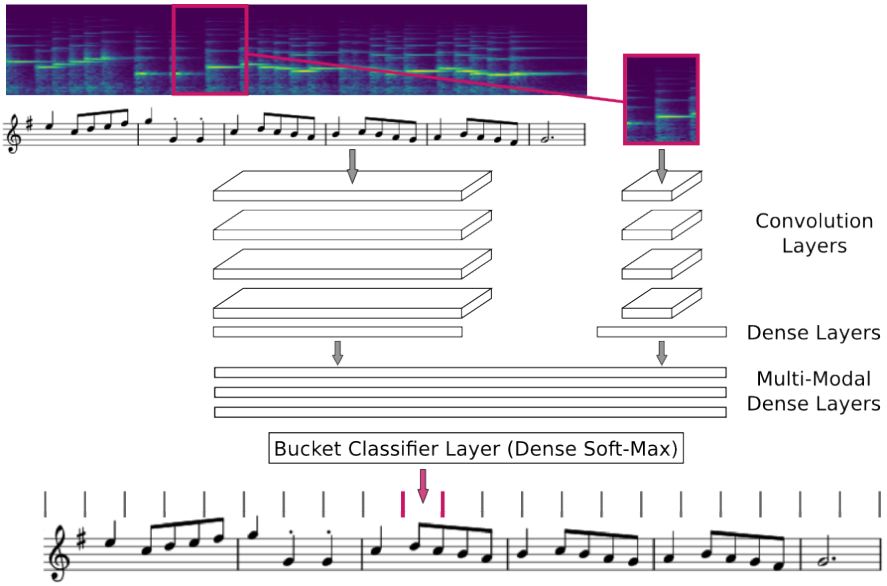


Figure 2: Overview of multi-modal convolutional neural network for audio-to-sheet matching. The network takes a staff image and a spectrogram excerpt as input. Two specialized convolutional network parts, one for the sheet image and one for the audio input, are merged into one multi-modality network. The output part of the network predicts the region in the sheet image – the classification bucket – to which the audio snippet corresponds.

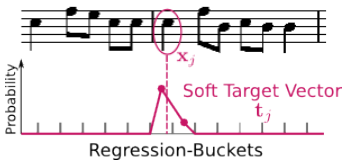


Figure 3: Part of a staff of sheet music along with soft target vector $\mathbf{t}_j$ for target note j surrounded with an ellipse. The two buckets closest to the note share the probability (indicated as dots) of containing the note. The short vertical lines highlight the bucket borders.

2.3 Sheet Location Prediction

Once the model is trained, we use it at test time to predict the expected location $\hat{x}_j$ of an audio snippet with target note j in a corresponding image of sheet music. The output of the network is a vector $\mathbf{p}_j = \{p_{j,b}\}$ holding the probabilities that the given test snippet j matches with bucket b in the sheet image. Having these probabilities we consider two different types of predictions: (1) We compute the center c_b^* of bucket $b^* = \text{argmax}_b\, p_{j,b}$ holding the highest overall matching probability. (2) For the second case we take, in addition to b^*, the two neighbouring buckets $b^* - 1$ and $b^* + 1$ into account and compute a (linearly) probability weighted position prediction in the sheet image as

$$\hat{x}_j = \sum_{k \in \{b^*-1, b^*, b^*+1\}} w_k c_k \qquad (3)$$

where weight vector $\mathbf{w}$ contains the probabilities $\{p_{j,b^*-1}, p_{j,b^*}, p_{j,b^*+1}\}$ normalized to sum up to one and c_k are the center coordinates of the respective buckets.

3. EXPERIMENTAL EVALUATION

This section evaluates our audio-to-sheet matching model on a publicly available dataset. We describe the experimental setup, including the data and evaluation measures, the particular network architecture as well as the optimization strategy, and provide quantitative results.

3.1 Experiment Description

The aim of this paper is to show that it is feasible to learn correspondences between audio (spectrograms) and images of sheet music in an *end-to-end* neural network fashion, meaning that an algorithm learns the entire task purely from data, so that no hand crafted feature engineering is required. We try to keep the experimental setup simple and consider one staff of sheet music per train/test sample (this is exactly the setup drafted in Figure 2). To be perfectly clear, the task at hand is the following: For a given audio snippet, find its x-coordinate pixel position in a corresponding staff of sheet music. We further restrict the audio to monophonic music containing half, quarter and eighth notes but allow variations such as dotted notes, notes tied across bar lines as well as accidental signs.

3.2 Data

For the evaluation of our approach we consider the Nottingham [2] data set which was used, e.g., for piano transcription in [4]. It is a collection of midi files already split into train, validation and test tracks. To be suitable for audio-to-sheet matching we prepare the data set (midi files) as follows:

[2] www-etud.iro.umontreal.ca/~boulanni/icml2012

Sheet-Image 40×390	Spectrogram 136×40
5×5 Conv(pad-2, stride-1-2)-64-BN-ReLu	3×3 Conv(pad-1)-64-BN-ReLu
3×3 Conv(pad-1)-64-BN-ReLu	3×3 Conv(pad-1)-64-BN-ReLu
2×2 Max-Pooling + Drop-Out(0.15)	2×2 Max-Pooling + Drop-Out(0.15)
3×3 Conv(pad-1)-128-BN-ReLu	3×3 Conv(pad-1)-96-BN-ReLu
3×3 Conv(pad-1)-128-BN-ReLu	2×2 Max-Pooling + Drop-Out(0.15)
2×2 Max-Pooling + Drop-Out(0.15)	3×3 Conv(pad-1)-96-BN-ReLu
	2×2 Max-Pooling + Drop-Out(0.15)
Dense-1024-BN-ReLu + Drop-Out(0.3)	Dense-1024-BN-ReLu + Drop-Out(0.3)
Concatenation-Layer-2048	
Dense-1024-BN-ReLu + Drop-Out(0.3)	
Dense-1024-BN-ReLu + Drop-Out(0.3)	
B-way Soft-Max Layer	

Table 1: Architecture of Multi-Modal Audio-to-Sheet Matching Model: BN: Batch Normalization, ReLu: Rectified Linear Activation Function, CCE: Categorical Cross Entropy, Mini-batch size: 100

1. We select the first track of the midi files (right hand, piano) and render it as sheet music using Lilypond. [3]
2. We annotate the sheet coordinate x_j of each note.
3. We synthesize the midi-tracks to *flac*-audio using Fluidsynth [4] and a *Steinway* piano sound font.
4. We extract the audio timestamps of all note onsets.

As a last preprocessing step we compute *log-spectrograms* of the synthesized flac files [3], with an audio sample rate of 22.05kHz, FFT window size of 2048 samples, and computation rate of 31.25 frames per second. For dimensionality reduction we apply a normalized 24-band logarithmic filterbank allowing only frequencies from 80Hz to 8kHz. This results in 136 frequency bins.

We already showed a spectrogram-to-sheet annotation example in Figure 1a. In our experiment we use spectrogram excerpts covering 1.2 seconds of audio (40 frames). This context is kept the same for training and testing. Again, annotations are aligned in a way so that the rightmost onset in a spectrogram excerpt corresponds to the pixel position of target note j in the sheet image. In addition, the spectrogram is shifted 5 frames to the right to also contain some information on the current target note's onset and pitch. We chose this annotation variant with the rightmost onset as it allows for an online application of our audio-to-sheet model (as would be required, e.g., in a score following task).

3.3 Evaluation Measures

To evaluate our approach we consider, for each test note j, the following ground truth and prediction data: (1) The true position x_j as well as the corresponding target bucket b_j (see Figure 3). (2) The estimated sheet location $\hat{x}_j$ and the most likely target bucket b^* predicted by the model. Given this data we compute two types of evaluation measures.

The first – the *top-k bucket hit rate* – quantifies the ratio of notes that are classified into the correct bucket allowing a tolerance of $k-1$ buckets. For example, the *top-1 bucket hit rate* counts only those notes where the predicted bucket b^* matches exactly the note's target bucket b_j. The *top-2 bucket hit rate* allows for a tolerance of one bucket and so on. The second measure – the *normalized pixel distance* – captures the actual distance of a predicted sheet location $\hat{x}_j$ to its corresponding true position x_j. To allow for an evaluation independent of the image resolution used in our experiments we normalize the pixel errors by dividing them by the width of the sheet image as $(\hat{x}_j - x_j)/width(\mathbf{S}_i)$. This results in distance errors living in range $(-1, 1)$.

We would like to emphasise that the quantitative evaluations based on the measures introduced above are performed only at time steps where a note onset is present. At those points in time an explicit correspondence between spectrogram (onset) and sheet image (note head) is established. However, in Section 4 we show that a time-continuous prediction is also feasible with our model and onset detection is not required at run time.

3.4 Model Architecture and Optimization

Table 1 gives details on the model architecture used for our experiments. As shown in Figure 2, the model is structured into two disjoint convolutional networks where one considers the sheet image and one the spectrogram (audio) input. The convolutional parts of our model are inspired by the VGG model built from sequences of small convolution kernels (e.g. 3×3) and max-pooling layers. The central part of the model consists of a concatenation layer bringing the image and spectrogram sub-networks together. After two dense layers with 1024 units each we add a B-way soft-max output layer. Each of the B soft-max output neurons corresponds to one of the disjoint buckets which in turn represent quantised sheet image positions. In our experiments we use a fixed number of 40 buckets selected as follows: We measure the minimum distance between two subsequent notes – in our sheet renderings – and select the number of buckets such that each bucket contains at most one note. It is of course possible that no note is present in a bucket – e.g., for the buckets covering the clef at the

[3] http://www.lilypond.org/
[4] http://www.fluidsynth.org/

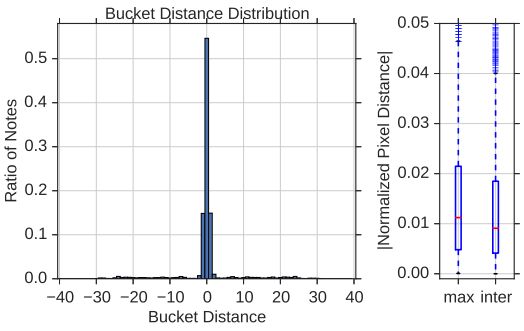


Figure 4: Summary of matching results on test set. *Left*: Histogram of bucket distances between predicted and true buckets. *Right*: Box-plots of absolute *normalized pixel distances* between predicted and true image position. The box-plot is shown for both location prediction methods described in Section 2.3 (maximum, interpolated).

	Train	**Valid**	**Test**
Top-1-Bucket-Hit-Rate	79.28%	51.63%	54.64%
Top-2-Bucket-Hit-Rate	94.52%	82.55%	84.36%
mean($\|NPD_{max}\|$)	0.0316	0.0684	0.0647
mean($\|NPD_{int}\|$)	0.0285	0.0670	0.0633
median($\|NPD_{max}\|$)	0.0067	0.0119	0.0112
median($\|NPD_{int}\|$)	0.0033	0.0098	0.0091
$\|NPD_{max}\| < w_b$	93.87%	76.31%	79.01%
$\|NPD_{int}\| < w_b$	94.21%	78.37%	81.18%

Table 2: *Top-k bucket hit rates* and *normalized pixel distances* (NPD) as described in Section 3.4 for train, validation and test set. We report mean and median of the absolute NPDs for both interpolated (int) and maximum (max) probability bucket prediction. The last two rows report the percentage of predictions not further away from the true pixel location than the width w_b of one bucket.

beginning of a staff. As activations function for the inner layers we use rectified linear units [10] and apply batch normalization [11] after each layer as it helps training and convergence.

Given this architecture and data we optimize the parameters of the model using mini-batch stochastic gradient descent with Nesterov style momentum. We set the batch size to 100 and fix the momentum at 0.9 for all epochs. The initial learn-rate is set to 0.1 and divided by 10 every 10 epochs. We additionally apply a weight decay of 0.0001 to all trainable parameters of the model.

3.5 Experimental Results

Figure 4 shows a histogram of the signed bucket distances between predicted and true buckets. The plot shows that more than 54% of all unseen test notes are matched exactly with the corresponding bucket. When we allow for a tolerance of ± 1 bucket our model is able to assign over 84% of the test notes correctly. We can further observe that the prediction errors are equally distributed in both directions – meaning too early and too late in terms of audio. The results are also reported in numbers in Table 2, as the top-k bucket hit rates for train, validation and test set.

The box plots in the right part of Figure 4 summarize the absolute *normalized pixel distances (NPD)* between predicted and true locations. We see that the probability-weighted position interpolation (Section 2.3) helps improve the localization performance of the model. Table 2 again puts the results in numbers, as means and medians of the absolute NPD values. Finally, Fig. 2 (bottom) reports the ratio of predictions with a pixel distance smaller than the width of a single bucket.

4. DISCUSSION AND REAL MUSIC

This section provides a representative prediction example of our model and uses it to discuss the proposed approach. In the second part we then show a first step towards matching *real* (though still very simple) music to its corresponding sheet. By *real music* we mean audio that is not just synthesized midi, but played by a human on a piano and recorded via microphone.

4.1 Prediction Example and Discussion

Figure 5 shows the image of one staff of sheet music along with the predicted as well as the ground truth pixel location for a snippet of audio. The network correctly matches the spectrogram with the corresponding pixel location in the sheet image. However, we observe a second peak in the bucket prediction probability vector. A closer look shows that this is entirely reasonable, as the music is quite repetitive and the current target situation actually appears twice in the score. The ability of predicting probabilities for multiple positions is a desirable and important property, as repetitive structures are immanent to music. The resulting prediction ambiguities can be addressed by exploiting the temporal relations between the notes in a piece by methods such as dynamic time warping or probabilistic models. In fact, we plan to combine the probabilistic output of our matching model with existing score following methods, as for example [2]. In Section 2 we mentioned that training a sheet location prediction with MSE-regression is difficult to optimize. Besides this technical drawback it would not be straightforward to predict a variable number of locations with an MSE-model, as the number of network outputs has to be fixed when designing the model.

In addition to the network inputs and prediction Fig. 5 also shows a *saliency map* [19] computed on the input sheet image with respect to the network output.[5] The saliency can be interpreted as the input regions to which most of the net's attention is drawn. In other words, it highlights the regions that contribute most to the current output produced by the model. A nice insight of this visualization is that the network actually focuses and recognizes the heads of the individual notes. In addition it also directs some attention to the style of stems, which is necessary to distinguish for example between quarter and eighth notes.

[5] The implementation is adopted from an example by Jan Schlüter in the recipes section of the deep learning framework *Lasagne* [7].

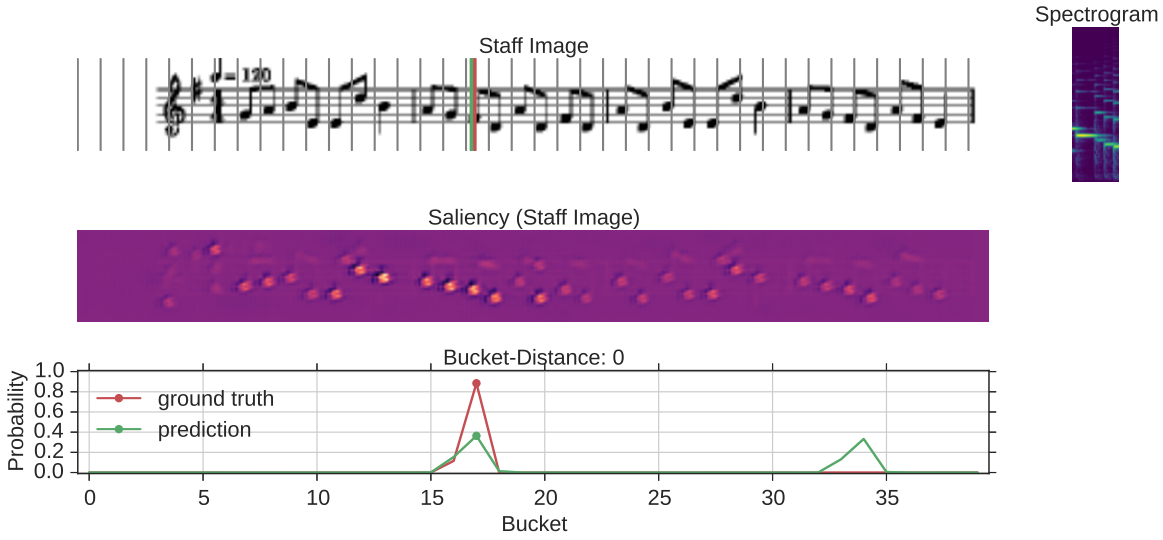


Figure 5: Example prediction of the proposed model. The top row shows the input staff image $\mathbf{S}_i$ along with the bucket borders as thin gray lines, and the given query audio (spectrogram) snippet $\mathbf{E}_{i,j}$. The plot in the middle visualizes the salience map (representing the attention of the neural network) computed on the input image. Note that the network's attention is actually drawn to the individual note heads. The bottom row compares the ground truth bucket probabilities with the probabilities predicted by the network. In addition, we also highlight the corresponding true and predicted pixel locations in the staff image in the top row.

The optimization on soft target vectors is also reflected in the predicted bucket probabilities. In particular the neighbours of the bucket with maximum activation are also active even though there is no explicit neighbourhood relation encoded in the soft-max output layer. This helps the interpolation of the true position in the image (see Fig. 4).

4.2 First Steps with Real Music

As a final point, we report on first attempts at working with "real" music. For this purpose one of the authors played the right hand part of a simple piece (Minuet in G Major by Johann Sebastian Bach, BWV Anhang 114) – which, of course, was not part of the training data – on a *Yamaha AvantGrand N2* hybrid piano and recorded it using a single microphone. In this application scenario we predict the corresponding sheet locations not only at times of onsets but for a continuous audio stream (subsequent spectrogram excerpts). This can be seen as a simple version of online score following in sheet music, without taking into account the temporal relations of the predictions. We offer the reader a video [6] that shows our model following the first three staff lines of this simple piece. [7] The ratio of predicted notes having a pixel-distance smaller than the bucket width (compare Section 3.5) is 71.72% for this real recording. This corresponds to a average normalized-pixel-distance of 0.0402.

[6] https://www.dropbox.com/s/0nz540i1178hjp3/Bach_Minuet_G_Major_net4b.mp4?dl=0

[7] Note: our model operates on single staffs of sheet music and requires a certain context of spectrogram frames for prediction (in our case 40 frames). For this reason it cannot provide a localization for the first couple of notes in the beginning of each staff at the current stage. In the video one can observe that prediction only starts when the spectrogram in the top right corner has grown to the desired size of 40 frames. We kept this behaviour for now as we see our work as a proof of concept. The issue can be easily addressed by concatenating the images of subsequent staffs in horizontal direction. In this way we will get a "continuous stream of sheet music" analogous to a spectrogram for audio.

5. CONCLUSION

In this paper we presented a multi-modal convolutional neural network which is able to match short snippets of audio with their corresponding position in the respective image of sheet music, without the need of any symbolic representation of the score. First evaluations on simple piano music suggest that this is a very promising new approach that deserves to be explored further.

As this is a proof of concept paper, naturally our method still has some severe limitations. So far our approach can only deal with monophonic music, notated on a single staff, and with performances that are roughly played in the same tempo as was set in our training examples.

In the future we will explore options to lift these limitations one by one, with the ultimate goal of making this approach applicable to virtually any kind of complex sheet music. In addition, we will try to combine this approach with a score following algorithm. Our vision here is to build a score following system that is capable of dealing with any kind of classical sheet music, out of the box, with no need for data preparation.

6. ACKNOWLEDGEMENTS

This work is supported by the Austrian Ministries BMVIT and BMWFW, and the Province of Upper Austria via the COMET Center SCCH, and by the European Research Council (ERC Grant Agreement 670035, project CON ESPRESSIONE). The Tesla K40 used for this research was donated by the NVIDIA corporation.

7. REFERENCES

[1] Andreas Arzt, Harald Frostel, Thassilo Gadermaier, Martin Gasser, Maarten Grachten, and Gerhard Widmer. Artificial intelligence in the concertgebouw. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Buenos Aires, Argentina, 2015.

[2] Andreas Arzt, Gerhard Widmer, and Simon Dixon. Automatic page turning for musicians via real-time machine listening. In *Proc. of the European Conference on Artificial Intelligence (ECAI)*, Patras, Greece, 2008.

[3] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. madmom: a new Python Audio and Music Signal Processing Library. *arXiv:1605.07008*, 2016.

[4] Nicolas Boulanger-lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1159–1166, 2012.

[5] Arshia Cont. A coupled duration-focused architecture for realtime music to score alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):837–846, 2009.

[6] Nicholas Cook. Performance analysis and chopin's mazurkas. *Musicae Scientae*, 11(2):183–205, 2007.

[7] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Eric Battenberg, Aäron van den Oord, et al. Lasagne: First release., August 2015.

[8] Zhiyao Duan and Bryan Pardo. A state space model for on-line polyphonic audio-score alignment. In *Proc. of the IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, 2011.

[9] Jon W. Dunn, Donald Byrd, Mark Notess, Jenn Riley, and Ryan Scherle. Variations2: Retrieving and using music in an academic setting. *Communications of the ACM, Special Issue: Music information retrieval*, 49(8):53–48, 2006.

[10] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[12] Özgür İzmirli and Gyanendra Sharma. Bridging printed music and audio through alignment using a mid-level score representation. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, Porto, Portugal, 2012.

[13] Mark S. Melenhorst, Ron van der Sterren, Andreas Arzt, Agustín Martorell, and Cynthia C. S. Liem. A tablet app to enrich the live and post-live experience of classical concerts. In *Proceedings of the 3rd International Workshop on Interactive Content Consumption (WSICC) at TVX 2015*, 06/2015 2015.

[14] Marius Miron, Julio José Carabias-Orti, and Jordi Janer. Audio-to-score alignment at note level for orchestral recordings. In *Proc. of the International Conference on Music Information Retrieval (ISMIR)*, Taipei, Taiwan, 2014.

[15] Meinard Müller, Frank Kurth, and Michael Clausen. Audio matching via chroma-based statistical features. In *Proc. of the International Society for Music Information Retrieval Conference (ISMIR)*, London, Great Britain, 2005.

[16] Bernhard Niedermayer and Gerhard Widmer. A multipass algorithm for accurate audio-to-score alignment. In *Proc. of the International Society for Music Information Retrieval Conference (ISMIR)*, Utrecht, The Netherlands, 2010.

[17] Matthew Prockup, David Grunberg, Alex Hrybyk, and Youngmoo E. Kim. Orchestral performance companion: Using real-time audio to score alignment. *IEEE Multimedia*, 20(2):52–60, 2013.

[18] Christopher Raphael. Music Plus One and machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

[19] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv:1412.6806*, 2014.

[20] Verena Thomas, Christian Fremerey, Meinard Müller, and Michael Clausen. Linking Sheet Music and Audio - Challenges and New Approaches. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing*, volume 3 of *Dagstuhl Follow-Ups*, pages 1–22. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2012.

EXTRACTING GROUND TRUTH INFORMATION FROM MIDI FILES: A MIDIFESTO

Colin Raffel and Daniel P. W. Ellis
LabROSA
Department of Electrical Engineering
Columbia University
New York, NY

ABSTRACT

MIDI files abound and provide a bounty of information for music informatics. We enumerate the types of information available in MIDI files and describe the steps necessary for utilizing them. We also quantify the reliability of this data by comparing it to human-annotated ground truth. The results suggest that developing better methods to leverage information present in MIDI files will facilitate the creation of MIDI-derived ground truth for audio content-based MIR.

1. MIDI FILES

MIDI (Music Instrument Digital Interface) is a hardware and software standard for communicating musical events. First proposed in 1983 [1], MIDI remains a highly pervasive standard both for storing musical scores and communicating information between digital music devices. Its use is perhaps in spite of its crudeness, which has been lamented since MIDI's early days [21]; most control values are quantized as 7-bit integers and information is transmitted at the relatively slow (by today's standards) 31,250 bits per second. Nevertheless, its efficiency and well-designed specification make it a convenient way of formatting digital music information.

In the present work, we will focus on MIDI files, which in a simplistic view can be considered a compact way of storing a musical score. MIDI files are specified by an extension to the MIDI standard [2] and consist of a sequence of MIDI messages organized in a specific format. A typical MIDI file contains timing and meter information in addition to a collection of one or more "tracks", each of which contains a sequence of notes and control messages. The General MIDI standard [3] further specifies a collection of 128 instruments on which the notes can be played, which standardizes the playback of MIDI files and has therefore been widely adopted.

When paired with a General MIDI synthesizer, MIDI files have been used as a sort of *semantic* audio codec, with entire songs only requiring a few kilobytes of storage. The early availability of this "coding method", combined with the expense of digital storage in the 90s, made MIDI files a highly pervasive method of storing and playing back songs before the advent of the MP3. Even after high-quality perceptual audio codecs were developed and storage prices plummeted, MIDI files remained in use in resource-scarce settings such as karaoke machines and cell phone ringtones. As a result, there is an abundance of MIDI file transcriptions of music available today; through a large-scale web scrape, we obtained 178,561 MIDI files with unique MD5 checksums. Given their wide availability, we believe that MIDI files are underutilized in the Music Information Retrieval community.

In this paper, we start by outlining the various sources of information present in MIDI files and reference relevant works which utilize them in Section 2. In Section 3, we discuss the steps needed to leverage MIDI-derived information as ground truth for content-based MIR. We then establish a baseline for the reliability of MIDI-derived ground truth by comparing it to handmade annotations in Section 4. Finally, in Section 5, we argue that improving the process of extracting information from MIDI files is a viable path for creating large amounts of ground truth data for MIR.

2. INFORMATION AVAILABLE IN MIDI FILES

While various aspects of MIDI files have been used in MIR research, to our knowledge there has been no unified overview of the information they provide, nor a discussion of the availability and reliability of this information in MIDI transcriptions found "in the wild". We therefore present an enumeration of the different information sources in a typical MIDI file and discuss their applicability to different MIR tasks. Because not all MIDI files are created equal, we also computed statistics about the presence and quantity of each information source across our collection of 178,561 MIDI files; the results can be seen in Figure 1 and will be discussed in the following sections.

2.1 Transcription

MIDI files are specified as a collection of "tracks", where each track consists of a sequence of MIDI events on one of 16 channels. Commonly used MIDI events are note-on and note-off messages, which together specify the start

© Colin Raffel and Daniel P. W. Ellis. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Colin Raffel and Daniel P. W. Ellis. "Extracting Ground Truth Information from MIDI Files: A MIDIfesto", 17th International Society for Music Information Retrieval Conference, 2016.

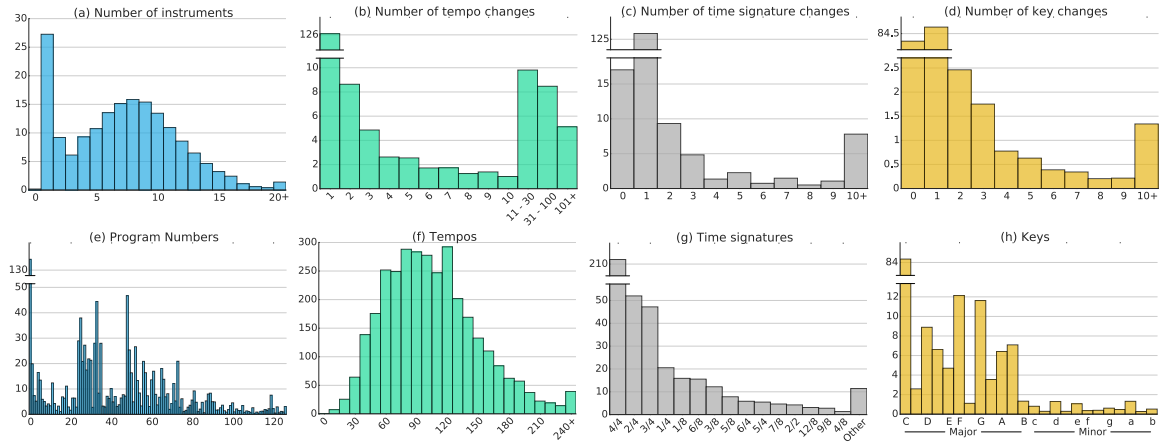


Figure 1: Statistics about sources of information in 178,561 unique MIDI files scraped from the internet. Histograms in the top row show the number of MIDI files which had a given number of events for different event types; in the bottom row, we show distributions of the different values set by these events across all MIDI files. All counts are reported in thousands. For example, about 125,000 MIDI files had a single time signature change event, and about 210,000 4/4 time signature changes were found in all of our MIDI files.

and end time of notes played at a given pitch on a given channel. Various control events also exist, such as pitch bends, which allow for finer control of the playback of the MIDI file. Program change events determine which instrument these events are sent to. The General MIDI standard defines a correspondence between program numbers and a predefined list of 128 instruments. General MIDI also specifies that all notes occurring on MIDI channel 10 play on a separate percussion instrument, which allows for drum tracks to be transcribed. The distribution of the total number of program change events (corresponding to the number of instruments) across the MIDI files in our collection and the distribution of these program numbers are shown in Figures 1(a) and 1(e) respectively. The four most common program numbers (shown as the four tallest bars in Figure 1(e)) were 0 ("Acoustic Grand Piano"), 48 ("String Ensemble 1"), 33 ("Electric Bass (finger)"), and 25 ("Acoustic Guitar (steel)").

This specification makes MIDI files naturally suited to be used as transcriptions of pieces of music, due to the fact that they can be considered a sequence of notes played at different "velocities" (intensities) on a collection of instruments. As a result, many MIDI files are transcriptions and are thus commonly used as training data for automatic transcription systems (see [32] for an early example). This type of data also benefits score-informed source separation methods, which utilize the score as a prior to improve source separation quality [15]. An additional natural use of this information is for "instrument activity detection", i.e. determining when certain instruments are being played over the course of a piece of music. Finally, the enumeration of note start times lends itself naturally to onset detection, and so MIDI data has been used for this task [4].

2.2 Music-Theoretic Features

Because many MIDI files are transcriptions of music, they can also be used to compute high-level musicological characteristics of a given piece. Towards this end, the software library `jSymbolic` [20] includes functionality to extract a wide variety of features, including instrumentation, rhythm, and pitch statistics. Similarly, `music21` [9] provides a general-purpose framework for analyzing collections of digital scores (including MIDI files). Computing these features on a collection of MIDI transcriptions is valuable for computational musicology and can enable data-driven corpus studies. For example, [10] discusses the use of `music21` and `jSymbolic` to extract features from scores and uses them to distinguish music from different composers and musical traditions.

2.3 Meter

Timing in MIDI files is determined by two factors: The MIDI file's specified "resolution" and tempo change events. Each event within the MIDI file specifies the number of "ticks" between it and the preceding event. The resolution, which is stored in the MIDI file's header, sets the number of ticks which correspond to a single beat. The amount of time spanned by each tick is then determined according to the current tempo, as set by tempo change events. For example, if a MIDI file has a resolution of 220 ticks per beat and the current tempo is 120 beats per minute,[1] each tick would correspond to $60/(120 * 220) = 0.002\overline{27}$ seconds. If a MIDI event in this file is specified to occur 330 ticks after the previous event, then it would occur $330 * 0.002\overline{27} = .75$ seconds later.

The timing in a MIDI file can vary over time by including many tempo change events. In practice, as shown in Figure 1(b), most MIDI files only contain a single tempo change and are therefore transcribed at a fixed tempo. However, there are many MIDI files in our collection which have a large number of tempo change events (as indicated by the rightmost bars in Figure 1(b)). We have found that this is a common practice for making the timing of a MIDI transcription closely match that of an audio

[1] Actually, tempo change events specify the number of microseconds per quarter beat, but this can be readily converted to beats per minute.

recording of the same song. Despite the fact that the default tempo for a MIDI file is 120 beats per minute, Figure 1(f) demonstrates that a wide range of tempos are used. In practice, we find that this is due to the fact that even when a single tempo event is used, it is often set so that the MIDI transcription's tempo approximates that of an audio recording of the same song.

Time signature change events further augment MIDI files with the ability to specify time signatures, and are also used to indicate the start of a measure. By convention, MIDI files have a time signature change at the first tick, although this is not a requirement. Because time signature changes are relatively rare in western popular music, the vast majority of the MIDI files in our collection contain a single time signature change, as seen in Figure 1(c). Despite the fact that 4/4 is the default time signature for MIDI files and is pervasive in western popular music, a substantial portion (about half) of the time signature changes were not 4/4, as shown in Figure 1(g).

Because MIDI files are required to include tempo information in order to specify their timing, it is straightforward to extract beat locations from a MIDI file. By convention, the first (down)beat in a MIDI transcription occurs at the first tick. Determining the beat locations in a MIDI file therefore involves computing beat locations starting from the first tick and adjusting the tempo and time signature according to any tempo change or time signature change events found. Despite this capability, to our knowledge MIDI files have not been used as ground truth for beat tracking algorithms. However, [19] utilized a large dataset of MIDI files to study drum patterns using natural language processing techniques.

2.4 Key

An additional useful event in MIDI files is the key change event. Any of the 24 major or minor keys may be specified. Key changes simply give a suggestion as to the tonal content and do not affect playback, and so are a completely optional meta-event. As seen in Figure 1(d), this results in many MIDI files omitting key change events altogether. A further complication is that a disproportionate number (about half) of the key changes in the MIDI files in our collection were C major, as shown in Figure 1(h). This disagrees with corpus studies of popular music, e.g. [8] which found that only about 26% of songs from the Billboard 100 were in C major. We believe this is because many MIDI transcription software packages automatically insert a C major key change at the beginning of the file.

2.5 Lyrics

Lyrics can be added to MIDI transcriptions by the use of lyrics meta-events, which allow for timestamped text to be included over the course of the song. This capability enables the common use of MIDI files for karaoke; in fact, a separate file extension ".kar" is often used for MIDI files which include lyrics meta-events. Occasionally, the generic text meta-event is also used for lyrics, but this is not its intended use. In our collection, we found 23,801 MIDI files (about 13.3%) which had at least one lyrics meta-event.

2.6 What's Missing

Despite the wide variety of information sources available in MIDI files outlined in the previous sections, there are various types of information which are not possible (or not common) to store in MIDI files. While the General MIDI specification includes the vocal instruments "Choir Aahs", "Voice Oohs", "Synth Choir", "Lead 6 (voice)" and "Pad 4 (choir)", in practice there is no specific program number (or numbers) which is consistently used to transcribe vocals. As a result, in a given MIDI file there is no reliable way of determining which instrument is a transcription of the vocals in a song. Furthermore, because a substantial portion of MIDI files were designed for karaoke, the vocals may not be transcribed at all.

While the MIDI specification does include "track name", "program name", and "instrument name" meta-events, they are not standardized and so are not used consistently. It follows that there is no simple way to retrieve the "melody" from a MIDI transcription, although the fact that all instruments are transcribed separately can make its estimation more straightforward than for audio files. For example, [31] explores the use of simple features such as the average velocity and note range within a track to predict whether it is a melody, and also finds that in a small dataset the track name reliably indicates a melody track 44.3% of the time. Similarly, [23] uses heuristic features and a random forest classifier to predict with high accuracy whether a track is a melody.

There is also no explicit way for MIDI files to include chord labels or structural segmentation boundaries (e.g. "verse", "chorus", "solo"). While this would in principle be possible thanks to the generic MIDI "text" meta-event, we have yet to find any MIDI files which store this information. Nevertheless, estimating chords in particular is greatly facilitated by the presence of a ground truth transcription. Both `music21` [9] and `melisma` [30] include functionality for estimating chord sequences from symbolic data. Rhodes et al. [29] also proposed a symbolic chord estimation method using Bayesian Model Selection, which was shown to outperform `melisma` on a dataset of Beatles MIDI files in [14].

While text meta-events could also be used to store song-level metadata (song title, artist name, etc.) in a MIDI file, we seldom encountered this. There is no standardized way to store this metadata in a MIDI file, although we found that a minority of the filenames in our collection indicated the song title and occasionally the artist name. The lack of a metadata specification also inhibits the attribution of MIDI transcriptions to the person who transcribed them.

3. UTILIZING MIDI FILES AS GROUND TRUTH

Utilizing MIDI files as ground truth information for audio content-based MIR tasks requires the following: First, the compact low-level binary format used by MIDI files must be parsed so that the information can be readily extracted. Second, the artist and song of a MIDI file must be determined so it can be paired with a corresponding audio recording. Finally, for many uses, the MIDI file must be aligned in time with its matching audio.

3.1 Extracting Information

The information sources enumerated in Section 2 are not readily available from MIDI files due to fact that they follow a low-level binary protocol. For example, in order to extract the time (in seconds) of all onsets from a given instrument in a MIDI file, note events which occur on the same track and channel as program change events for the instrument must be collected and their timing must be computed from their relative ticks using the global tempo change events. Fortunately, various software libraries have been created to facilitate this process. `pretty_midi` [24] simplifies the extraction of useful information from MIDI transcriptions by taking care of most of the low-level parsing needed to convert the information to a more human-friendly format. It contains functions for retrieving beats, onsets, and note lists from specific instruments, and the times and values of key, tempo, and time signature changes. It also can be used to modify MIDI files, as well as to convert them to synthesized audio or a spectrogram-like piano roll representation. The aforementioned `jSymbolic` contains an extensive collection of routines for computing musicological features from MIDI files. Finally, both `music21` and `melisma` are capable of inferring high-level music information from symbolic data of various types, including MIDI.

3.2 Matching

Apart from metadata-agnostic corpus studies such as [19], determining the song a given MIDI file represents is usually required. Matching a given MIDI file to, for example, a corresponding entry in the Million Song Dataset [5] can be beneficial even in experiments solely involving symbolic data analysis because it can provide additional metadata for the track including its year, genre, and user-applied tags. Utilizing information in a MIDI file for ground truth in audio content-based MIR tasks further requires that it be matched to an audio recording of the song, but this is made difficult by the lack of a standardized method for storing song-level metadata in MIDI files (as discussed in Section 2.6). Content-based matching offers a solution; for example, early work by Hu et al. [17] assigned matches by finding the smallest dynamic time warp (DTW) distance between spectrograms of MIDI syntheses and audio files across a corpus. This approach is prohibitively slow for very large collections of MIDI and/or audio files, so [25] explored learning a mapping from spectrograms to downsampled sequences of binary vectors, which greatly accelerates DTW. [27] provided further speed-up by mapping entire spectrograms to fixed-length vectors in a Euclidean space where similar songs are mapped close together. These methods make it feasible to match a MIDI file against an extremely large corpus of music audio.

3.3 Aligning

There is no guarantee that a MIDI transcription for a given song was transcribed so that its timing matches an audio recording of a performance of the song. For the many types of ground truth data that depend on timing (e.g. beats, note transcription, or lyrics), the MIDI file must therefore have its timing adjusted so that it matches that of the performance. Fortunately, score-to-audio alignment, of which MIDI-to-audio alignment is a special "offline" case, has received substantial research attention. A common method is to use DTW or another edit-distance measure to find the best alignment between spectrograms of the synthesized MIDI and the audio recording; see [26] or [14] for surveys.

In practice, audio-to-MIDI alignment systems can fail when there are overwhelming differences in timing or deficiencies in the transcription, e.g. missing or incorrect notes or instruments. Ideally, the alignment and matching processes would automatically report the success of the alignment and the quality of the MIDI transcription. [26] explores the ability of DTW-based alignment systems to report a "confidence" score indicating the success of the alignment. We do not know of any research into automatically determining the quality of a MIDI transcription.

4. MEASURING A BASELINE OF RELIABILITY FOR MIDI-DERIVED INFORMATION

Given the potential availability of ground truth information in MIDI transcriptions, we wish to measure the reliability of MIDI transcriptions found "in the wild". A straightforward way to evaluate the quality of MIDI-derived annotations is to compare them with hand-made annotations for the same songs. Given a MIDI transcription and human-generated ground truth data, we can extract corresponding information from the MIDI file and compare using the evaluation metrics employed in the Music Information Retrieval Evaluation eXchange (MIREX) [12]. We therefore leveraged the Isophonics Beatles annotations [18] as a source of ground truth to compare against MIDI-derived information. MIDI transcriptions of these songs are readily available due to The Beatles' popularity.

Our choice in tasks depends on the overlap in sources of information in the Isophonics annotations and MIDI files. Isophonics includes beat times, song-level key information, chord changes, and structural segmentation. As noted in Section 2, beat times and key changes may be included in MIDI files but there is no standard way to include chord change or structural segmentation information. We therefore performed experiments to evaluate the quality of key labels and beat times available in MIDI files. Fortuitously, these two experiments give us an insight into both song-level timing-agnostic information (key) and alignment-dependent timing-critical information (beats). To carry out these experiments, we first manually identified 545 MIDI files from our collection which had filenames indicating that they were transcriptions of one of the 179 songs in the Isophonics Beatles collection; we found MIDI transcriptions for all but 11. The median number of MIDI transcriptions per song was 2; the song "Eleanor Rigby" had the most, with 14 unique transcriptions.

4.1 Key Experiment

In our first experiment, we evaluated the reliability of key change events in MIDI files. We followed the MIREX methodology for comparing keys [13], which proceeds as follows: Each song may only have a single key. All keys

Source	Score	Comparisons
MIDI, all keys	0.400	223
MIDI, C major only	0.167	146
MIDI, non-C major	0.842	77
QM Key Detector	0.687	151
`whatkeyisitin.com`	0.857	145

Table 1: Mean scores achieved, and the number of comparisons made, by different datasets compared to Isophonics Beatles key annotations.

must be either major or minor, e.g. "C# Major" and "E minor" are allowed but "D Mixolydian" is not. An estimated key is given a score of 1.0 when it exactly matches a ground truth key, 0.5 when it is a perfect fifth above the ground truth key, 0.3 when it is a relative major or minor, 0.2 when it is a parallel major or minor, and 0.0 otherwise. We utilized the evaluation library `mir_eval` [28] to compute this score.

The Isophonics annotations mostly follow this format, except that 21 songs contained multiple key annotations and 7 others contained non-major/minor keys. To simplify evaluation, we discarded these songs, leaving 151 ground truth key annotations. Of our 545 Beatles MIDIs, 221 had no key change event and 5 had more than one, which we also omitted from evaluation. This left 223 MIDI files for which we extracted key annotations and compared them to valid Isophonics annotations. Because of the preponderance of C major key change events noted in Section 2.4, we also evaluated MIDI-derived C Major and non-C major instances separately to see whether they were less reliable.

As a baseline, we also extracted keys using the QM Vamp Key Detector plugin [7] whose underlying algorithm is based on [22] which finds the key profile best correlated with the chromagram of a given song. This plugin achieved the highest score in MIREX 2013, and has been the only key detection algorithm submitted in 2014 and 2015. This gives us a reasonable expectation for a good audio content-based key estimator. To determine the extent to which human annotators agree on key labels, we also collected key annotations for Beatles' songs from `whatkeyisitin.com`. As with the Isophonics key annotations, some songs had multiple and/or modal key labels; we discarded these and ended up with 145 labels for songs in the Isophonics dataset.

The mean scores resulting from comparing each dataset to the Isophonics annotations can be seen in Table 1. At first glance, the mean score of 0.4 achieved by MIDI key change messages is discouraging. However, by omitting all MIDI files with C major key events (which achieved a mean score of 0.167), the mean score jumps to 0.842. This is comparable to the human baseline, and is substantially higher than the algorithmically estimated score. We therefore propose that *non-C major* MIDI key change events are as reliable as hand-annotated labels, but that C major key annotations in MIDI files are effectively useless.

4.2 Beat Experiment

Utilizing many of the sources of information in MIDI files depends on the precise alignment of a given MIDI file to an audio recording of a performance of the same song. We therefore performed an additional experiment to evaluate the quality of MIDI-derived beat annotations, which are evaluated on the scale of tens of milliseconds. Producing valid beat annotations from a MIDI file requires not only that the file's meter information is correct, but also that it has been aligned with high precision.

To align our Beatles MIDI files to corresponding audio recordings, we used the scheme proposed in [26], which was found by a large-scale search over common DTW-based audio-to-MIDI alignment systems. We give an outline of this method below; for a full description, see [26]. First, the MIDI file is synthesized using the `fluidsynth` program. Log-magnitude, constant-Q spectrograms of the synthesized MIDI and audio recording are extracted and their pairwise cosine distance matrix is computed. The lowest-cost path through the distance matrix is then found using DTW, with the constraint that the path must span at least 96% of the shorter of the two spectrograms. In addition, all paths are penalized by adding the median value of the distance matrix each time a frame in one spectrogram is mapped to multiple frames in the other. Finally, a "confidence score" is computed as the mean pairwise distance along the lowest-cost path, normalized by mean of the submatrix spanned by the path.

We followed [26] exactly, except for the following changes: First, instead of computing spectrograms with a hop size of 46 ms, we used 23 ms. This finer timescale is more appropriate for the beat evaluation metrics we will use, which have tolerances measured in tens of milliseconds. Second, the confidence scores computed using the method of [26] lie in the range $[0.5, 1.0]$ where 0.5 corresponds to "highly confident" and 1.0 corresponds to "likely wrong"; we mapped this linearly to a more easily-interpretable range of $[0.0, 1.0]$ where higher scores mean higher confidence.

We used `pretty_midi`'s `get_beats` method to extract beat times from our 545 Beatles MIDI files, and adjusted each beat's timing according to the MIDI file's alignment to corresponding audio recordings. For evaluation, we used the *F-measure*, *Any Metric Level Total*, and *Information Gain* metrics described in [11], as implemented in `mir_eval`. As a baseline, we also computed beat locations using the `DBNBeatTracker` from the `madmom` software library,[2] which is based on the algorithm from [6]. This represents a state-of-the-art general-purpose beat tracker which, on the Beatles data, can reliably produce high-quality annotations. If MIDI-derived beat annotations are to be taken as ground truth, they must achieve scores similar to or higher than the `DBNBeatTracker`.

We visualize the resulting scores in Figure 2. Because we don't expect beats to be extracted accurately from MIDI files that are poor transcriptions or when alignment failed, we plotted each MIDI file as a single point whose x coor-

[2] `https://github.com/CPJKU/madmom`

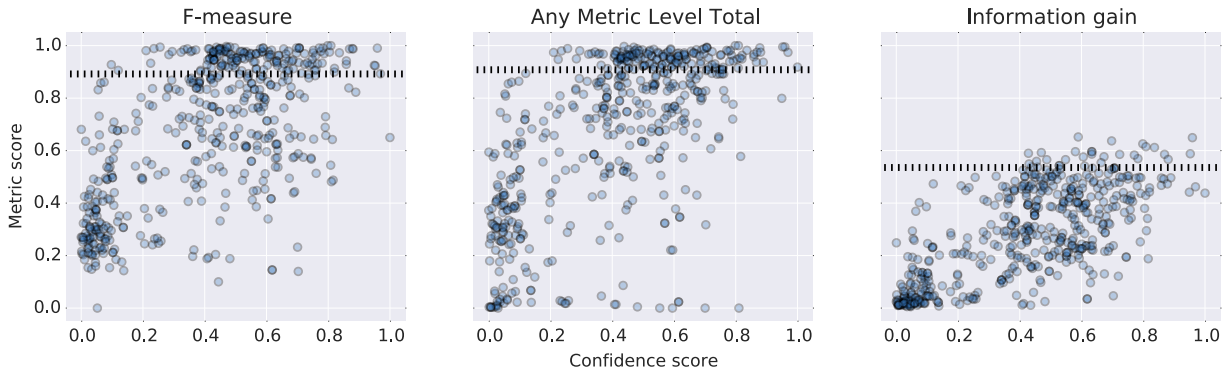


Figure 2: Beat evaluation metric scores (compared to Isophonics beat annotations) and alignment confidence scores achieved by different audio-to-MIDI alignments of Beatles MIDI files, with each shown as a blue dot. Mean scores for each metric achieved by the `DBNBeatTracker` [6] are shown as dashed lines.

dinate corresponds to the alignment confidence score and whose y coordinate is the resulting evaluation metric score achieved. Ideally, all points in these plots would be clustered in the bottom left (corresponding to failed alignments with low confidence scores) or top right (corresponding to a successful alignment and beat annotation extraction with a high confidence score). For reference, we plot the mean score achieved by the `DBNBeatTracker` as dotted lines for each metric. From these plots, we can see that in many cases, MIDI-derived annotations achieve near-perfect scores, particularly for the *F-Measure* and *Any Metric Level Total* metrics. However, there is no reliable correspondence between high confidence scores and high evaluation metric scores. For example, while it appears that a prerequisite for an accurate MIDI-derived beat annotation is a confidence score above .5, there are many MIDI files which had high confidence scores but low metric scores (appearing in the bottom-right corner of the plots in Figure 2).

We found that this undesirable behavior was primarily caused by a few issues: First, it is common that the alignment system would produce alignments which were slightly "sloppy", i.e. were off by one or two frames (corresponding to 23 milliseconds each) in places. This had less of an effect on the *F-measure* and *Any Metric Level Total* metrics, which are invariant to small temporal errors up to a certain threshold, but deflated the *Information Gain* scores because this metric rewards consistency even for fine-timing errors. Second, many MIDI files had tempos which were at a different metric level than the annotations (e.g. double, half, or a third of the tempo). This affected the *Any Metric Level Total* scores the least because it is invariant to these issues, except for the handful of files which were transcribed at a third of the tempo. Finally, we found that the confidence score produced by the alignment system is most reliable at producing a low score in the event of a total failure (indicated by points in the bottom left of the plots in Figure 2), but was otherwise insensitive to the more minor issues that can cause beat evaluation metrics to produce low scores.

5. DISCUSSION

Our results suggest that while MIDI files have the potential to be valuable sources of ground truth information, their usage may come with a variety of caveats. However, due to the enormous number of MIDI transcriptions available, we believe that developing better methods to leverage information present in MIDI files is a tantalizing avenue for obtaining more ground truth data for music information retrieval. For example, while C major key annotations cannot be trusted, developing a highly reliable C major vs. non-C major classification algorithm for symbolic data (which would ostensibly be much more tractable than creating a perfect general-purpose audio content-based key estimation algorithm) would enable the reliable usage of all key change messages in MIDI files. Further work into robust audio-to-MIDI alignment is also warranted in order to leverage timing-critical information, as is the neglected problem of alignment confidence score reporting. Novel questions such as determining whether all instruments have been transcribed in a given MIDI file would also facilitate their use as ground truth transcriptions. Fortunately, all of these tasks are made easier by the fact that MIDI files are specified in a format from which it is straightforward to extract pitch information. Any techniques developed towards this end could also be applied to other ubiquitous symbolic digital music formats such as MusicXML [16].

To facilitate further investigation, all 178,561 of the MIDI files we obtained in our web scrape (including our collection of 545 Beatles MIDIs) are available online, [3] as well as all of the code used in the experiments in this paper. [4] We hope this data and discussion facilitates a groundswell of MIDI utilization in the MIR community.

6. ACKNOWLEDGMENTS

We thank Eric J. Humphrey and Hunter McCurry for discussion about key evaluation, Rafael Valle for investigations into MIDI beat tracking, and our anonymous reviewers for their suggestions.

[3] `http://colinraffel.com/projects/lmd`
[4] `https://github.com/craffel/midi-ground-truth`

7. REFERENCES

[1] International MIDI Association. MIDI musical instrument digital interface specification 1.0. 1983.

[2] International MIDI Association. Standard MIDI files. 1988.

[3] International MIDI Association. General MIDI level 1 specification. 1991.

[4] Juan Pablo Bello, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, and Mark B. Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047, 2005.

[5] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 591–596, 2011.

[6] Sebastian Böck, Florian Krebs, and Gerhard Widmer. A multi-model approach to beat tracking considering heterogeneous music styles. In *Proceedings of the 15th International Society for Music Information Retrieval Conference*, pages 603–608, 2014.

[7] Chris Cannam, Emmanouil Benetos, Matthias Mauch, Matthew E. P. Davies, Simon Dixon, Christian Landone, Katy Noland, and Dan Stowell. MIREX 2015 entry: Vamp plugins from the centre for digital music. In *11th Music Information Retrieval Evaluation eXchange*, 2015.

[8] Dave Carlton. I analyzed the chords of 1300 popular songs for patterns. This is what I found. `http://www.hooktheory.com/blog/`, June 2012.

[9] Michael Scott Cuthbert and Christopher Ariza. `music21`: A toolkit for computer-aided musicology and symbolic music data. In *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pages 637–642, 2010.

[10] Michael Scott Cuthbert, Christopher Ariza, and Lisa Friedland. Feature extraction and machine learning on symbolic music using the `music21` toolkit. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 387–392, 2011.

[11] Matthew E. P. Davies, Norberto Degara, and Mark D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. Technical Report C4DM-TR-09-06, Queen Mary University of London, 2009.

[12] J. Stephen Downie. The music information retrieval evaluation exchange (2005-2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

[13] Andreas Ehmann, Kris West, Mert Bay, Kahyun Choi, and Yun Hao. MIREX task: Audio key detection. `http://music-ir.org/mirex/wiki/2016:Audio_Key_Detection`, 2016.

[14] Sebastian Ewert, Meinard Müller, Verena Konz, Daniel Müllensiefen, and Geraint A. Wiggins. Towards cross-version harmonic analysis of music. *IEEE Transactions on Multimedia*, 14(3):770–782, 2012.

[15] Sebastian Ewert, Bryan Pardo, Meinard Müller, and Mark Plumbley. Score-informed source separation for musical audio recordings: An overview. *IEEE Signal Processing Magazine*, 31(3):116–124, 2014.

[16] Michael Good. MusicXML for notation and analysis. In Walter B. Hewelett and Eleanor Selfridge-Field, editors, *The virtual score: representation, retrieval, restoration*, volume 12, pages 113–124. MIT Press, 2001.

[17] Ning Hu, Roger B. Dannenberg, and George Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 185–188, 2003.

[18] Matthias Mauch, Chris Cannam, Matthew Davies, Simon Dixon, Christopher Harte, Sefki Kolozali, Dan Tidhar, and Mark Sandler. OMRAS2 metadata project 2009. In *10th International Society for Music Information Retrieval Conference Late Breaking and Demo Papers*, 2009.

[19] Matthias Mauch and Simon Dixon. A corpus-based study of rhythm patterns. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, pages 163–168, 2012.

[20] Cory McKay and Ichiro Fujinaga. `jSymbolic`: A feature extractor for MIDI files. In *Proceedings of the International Computer Music Conference*, pages 302–5, 2006.

[21] F. Richard Moore. The dysfunctions of MIDI. *Computer music journal*, 12(1):19–28, 1988.

[22] Katy Noland and Mark Sandler. Signal processing parameters for tonality estimation. In *Audio Engineering Society Convention 122*, 2007.

[23] Pedro José Ponce de León Amador, José Manuel Iñesta Quereda, and David Rizo Valero. Mining digital music score collections: melody extraction and genre recognition. In Peng-Yeng Yin, editor, *Pattern Recognition Techniques, Technology and Applications*, chapter 25, pages 559–590. InTech, 2008.

[24] Colin Raffel and Daniel P. W. Ellis. Intuitive analysis, creation and manipulation of MIDI data with `pretty_midi`. In *15th International Society for Music Information Retrieval Conference Late Breaking and Demo Papers*, 2014.

[25] Colin Raffel and Daniel P. W. Ellis. Large-scale content-based matching of MIDI and audio files. In *Proceedings of the 16th International Society for Music Information Retrieval Conference*, pages 234–240, 2015.

[26] Colin Raffel and Daniel P. W. Ellis. Optimizing DTW-based audio-to-MIDI alignment and matching. In *41st IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 81–85, 2016.

[27] Colin Raffel and Daniel P. W. Ellis. Pruning subsequence search with attention-based embedding. In *41st IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 554–558, 2016.

[28] Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis. `mir_eval`: A transparent implementation of common MIR metrics. In *Proeedings of the 15th International Society for Music Information Retrieval Conference*, pages 376–372, 2014.

[29] Christophe Rhodes, David Lewis, and Daniel Müllensiefen. Bayesian model selection for harmonic labelling. In *Mathematics and Computation in Music*, pages 107–116. Springer, 2007.

[30] Daniel Sleator and David Temperley. The `melisma` music analyzer. `http://www.link.cs.cmu.edu/melisma`, 2001.

[31] Michael Tang, Yip Chi Lap, and Ben Kao. Selection of melody lines for music databases. In *Proceedings of the 24th International Computer Software and Applications Conference*, pages 243–248, 2000.

[32] Robert J. Turetsky and Daniel P. W. Ellis. Ground-truth transcriptions of real music from force-aligned MIDI syntheses. In *Proceedings of the 4th International Society for Music Information Retrieval Conference*, pages 135–141, 2003.

Oral Session 7

Deep-learning

AUTOMATIC TAGGING USING DEEP CONVOLUTIONAL NEURAL NETWORKS

Keunwoo Choi, György Fazekas, Mark Sandler
Queen Mary University of London
{keunwoo.choi, g.fazekas, mark.sandler}@qmul.ac.uk

ABSTRACT

We present a content-based automatic music tagging algorithm using fully convolutional neural networks (FCNs). We evaluate different architectures consisting of 2D convolutional layers and subsampling layers only. In the experiments, we measure the AUC-ROC scores of the architectures with different complexities and input types using the *MagnaTagATune* dataset, where a 4-layer architecture shows state-of-the-art performance with mel-spectrogram input. Furthermore, we evaluated the performances of the architectures with varying the number of layers on a larger dataset (*Million Song Dataset*), and found that deeper models outperformed the 4-layer architecture. The experiments show that mel-spectrogram is an effective time-frequency representation for automatic tagging and that more complex models benefit from more training data.

1. INTRODUCTION

Music tags are a set of *descriptive keywords* that convey high-level information about a music clip, such as emotion (sad, anger, happy), genre (jazz, classical) and instrumentation (guitar, strings, vocal, instrumental). Since tags provide high-level information from the listeners' perspectives, they can be used for music discovery and recommendation.

Automatic tagging is a classification task that aims to predict music tags using the audio signal. This requires extracting acoustic features that are good estimators of the type of tags we are interested in, followed by a single or multi-label classification or in some cases, regression stage. From the perspective of feature extraction, there have been two main types of systems proposed in the literature. Conventionally, feature extraction relies on a signal processing front-end in order to compute relevant features from time or frequency domain audio representation. The features are then used as input to the machine learning stage. However, it is difficult to know what features are relevant to the task at hand. Although feature selection have been widely used to solve this problem [29], clear recommendations which provide good association of features with tag categories are yet to emerge. A more recent approach unifies feature extraction with machine learning to allow relevant features to be learnt automatically. This approach is known as feature learning and requires deep neural networks (DNNs).

Aggregating hand-crafted features for music tagging was introduced in [25]. Several subsequent works rely on a *Bag of frames* approach - where a collection of features are computed for each frame and then statistically aggregated. Typical features are designed to represent physical or perceived aspects of sound and include MFCCs, MFCC derivatives, and spectral features (e.g. spectral roll-off and centroids). Since these are frame-level features, their statistics such as mean and variance are computed [25], or they are clustered and vector quantised [15] to obtain clip-level features. Finally, classifiers such as k-NN or Support Vector Machines are applied to predict tags.

As alternative to the above systems, DNNs have recently become widely used in audio analysis, following their success in computer vision, speech recognition [19] and auto-tagging [6, 8, 18, 28]. From an engineering perspective, DNNs sidestep the problem of creating or finding audio features relevant to a task. Their general structure includes multiple hidden layers with hidden units trained to represent some underlying structure in data.

In computer vision, deep convolutional neural networks (CNNs) have been introduced because they can simulate the behaviour of the human vision system and learn hierarchical features, allowing object local invariance and robustness to translation and distortion in the model [14]. CNNs have been introduced in audio-based problems for similar reasons, showing state-of-the-art performance in speech recognition [19] and music segmentation [26].

Several DNN-related algorithms have been proposed for automatic music tagging too. In [5] and [28], spherical k-means and multi-layer perceptrons are used as feature extractor and classifier respectively. Multi-resolution spectrograms are used in [5] to leverage the information in the audio signal on different time scales. In [28], pre-trained weights of multilayer perceptrons are transferred in order to predict tags for other datasets. A two-layer convolutional network is used in [6] with mel-spectrograms as well as raw audio signals as input features. In [18], bag-of-features are extracted and input to stacked Restricted Boltzmann machines (RBM).

In this paper, we propose an automatic tagging algorithm based on deep *Fully Convolutional Networks* (FCN).

© Keunwoo Choi, György Fazekas, Mark Sandler.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Keunwoo Choi, György Fazekas, Mark Sandler. "Automatic tagging using deep convolutional neural networks', 17th International Society for Music Information Retrieval Conference, 2016.

FCNs are deep convolutional networks that only consists of convolutional layers (and subsampling) without any fully-connected layer. An FCN maximises the advantages of convolutional networks. It reduces the number of parameters by sharing weights and makes the learned features invariant to the location on the time-frequency plane of spectrograms, i.e., it provides advantages over hand-crafted and statistically aggregated features by allowing the networks to model the temporal and harmonic structure of audio signals. In the proposed architecture, three to seven convolutional layers are employed combined with subsampling layers, resulting in reducing the size of feature maps to 1×1 and making the whole procedure fully convolutional. 2D convolutional kernels are then adopted to take the local harmonic relationships into account.

We introduce CNNs in detail in Section 2 and define the problem in Section 3. Our architectures are explained in Section 4, and their evaluation is presented in Section 5, followed by conclusion in Section 6.

2. CNNS FOR MUSIC SIGNAL ANALYSIS

2.1 Motivation for using CNNs for audio analysis

In this section, we review the properties of CNNs with respect to music signals. The development of CNNs was motivated by biological vision systems where information of local regions are repeatedly captured by many sensory cells and used to capture higher-level information [14]. CNNs are therefore designed to provide a way of learning robust features that respond to certain visual objects with local, translation, and distortion invariances. These advantages often work well with audio signals too, although the topology of audio signals (or their 2D representations) is not the same as that of a visual image.

CNNs have been applied to various audio analysis tasks, mostly assuming that auditory events can be detected or recognised by *seeing* their time-frequency representations. Although the advantage of deep learning is to learn the features, one should carefully design the architecture of the networks, considering to what extent the properties (e.g. invariances) are desired.

There are several reasons which justify the use CNNs in automatic tagging. First, music tags are often considered among the topmost high-level features representing song-level information above intermediate level features such as chords, beats, tonality and temporal envelopes which change over time and frequency. This hierarchy fits well with CNNs as it is designed to learn hierarchical features over multilayer structures. Second, the properties of CNNs such as translation, distortion, and local invariances can be useful to learn musical features when the target musical events that are relevant to tags can appear at any time or frequency range.

2.2 Design of CNNs architectures

There have been many variants of applying CNNs to audio signals. They differ by the types of input representations, convolution axes, sizes and numbers of convolutional kernels or subsamplings and the number of hidden layers.

2.2.1 TF-representation

Mel-spectrograms have been one of the widespread features for tagging [5], boundary detection [26], onset detection [21] and latent feature learning [27]. The use of the mel-scale is supported by domain knowledge about the human auditory system [17] and has been empirically proven by performance gains in various tasks [6, 18, 21, 26, 27]. The Constant-Q transform (CQT) has been used predominantly where the fundamental frequencies of notes should be precisely identified, e.g. chord recognition [10] and transcription [22].

The direct use of Short-time Fourier Transform (STFT) coefficients is preferred when an inverse transformation is necessary [3, 23]. It has been used in boundary detection [7] for example, but it is less popular in comparison to its ubiquitous use in digital signal processing. Compared to CQT, the frequency resolution of STFT is inadequate in the low frequency range to identify the fundamental frequency. On the contrary, STFT provides finer resolutions than mel-spectrograms in frequency bands$>$2kHz given the same number of spectral bands which may be desirable for some tasks. So far, however, it has not been the most favoured choice.

Most recently, there have been studies focusing on learning an optimised transformation from raw audio given a task. These are called end-to-end models and applied both for music [6] and speech [20]. The performance is comparable to the mel-spectrogram in speech recognition [20]. It is also noteworthy that the learned filter banks in both [6] and [20] show similarities to the mel-scale, supporting the use of the known nonlinearity of the human auditory system.

2.2.2 Convolution - kernel sizes and axes

Each convolution layer of size $H\times W\times D$ learns D features of $H\times W$, where H and W refer to the height and the width of the learned kernels respectively. The kernel size determines the maximum size of a component it can precisely capture. If the kernel size is too small, the layer would fail to learn a meaningful representation of shape (or distribution) of the data. For this reason, relatively large-sized kernels such as 17×5 are proposed in [10]. This is also justified by the task (chord recognition) where a small change in the distribution along the frequency axis should yield different results and therefore frequency invariance shouldn't be allowed.

The use of large kernels may have two drawbacks however. First, it is known that the number of parameters per representation capacity increases as the size of kernel increases. For example, 5×5 convolution can be replaced with two stacked 3×3 convolutions, resulting in a fewer number of parameters. Second, large kernels do not allow invariance within its range.

The convolution axes are another important aspect of convolution layers. For tagging, 1D convolution along the time axis is used in [6] to learn the temporal distribution, assuming that different spectral band have different distributions and therefore features should be learned per fre-

quency band. In this case, the global harmonic relationship is considered at the end of the convolution layers and fully-connected layers follow to capture it. In contrast, 2D convolution can learn both temporal and spectral structures and has already been used in music transcription [22], onset detection [21], boundary detection [26] and chord recognition [10].

2.2.3 *Pooling - sizes and axes*

Pooling reduces the size of feature map with an operation, usually a *max* function. It has been adopted by the majority of works that are relying on CNN structures. Essentially, pooling employs subsampling to reduce the size of feature map while preserving the information of *an activation* in the region, rather than information about the whole input signal.

This non-linear behaviour of subsampling also provides distortion and translation invariances by discarding the original location of the selected values. As a result, pooling size determines the *tolerance* of the location variance within each layer and presents a trade-off between two aspects that affect network performance. If the pooling size is too small, the network does not have enough distortion invariance, if it is too large, the location of features may be missed when they are needed. In general, the pooling axes match the convolution axes, although it is not necessarily the case. What is more important to consider is the axis in which we need invariance. For example, time-axis pooling can be helpful for chord recognition, but it would hurt time-resolution in boundary detection methods.

3. PROBLEM DEFINITION

Automatic tagging is a *multi-label classification task*, i.e., a clip can be tagged with multiple tags. It is different from other audio classification problems such as genre classification, which are often formalised as a single-label classification problem. Given the same number of labels, the output space of multi-label classification can exponentially increase compared to single-label classification. Accordingly, multi-label classification tasks require more data, a model with larger capacity and efficient optimisation methods to solve. If there are K exclusive labels, the classifier only needs to be able to predict one among K different vectors, which are *one-hot vectors*. With multiple labels however, the number of cases increases up to 2^K.

In crowd-sourced music tag datasets [2, 13], most of the tags are *false*(0) for most of the clips, which makes accuracy or mean square error inappropriate as a measure. Therefore we use the Area Under an ROC (Receiver Operating Characteristic) Curve abbreviated as AUC. This measure has two advantages. It is robust to unbalanced datasets and it provides a simple statistical summary of the performance in a single value. It is worth noting that a random guess is expected to score an AUC of 0.5 while a perfect classification 1.0, i.e., the effective range of AUC spans between [0.5, 1.0].

FCN-4
Mel-spectrogram *(input: 96×1366×1)*
Conv 3×3×128
MP (2, 4) *(output: 48×341×128)*
Conv 3×3×384
MP (4, 5) *(output: 24×85×384)*
Conv 3×3×768
MP (3, 8) *(output: 12×21×768)*
Conv 3×3×2048
MP (4, 8) *(output: 1×1×2048)*
Output 50×1 (sigmoid)

Table 1. The configuration of FCN-4

4. PROPOSED ARCHITECTURE

Table 1 and Figure 1 show one of the proposed architectures, a 4-layer FCN (*FCN-4*) which consists of 4 convolutional layers and 4 max-pooling layers. This network takes a log-amplitude mel-spectrogram sized 96×1366 as input and predicts a 50 dimensional tag vector. The input shape follows the size of the mel-spectrograms as explained in Section 5.1.

The architecture is extended to deeper ones with 5, 6 and 7 layers (*FCN-*{*5, 6, 7*}). The number of feature maps and subsampling sizes are summarised in Table 2. The number of feature maps of FCN-5 are adjusted based on FCN-4, making the hierarchy of the learned features deeper. FCN-6 and FCN-7 however have additional 1×1 convolutional layers(s) on the top of FCN-5. Here, the motivation of 1×1 is to take advantage of increased nonlinearity [16] in the final layer, assuming that the five layers of FCN-5 are sufficient to learn hierarchical features. An architecture with 3 layers (FCN-3) is also tested as a baseline with a pooling strategy of [(3,5),(4,16),(8,17)] and [256, 768, 2048] feature maps. The number of feature maps are adjusted based on FCN-4 while the pooling sizes are set to increase in each layer so that low-level features can have sufficient resolutions.

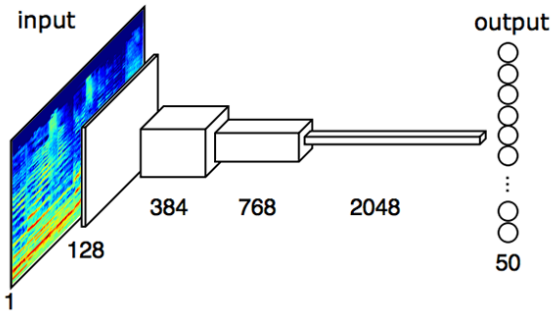


Figure 1. A block diagram of the proposed 4-layer architecture, *FCN-4*. The numbers indicate the number of feature maps (i.e. channels) in each layer. The subsampling layers decrease the size of feature maps to 1×1 while the convolutional layers increase the depth to 2048.

Other configurations follow the current generic optimisation methods in CNNs. Rectified Linear Unit (ReLU) is used as an activation function in every convolutional layer except the output layer, which uses Sigmoid to squeeze the output within [0, 1]. Batch Normalisation is added after every convolution and before activation [11]. Dropout of 0.5 is added after every max-pooling layer [24]. This accelerates the convergence while dropout prevents the network from overfitting.

Homogeneous 2D (3×3) convolutional kernels are used in every convolutional layers except the final 1×1 convolution. 2D kernels are adopted in order to encourage the system to learn the *local* spectral structures. The kernels at the first convolutional layer cover 64 ms×72 Hz. The coverage increases to 7s×692 Hz at the final 3×3 convolutional layer when the kernel is at the low-frequency. The time and frequency resolutions of feature maps become coarser as the max-pooling layer reduces their sizes, and finally a single value (in a 1×1 feature map) represents a feature of the whole signal.

Several features in the proposed architecture are distinct from previous studies. Compared to [28] and [18], the proposed system takes advantages of convolutional networks, which do not require any pre-training but fully trained in a supervised fashion. The architecture of [6] may be the most similar to ours. It takes mel-spectrogram as input, uses two 1D convolutional layers and two (1D) max-pooling layers as feature extractor, and employs one fully-connected layer as classifier. The proposed architectures however consist of 2D convolution and pooling layers, to take the potential local harmonic structure into account. Results from many 3s clips are averaged in [6] to obtain the final prediction. The proposed model however takes the whole 29.1s signal as input, incorporating a temporal nonlinear aggregation into the model.

The proposed architectures can be described as fully-convolutional networks (FCN) since they only consist of convolutional and subsampling layers. Conventional CNNs have been equipped with fully-connected layers at the end of convolutional layers, expecting each of them to perform as a feature extractor and classifier respectively. In general however, the fully connected layers account for the majority of parameters and therefore make the system prone to overfitting. This problem can be resolved by using FCNs with average-pooling at the final convolutional layer. For instance in [16], the authors assume that the target visual objects may show large activations globally in the corresponding images. Our systems resemble the architecture in [16] except the pooling method, where we only use max-pooling because some of the features are found to be local, e.g. the voice may be active only for the last few seconds of a clip.

FCN-5	FCN-6	FCN-7
Mel-spectrogram *(input: 96×1366×1)*		
Conv 3×3×128		
MP (2, 4) *(output: 48×341×128)*		
Conv 3×3×256		
MP (2, 4) *(output: 24×85×256)*		
Conv 3×3×512		
MP (2, 4) *(output: 12×21×512)*		
Conv 3×3×1024		
MP (3, 5) *(output: 4×4×1024)*		
Conv 3×3×2048		
MP (4, 4) *(output: 1×1×2048)*		
.	Conv 1×1×1024	Conv 1×1×1024
	·	Conv 1×1×1024
Output 50×1 (sigmoid)		

Table 2. The configurations of 5, 6, and 7-layer architectures. The only differences are the number of additional 1×1 convolution layers.

5. EXPERIMENTS AND DISCUSSION

5.1 Overview

Two datasets were used to evaluate the proposed system, the MagnaTagATune dataset [13] and the Million Song Dataset (MSD) [2]. The MagnaTagATune dataset has been relatively popular for content-based tagging, but similar performances from recent works [5, 6, 18, 28] seem to suggest that performances are saturated, i.e. a glass-ceiling has been reached due to noise in the annotation. The MSD contains more songs than MagnaTagATune, it has various types of annotations up to 1M songs. There have not been many works to compare our approach with, partly because audio signals do not come with the dataset. Consequently, we use the MagnaTagATune dataset to compare the proposed system with previous methods and evaluate the variants of the system using the MSD.

In Experiment I, we evaluate three architectures (FCN-{3,4,5}) with mel-spectrogram input as proposed in Section 4. Furthermore, we evaluated STFT, MFCC, and mel-spectrogram representations as input of FCN-4. The architecture of STFT input is equivalent to that of mel-spectrograms with small differences in pooling sizes in the frequency axis due to the different number of spectral bands. For the architecture of MFCCs, we propose a frame-based 4-layer feed-forward networks with time-axis pooling (instead of 2D convolutions and poolings) because relevant information is represented by each MFCC rather than its local relationships. In Experiment II, we evaluate five architectures (FCN-{3,4,5,6,7}) with mel-spectrogram input.

Computational cost is heavily affected by the size of the input layers which depends on basic signal parameters of the input data. A pilot experiment demonstrated similar performances with 12 and 16 kHz sampling rates and mel-bins of 96 and 128 respectively. As a result, the audio in both datasets was trimmed as 29.1s clips (the shortest signal in the dataset) and was downsampled to 12 kHz. The hop size was fixed at 256 samples (21 ms) during time-frequency transformation, yielding 1,366 frames in total. STFT was performed using 256-point FFT while the number of mel-bands was set as 96. For each frame, 30 MFCCs

and their first and second derivatives were computed and concatenated.

We used ADAM adaptive optimisation [12] on Keras [4] and Theano [1] framework during the experiments. Binary cross-entropy function is used since it shows faster convergence and better performance than distance-based functions such as mean squared error and mean absolute error.

5.2 Experiment I: MagnaTagATune

The MagnaTagATune dataset consists of 25,856 clips of 29.1-s, 16 kHz-sampled mp3 files with 188 tags. We only uses Top-50 tags, which includes genres (*classical, rock*), instruments (*piano, guitar, vocal, drums*), moods (*soft, ambient*) and other descriptions (*slow, Indian*). The dataset is not balanced, the most frequent tag is used 4,851 times while the 50-th most frequent one used 490 times in the training set. The labels of the dataset consist of 7,644 unique vectors in a 50-dimensional binary vector space.

The results of the proposed architecture and its variants are summarised in Table 3. There is little performance difference between FCN-4 and FCN-5. It is a common phenomenon that an additional layer does not necessarily lead to an improved performance if, *i*) the gradient may not flow well through the layers or *ii*) the additional layer is simply not necessary in the task but only adds more parameters. This results in overfitting or hindering the optimisation. In our case, the most likely reason is the latter of the two. First, the scores are only slightly different, second, both FCN-4 and FCN-5 showed similar performances compared to previous research as shown in Table 4. Similar results were found in the comparison of FCN-5, FCN-6, and FCN-7 in Experiment II. These are discussed in Section 5.3.

Methods	AUC
FCN-3, mel-spectrogram	.852
FCN-4, mel-spectrogram	**.894**
FCN-5, mel-spectrogram	.890
FCN-4, STFT	.846
FCN-4, MFCC	.862

Table 3. The results of the proposed architectures and input types on the MagnaTagATune Dataset

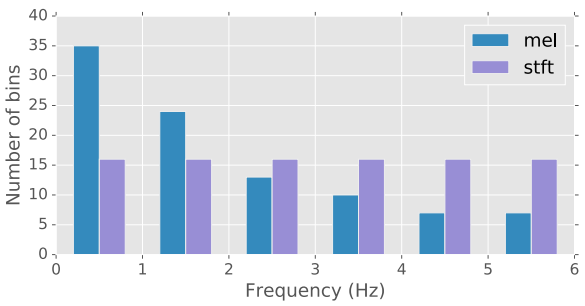


Figure 2. The numbers of bins per 1kHz bandwidth in mel-spectrograms and STFTs .

Methods	AUC
The proposed system, FCN-4	.894
2015, Bag of features and RBM [18]	.888
2014, 1D convolutions [6]	.882
2014, Transferred learning [28]	.88
2012, Multi-scale approach [5]	.898
2011, Pooling MFCC [8]	.861

Table 4. The comparison of results from the proposed and the previous systems on the MagnaTagATune Dataset

The degradations with other types of input signals–STFT and MFCC–are rather significant. This result is aligned with the preferences of mel-spectrograms over STFT on automatic tagging [5,6,18,27]. However, this claim is limited to this or very similar tasks where the system is trained on labels such as genres, instruments, and moods. Figure 2 shows how 96 frequency bins are allocated by mel-spectrograms and STFT in every 1kHz bandwidth. This figure, combined with the result in Table 3 shows that high-resolution in the low-frequency range helps automatic tagging. It also supports the use of downsampling for automatic tagging. Focusing on low-frequency can be more efficient.

Table 4 shows the performance of FCN-4 in comparison to the previous algorithms. The proposed algorithm performs competitively against the other approaches. However, many different algorithms only show small differences in the range of an AUC score of 0.88 – 0.89, making their performances difficult to compare. This inspired the authors to execute a second experiment discussed in the next section. In summary, the mel-spectrograms showed better performance than other types of inputs while FCN-4 and FCN-5 outperformed many previously reported architectures and configurations.

5.3 Experiment II: Million Song Dataset

We further evaluated the proposed structures using the Million Song Dataset (MSD) with *last.fm* tags. We select the top 50 tags which include genres (*rock, pop, jazz, funk*), eras (*60s – 00s*) and moods (*sad, happy, chill*). 214,284 (201,680 for training and 12,605 for validation) and 25,940 clips are selected from the provided training/test sets by filtering out items without any top-50 tags. The number of tags ranges from 52,944 (*rock*) to 1,257 (*happy*) and there are 12,348 unique tag vectors. Note that the size of the MSD is more than 9 times larger than the MagnaTagATune dataset.

The results of the proposed architectures with different numbers of layers are summarised in Table 5. Unlike the result from Experiment I, where FCN-4 and FCN-5 showed a slight difference of the performance (AUC difference of 0.008), FCN-5,6,7 resulted in significant improvements compared to FCN-4, showing that deeper structures benefit more from sufficient data. However, FCN-6 outperformed FCN-5 only by AUC 0.003 while FCN-7 even

Methods	AUC
FCN-3, mel-spectrogram	.786
FCN-4, —	.808
FCN-5, —	.848
FCN-6, —	**.851**
FCN-7, —	.845

Table 5. The results from different architectures of the proposed system on the Million Song Dataset

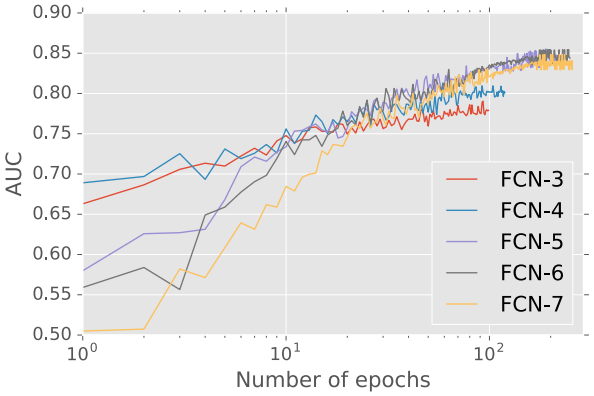


Figure 3. The learning curves of the AUC scores measured on the validation set (on the Million Song Dataset)

showed a slightly worse performance than FCN-6. This result agrees with a known insight in using deep neural networks. The structures of DNNs need to be designed for easier training when there are a larger number of layers [9]. In theory, more complex structures can perform at least equal to simple ones by learning an identity mapping. Our results supports this. In the experiment, the performances of FCN-6 and FCN-7 were still making small improvements at the end of the training, implying it may perform equal to or even outperform FCN-5. In practice, this approach is limited by computational resources and therefore *very deep* structures may need to be designed to motivate efficient training, for instance, using deep residual networks [9].

Figure 3 illustrates the learning curves of the AUC scores on the validation set. At the beginning of the training, there is a tendency that simpler networks show better performance because there is a fewer number of parameters to learn. FCN-4 and FCN-5 show similar performance between around 20–40 epochs. Based on this, it can be assumed that learning on the MagnaTagATune dataset stayed within this region and failed to make more progress due to the scarcity of training data. To summarise, FCN-5, FCN-6, and FCN-7 significantly outperformed FCN-3 and FCN-4. The results imply that more complex models benefit from more training data. The similar results obtained using FCN-5, FCN-6 and FCN-7 indicate the need for more advanced design methodologies and training of deep neural networks.

6. CONCLUSION

We presented an automatic tagging algorithm based on deep fully convolutional neural networks (FCN). It was shown that deep FCN with 2D convolutions can be effectively used for automatic music tagging and classification tasks. In Experiment I (Section 5.2), the proposed architectures with different input representations and numbers of layers were compared using the MagnaTagATune dataset against the results reported in previous works showing competitive performance. With respect to audio input representations, using mel-spectrograms resulted in better performance compared to STFTs and MFCCs. In Experiments II (Section 5.3), different number of layers were evaluated using the Million Song Dataset which contains nine times as many music clips. The optimal number of layers were found to be different in this experiment indicating deeper networks benefit most from the availability of large training data. In the future, automatic tagging algorithms with variable input lengths will be investigated.

7. ACKNOWLEDGEMENTS

This work was part funded by the FAST IMPACt EPSRC Grant EP/L019981/1 and the European Commission H2020 research and innovation grant AudioCommons (688382). Sandler acknowledges the support of the Royal Society as a recipient of a Wolfson Research Merit Award.

8. REFERENCES

[1] Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian Goodfellow, Arnaud Bergeron, Nicolas Bouchard, David Warde-Farley, and Yoshua Bengio. Theano: new features and speed improvements. *arXiv preprint arXiv:1211.5590*, 2012.

[2] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, pages 591–596, 2011.

[3] Keunwoo Choi, George Fazekas, Mark Sandler, and Jeonghee Kim. Auralisation of deep convolutional neural networks: Listening to learned features. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, 2015.

[4] François Chollet. Keras: Deep learning library for theano and tensorflow. https://github.com/fchollet/keras, 2015.

[5] Sander Dieleman and Benjamin Schrauwen. Multi-scale approaches to music audio feature learning. In *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013*, 2013.

[6] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 6964–6968. IEEE, 2014.

[7] Thomas Grill and Jan Schlüter. Music boundary detection using neural networks on spectrograms and self-similarity lag matrices. In *Proceedings of the 23rd European Signal Processing Conference (EUSPICO 2015), Nice, France*, 2015.

[8] Philippe Hamel, Simon Lemieux, Yoshua Bengio, and Douglas Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, pages 729–734, 2011.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[10] Eric J Humphrey and Juan P Bello. Rethinking automatic chord recognition with convolutional neural networks. In *Machine Learning and Applications, 11th International Conference on*, volume 2, pages 357–362. IEEE, 2012.

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[13] Edith Law, Kris West, Michael I Mandel, Mert Bay, and J Stephen Downie. Evaluation of algorithms using games: The case of music tagging. In *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe, Japan, October 26-30, 2009*, pages 387–392. ISMIR, 2009.

[14] Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10), 1995.

[15] Dawen Liang, Minshu Zhan, and Daniel PW Ellis. Content-aware collaborative music recommendation using pre-trained neural networks. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, 2015.

[16] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2013.

[17] Brian CJ Moore. *An introduction to the psychology of hearing*. Brill, 2012.

[18] Juhan Nam, Jorge Herrera, and Kyogu Lee. A deep bag-of-features model for music auto-tagging. *arXiv preprint arXiv:1508.04999*, 2015.

[19] Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran. Deep convolutional neural networks for lvcsr. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8614–8618. IEEE, 2013.

[20] Tara N Sainath, Ron J Weiss, Andrew Senior, Kevin W Wilson, and Oriol Vinyals. Learning the speech front-end with raw waveform cldnns. In *Proc. Interspeech*, 2015.

[21] Jan Schluter and Sebastian Bock. Improved musical onset detection with convolutional neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*. IEEE, 2014.

[22] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic music transcription. *arXiv preprint arXiv:1508.01774*, 2015.

[23] Andrew JR Simpson, Gerard Roma, and Mark D Plumbley. Deep karaoke: Extracting vocals from musical mixtures using a convolutional deep neural network. *arXiv preprint arXiv:1504.04658*, 2015.

[24] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[25] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *Speech and Audio Processing, IEEE transactions on*, 10(5):293–302, 2002.

[26] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary detection in music structure analysis using convolutional neural networks. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan*, 2014.

[27] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*, pages 2643–2651, 2013.

[28] Aäron Van Den Oord, Sander Dieleman, and Benjamin Schrauwen. Transfer learning by supervised pre-training for audio-based music classification. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, 2014.

[29] Yusuf Yaslan and Zehra Cataltepe. Audio music genre classification using different classifiers and feature selection methods. In *18th ICPR 2006*, volume 2, pages 573–576. IEEE, 2006.

A HYBRID GAUSSIAN-HMM-DEEP-LEARNING APPROACH FOR AUTOMATIC CHORD ESTIMATION WITH VERY LARGE VOCABULARY

Junqi Deng and Yu-Kwong Kwok
Department of Electrical and Electronic Engineering
The University of Hong Kong
{jqdeng,ykwok}@eee.hku.hk

ABSTRACT

We propose a hybrid Gaussian-HMM-Deep-Learning approach for automatic chord estimation with very large chord vocabulary. The Gaussian-HMM part is similar to Chordino, which is used as a segmentation engine to divide input audio into note spectrogram segments. Two types of deep learning models are proposed to classify these segments into chord labels, which are then connected as chord sequences. Two sets of evaluations are conducted with two large chord vocabularies. The first evaluation is conducted in a recent MIREX standard way. Results show that our approach has obvious advantage over the state-of-the-art large-vocabulary-with-inversions supportable ACE system in terms of large vocabularies, although is outperformed by in small vocabularies. Through analyzing and deducing system behaviors behind the results, we see interesting chord confusion patterns made by different systems, which conceivably point to a demand of more balanced and consistent annotated datasets for training and testing. The second evaluation preliminarily demonstrates our approach's superiority on a jazz chord vocabulary with 36 chord types, compared with a Chordino-like Gaussian-HMM baseline system with augmented vocabulary capacity.

1. INTRODUCTION

Automatic chord estimation (ACE) is currently undergoing a paradigm shift from Gaussian-HMM (Hidden Markov Model) approaches to deep learning approaches. Recently, there have been quite a few deep learning powered ACE approaches in the field, including a convolutional neural network (CNN) approach [10], a hybrid feedforward-recurrent neural network (DNN-RNN) approach [3], a deep belief network (DBN) approach [19], and a hybrid DBN-RNN approach [16]. Some are more purely deep learning oriented, which only apply minimal amount of feature extractions, while others consider combination of traditional signal processing techniques and deep learning.

© Junqi Deng and Yu-Kwong Kwok. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Junqi Deng and Yu-Kwong Kwok. "A Hybrid Gaussian-HMM-Deep-Learning Approach For Automatic Chord Estimation With Very Large Vocabulary", 17th International Society for Music Information Retrieval Conference, 2016.

One common point of these approaches is that they are all evaluated under major/minor vocabulary (MajMin), which is far from reflecting the complexity of chord vocabulary in pop/rock music practice. In 2013, MIREX ACE has introduced a new evaluation scheme [14] focusing on much more complicated chord vocabulary, the "SeventhsBass", which includes MajMin, three types of their seventh chords, and all of their inversions. The SeventhsBass, although also omitting some rare chords in pop/rock practice, is much closer to the reality compared with MajMin. It differentiates among triads, sevenths and their inversions because they all have different harmonic qualities. It is not only important for ACE systems to be evaluated on more complex chord vocabulary, but also to actually support that vocabulary. Unfortunately from 2013 to 2015, there have been only two systems that actually support SeventhsBass [6], others mostly do not even support chord inversions. Not being able to generate inversions is musically problematic since in some musical context they have very different harmonic qualities from their root positions. As shown in Figure 1, for example, the chord inversions serve as a diatonic or chromatic continuations of the bass line. If some of these are replaced by their root positions, the continuations are broken and thus the pieces will sound very different.

1) | G | D/F# | F | C/E | Cm/Eb |
2) | A | Bm | A/C# | D |
3) | C | G/B | Am | Am/G | F | C/E |
4) | C | F | C/E | D/F# | E/G# | F#/A# | Bm7 | C# |

Figure 1. Four chord progressions that contain bass line continuations which demand chord inversions. Progressions like 1,2 and 3 are very popular among pop/rock. Progression 4 induces a key shift from C major to F# minor.

Following the above argument, we propose an ACE system that not only supports but also be evaluated on SeventhsBass. This system uses a Chordino-like module [13] as a chord segmentation engine, and classifies chords within each segment using a deep learning model. Evaluation results show that the best system variants have obvious advantage over the state-of-the-art SeventhsBass supportable ACE system in terms of Sevenths (MajMin + maj7, min7, 7) and SeventhsBass.

Besides, we also try the proposed approach on a jazz vocabulary. The comparison target remains the same except for an augmentation of its chord vocabulary capacity. Since the standard evaluation tool [14] does not apply for this vocabulary, evaluation is done manually via comparison of weighted chord symbol recall [1] . Results show similar ranking as in the SeventhsBass' results, and still the best system significantly outscores the baseline approach.

The rest of this paper is organized as follows: Section 2 gives an overview of the proposed ACE system framework and its workflow; Section 3 elaborates the implementations of two deep learning based models (DBN and BLSTM-RNN); Section 4 reports both SeventhsBass and jazz vocabulary evaluation results, with a detailed discussion of chord confusion and how they affect systems' performances; Section 5 concludes the paper and puts forward some possible future considerations in ACE.

2. SYSTEM OVERVIEW

The proposed ACE approach [2] has a simple workflow as shown in Figure 2. The test data goes through a Chordino-like module for segmentation. Then each note spectrogram (referred to as "notegram" below) segment will be classified using a deep learning model. The output chord sequence is obtained by connecting the classified labels.

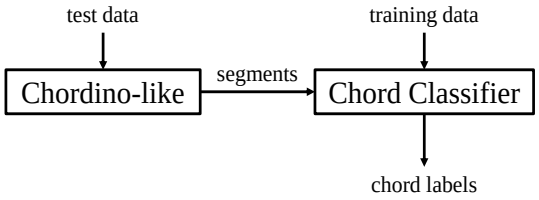


Figure 2. System overview. The audio input (test data) goes through a Chordino-like process for segmentation, then the segments are classified into chord labels.

The Chordino-like module is implemented according to the algorithmic description of Chordino [12, 13]. The audio input is first resampled at 11025 Hz, and transformed by a 4096-point Hamming window short-time-Fourier-transform (STFT) with 512 point hop size. The linear-scale spectrogram is then mapped to a log-scale spectrogram, or notegram. After standard tuning (tuned notegram) and feature scaling, note activation patterns are extracted from the notegram via non-negative-least-square (NNLS) method. A piece of chromagram is derived by bass-treble profiling of the note activation patterns. The chromagram is then decoded and segmented by a Gaussian-HMM with very high self-transition weights.

The chord classifier is implemented using deep learning models, which will be discussed in the following section. Applying different deep learning models leads to different system variants out of the proposed framework. In the following, we refer to these "variants" as "systems", and the framework as the "approach".

[1] http://www.music-ir.org/mirex/wiki/2013:Audio_Chord_Estimation

[2] the full implementation of this ACE system is accessible via: https://github.com/tangkk/tangkk-mirex-ace

3. DEEP LEARNING MODELS

We consider two types of deep learning models. They both have input at the tuned notegram level. The deep neural network will learn the rest of the transformations from tuned notegram all the way to chord label. Since there are different numbers of frames in different chord segments, in order to use a fixed-length input structure, we conducted a preliminary study and found that 6 sub-segments are good for single chord classification task. Note that the number of sub-segments should at least reflect the temporal order of bass line in order to differentiate root position from inversions. Thus we compute a 6-frame notegram for each segment as follows: at first the segment is divided into 6 equal-size sub-segments; if the total number of frames is not divisible by 6, the last frame is extended several times to make it divisible; then notegram in each sub-segment is averaged over time, resulting in one frame per sub-segment.

3.1 DBN Model

We first consider a DBN model. It contains two hidden layers, each of 800 neurons. The input layer is of 6×252-dimension (252 is the size of a notegram frame), and the output layer is a #chord-way softmax layer. The neurons of both input and output layers are of Gaussian type (real value from 0 to 1). The neurons in both hidden layers are of Bernoulli type (binary value 0 or 1).

During unsupervised pre-training, the first restricted-Boltzmann-machine (RBM) formed by the first two layers is considered as a Gaussian-Bernoulli RBM, and the second RBM formed by the two hidden layers is considered as a Bernoulli-Bernoulli RBM. The pre-training is conducted using persistent-contrastive-divergence-10 [17] (PCD-10), for 100 epochs with learning rate 0.001. During supervised fine-tuning, the network connections are updated using mini-batch stochastic gradient descent, and the updates are regularized by dropout [8] (with 0.5 dropout probability) and early-stopping. The stopping criteria is monitored by a validation set, which randomly contains 20% of the training set. The other 80% are used for computing the gradients. Due to the randomness of train/validation split, we repeatedly train 6 models. The model with the best validation score will be saved for testing.

For comparison, we also consider a feed-forward multilayer perceptron (MLP) model, whose network configuration is the same as the DBN, but trained using only the fine-tuning procedure described above.

3.2 BLSTM-RNN Model

Historically, long-short-term-memory (LSTM) [9] unit is introduced to try to solve the gradient vanishing problem [2] when training a recurrent neural network with a long sequence of examples.

3.2.1 LSTM Unit

Instead of having only one input port, an LSTM unit has four input ports. As shown in Figure 3, three of them are used for gating purpose, and the other is used for normal purpose. Each gate computes an output gating signal from the weighted sum of its inputs using a non-linear activation function. The gating signal computed by input gate, output gate and forget gate will interact with both the LSTM unit's input value and the LSTM cell value through simple multiplications, resulting in the LSTM unit's output value. Input gate regulates the amount of input feeding into the cell; forget gate regulates the current cell value by the previous cell value; and output gate regulates the amount of output by interacting with the current cell value. Since all functions involved in an LSTM unit are differentiable or partially differentiable, all connections can be trained using the same back-propagation-through-time (BPTT) [7] technique as used in training a normal RNN.

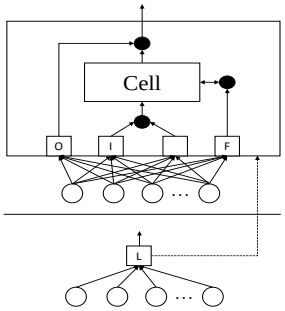


Figure 3. LSTM unit. O = output gate; I = input gate; F = forget gate; Black dots indicate multiplication operations

3.2.2 BLSTM-RNN

We then consider a BLSTM-RNN model [7] as shown in Figure 4. It has both forward and backward LSTM layers, each of which has 800 LSTM units. Before the #chord-way softmax output layer, it performs mean pooling to summarize results from all frames. During training, the RNN is always unrolled to 6 frames, and the weights are updated via BPTT using AdaDelta algorithm [18], regularized with dropout (with 0.5 dropout probability) and early-stopping, monitored by a validation set chosen in the same way as in DBN's case. Due to the randomness of train/validation split, we repeatedly train 6 models. The model with the best validation score will be saved for testing.

4. EVALUATION

For SeventhsBass ACE implementation, four datasets of 266 tracks in total are used in training. They contain both eastern and western pop/rock songs. They are: 1, JayChou29 dataset [5]; 2, a Chinese pop song dataset (CNPop20) [3]; 3, Carole King + Queen dataset

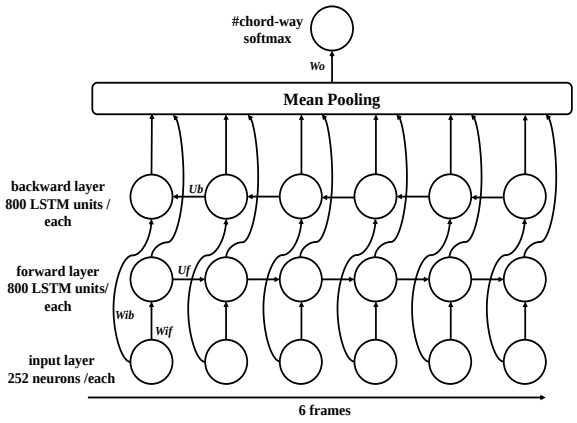


Figure 4. Bidirectional-long-short-term-memory recurrent neural network (BLSTM-RNN) used in the proposed approach

(KingQueen26) [4]; 4, 191 songs from USPop dataset (U) [5]. In order to see the effect of data size, all models will be incrementally trained on: 1, JayChou29 and CNPop20 (CJ); 2, CJ + KingQueen26 (CJK); 3, all four datasets (CJKU).

For Jazz ACE implementation, 99 pieces of jazz chord comping + soloing dataset extracted from a jazz guitar book [15] (JazzGuitar99) are used as training/validation dataset, and 7 pieces from Gary Burton's online course [1] (GaryBurton7) are used as test dataset. JazzGuitar99's annotations are taken directly from the book, and GaryBurton7's annotations are taken from the leadsheets provided along with the course. The jazz chord vocabulary contains 36 types [6]. Note that inversions are not considered in this preliminary jazz ACE study because: 1. there are very few inversion notations in the currently used datasets; 2. it results in huge number of classes based on these 36 types.

All training data are to be used at their tuned notegram level, which does not contain phase information. Assuming well temperament, we can augment all training data by pitch shifting their notegrams to all 12 keys with zero padding. Adjusting the chord labels accordingly, this results in 12 times of training data.

4.1 SeventhsBass Vocabulary Systems Evaluation

SeventhsBass evaluation is conducted in a MIREX standard way. We use TheBeatles180 (B) as the test set and run end-to-end automatic chord transcriptions from raw audio to chord progression for every track within. The metric score is computed in a weighted chord symbol recall (WCSR) way using the MIREX ACE evaluation tool [14]. All systems are compared with each other and compared with Chordino. Chordino is the only other suitable system

[3] containing 20 songs from both male and female singer-songwriters from Chinese cultural backgrounds including mainland China, Hong Kong and Taiwan

[4] http://isophonics.net/datasets

[5] https://github.com/tmc323/Chord-Annotations

[6] They are: maj, min, min6, 6, maj7, maj7#5, maj7#11, maj7b5, min7, minmaj7, min7b5, min7#5, 7, 7b5, 7b9, 7#9, 7#5#9, 7#5b9, 7b5b9, 7#5, 7sus4, aug7, dim7, maj9, min9, 9, 9#11, min11, min11b5, 11, min13, maj13, 13, 13b9, 69 and N

	Mm	MmB	S	**SB**
Chordino	**74.30**	**71.40**	52.99	50.60
CJ-MLP	67.25	62.27	55.15	50.86
CJ-DBN	70.68	66.52	**58.23**	**54.71**
CJ-BLSTM	69.09	64.51	56.47	52.74
CJK-MLP	65.18	63.12	53.82	52.00
CJK-DBN	67.44	65.56	55.64	54.03
CJK-BLSTM	70.46	68.56	**59.11**	**57.50**
CJKU-MLP	67.95	65.87	55.98	54.09
CJKU-DBN	68.53	66.49	56.19	54.37
CJKU-BLSTM	72.62	70.47	**59.37**	**57.47**

Table 1. WCSRs of four main MIREX ACE vocabulary (Mm = MajMin, MmB = MajMinBass, S = Sevenths, SB = SeventhsBass; CJ = JayChou29 + CNPop20; CJK = CJ + KingQueen26; CJKU = CJK + USPop191)

for comparison because this is the only publicly available system which also supports SeventhsBass vocabulary [4].

Here we argue for the validity of our evaluation methodology. Note that our systems are trained with combination of C, J, K, U, and tested on B. Some may challenge that since these two sets may be drawn from two different chord populations (NOT in terms of chord types, but chord rendering styles), thus the test results may not reflect the true system performance. It is true that they contain different distributions of chord rendering styles, especially in terms of the "dominant sevenths" chord, as also reflected in the results and discussions in Section 4.1.2. But as we will see in the results, in general, the C,J,K,U-trained systems generalize very well on B. In fact, since there could be countless of possible chord rendering styles of each chord, it is difficult to "make sure" that two datasets are drawn from the "same" population, not to mention that it is even more difficult to define the possible "properties" of such "population". An average k-fold cross-validation score could be a better indication of system performance in terms of the combined CJKUB training/test set, but neither is this a standard benchmarking method, nor can this score be directly compared with an expert system such as Chordino.

4.1.1 Overall Results of SeventhsBass

The WCSRs of four main MIREX ACE vocabularies are shown in Table 1. In MajMin and MajMinBass, Chordino still does the best among all systems. But in Sevenths and SeventhsBass (the main focus in this paper), all our systems perform better than Chordino, with CJ-DBN, CJK-BLSTM and CJKU-BLSTM performing best.

Let's take CJ-DBN as representative for the moment. It seems that it performs better at recognizing seventh chords but worse at inversions compared with Chordino, but this is not a correct deduction from the table. Note that Sevenths is a collapse of chords, regardless of root positions or inversions, to their maj, min, maj7, min7 or 7 forms; and MajMinBass is a collapse of chords, regardless of tetrads or triads, to their maj, min, maj/3, maj/5, min/b3 or min/5 forms. Considering SeventhsBass as all chords in their original forms, the score boost from SeventhsBass to Sevenths indicates the amount of confusion between root positions and inversions (let's call it "bass confusion"); the score boost from SeventhsBass to MajMinBass indicates

	maj	min	maj/3	maj/5	min/b3	min/5
maj (r)	0.66	0.03	0.00	0.02	0.00	0.00
min (r)	0.10	0.60	0.00	0.01	0.00	0.00
maj/3 (r)	0.35	0.13	0.19	0.00	0.00	0.00
maj/5 (r)	0.50	0.08	0.00	0.23	0.00	0.00
min/b3 (r)	0.36	0.30	0.01	0.06	0.00	0.00
min/5 (r)	0.19	0.55	0.04	0.04	0.00	0.00

Table 3. Chordino's bass confusion matrix.

	maj	min	maj/3	maj/5	min/b3	min/5
maj (r)	0.72	0.06	0.03	0.03	0.00	0.00
min (r)	0.15	0.63	0.02	0.02	0.00	0.00
maj/3 (r)	0.34	0.28	0.23	0.01	0.00	0.02
maj/5 (r)	0.49	0.11	0.03	0.19	0.01	0.00
min/b3 (r)	0.39	0.20	0.06	0.07	0.01	0.00
min/5 (r)	0.28	0.28	0.11	0.06	0.00	0.06

Table 4. CJ-DBN's bass confusion matrix.

the amount of confusion between tetrads and triads (let's call it "seventh confusion"); and the score boost from SeventhsBass to MajMin approximately sums up two types of confusion. It should be noted that there are yet other types of confusion, such as confusion of roots, or of maj and min, which could not be regarded as correct in any ways under the current evaluation method.

Following this deduction, the Sevenths result actually indicates that CJ-DBN still scores much better than Chordino if bypassing bass confusion; while the MajMinBass result indicates that CJ-DBN scores much lower than Chordino if bypassing seventh confusion. Therefore compared with Chordino, CJ-DBN has a better chance of bass confusion, but less chance of seventh confusion. Notice that in CJ-DBN, the difference between MmB and SB is much larger than that between S and SB, which means the net amount of seventh confusion is much more than that of bass confusion. The same is also true in Chordino. Therefore in both systems, there are much higher chances of making seventh confusion than bass confusion.

As for the intra-comparison among all proposed systems, three observations are noticeable: 1, DBN has advantage over MLP, and this advantage decreases with the increase of training data size; 2, BLSTM-RNN has obvious advantage over DBN with big enough training data size; 3, the investment of more data yields diminishing return. The first point is mainly due to the intensive unsupervised pre-training in DBN. The second point demonstrates that the proposed BLSTM-RNN model has better capability in modeling a single chord than the proposed DBN model. BLSTM-RNN is good at modeling temporal dependency, but DBN is good at modeling spacial dependency. An input feature with 6 frames of time dependent notegrams should be more suitable for temporal modeling, thus a plausible reason behind the second observation. The third observation may possibly point to a ground truth annotation consistency problem [11], which will be explained in next subsection.

4.1.2 Details of SeventhsBass

A deeper look at the per chord-type WCSR of SeventhsBass may reveal more details behind the overall scores.

SeventhsBass	M/5	M/3	M	M7/5	M7/3	M7/7	M7	7/5	7/3	7/b7	7	m/5	m/b3	m	m7/5	m7/b3	m7/b7	m7
B%	2.0	1.0	63.3	0.0	0.2	0.3	0.8	0.1	0.1	0.4	8.3	0.6	0.4	15.0	0.0	0.1	0.4	2.4
Chordino	19.9	17.1	54.4	0.0	0.0	0.0	55.6	0.0	0.0	5.7	41.0	0.0	0.0	54.3	0.0	0.0	0.0	51.0
CJ-MLP	15.8	19.8	58.2	0.0	0.0	0.0	30.0	0.0	0.0	9.5	11.5	3.7	0.7	54.2	0.0	0.0	0.2	19.9
CJ-DBN	19.2	21.7	63.0	0.0	0.0	0.0	35.5	0.0	0.0	20.8	9.0	5.6	0.7	59.8	0.0	0.0	0.0	21.6
CJ-BLSTM	15.3	22.4	60.4	0.0	0.0	0.0	34.8	0.0	0.0	13.1	10.2	10.2	1.0	59.0	0.0	0.0	0.0	28.0
CJK-MLP	5.6	14.2	62.7	0.0	0.0	0.0	30.7	0.0	0.0	2.7	10.0	1.7	0.0	46.7	0.0	0.0	0.0	22.8
CJK-DBN	7.6	19.2	64.2	0.0	0.0	0.0	37.7	0.0	0.0	5.0	13.5	1.9	1.6	51.5	0.0	0.0	0.0	27.0
CJK-BLSTM	6.4	12.0	70.5	0.0	0.0	0.0	37.8	0.0	0.0	10.2	8.5	9.8	1.9	48.7	0.0	0.0	0.3	32.1
CJKU-MLP	11.8	18.3	63.8	0.0	0.0	0.0	18.4	0.0	0.0	3.7	19.4	0.5	0.4	52.7	0.0	0.0	0.6	20.9
CJKU-DBN	8.2	16.2	64.3	0.0	0.0	0.0	19.5	0.0	0.0	1.2	20.4	1.9	2.1	52.9	0.0	0.0	0.0	20.2
CJKU-BLSTM	22.4	16.1	66.6	0.0	0.0	0.0	33.2	0.0	0.0	8.9	23.9	2.8	3.2	59.0	0.0	0.0	0.3	26.6

Table 2. WCSRs of every SeventhsBass category. (M=maj, m=min). %B shows the constitution of chord in test set.

Table 2 shows the categorical breakdowns of the SeventhsBass' WCSRs. Our systems' advantages in M and m are as expected. As the training data contains huge amount of their examples, deep learning models can take full advantages and draw clear boundaries between M v.s.non-M and m v.s. non-m. Table 3 and 4 show a comparison of bass confusion in Chordino and CJ-DBN [7] , which not only reflects CJ-DBN's advantages in M and m, but also confirms our previous deduction that CJ-DBN makes slightly more bass confusion than Chordino.

The results of M/5, M/3 and 7/b7 deserve further investigation. The "CJ-" systems generally perform better than Chordino in these categories. This could be due to both C and J contain a large number of consistent annotations of these chords. In the meantime we observe their scores generally drop with introduction of K and U, seemingly in exchange for more score boost from M. This seems contradictory: since all three chord types (M, M/3 and M/5) have clear distinctions by definition, thus given a neural network with enough modeling capacity and properly trained (which we assume is the case), more ground truth data should yield better classification boundaries. But instead the introduction of K and U also introduces chaotic classification behaviors regarding, M/3, M/5, 7/b7 and M. Thus we have to believe that these results conceivably point to a ground truth annotation consistency problem [11], where, for example, some similarly rendering M/3 chords in different datasets are annotated differently (as M, M/3, M/5 or others), so that when trained on a combined dataset, the classifier is getting confused about the boundaries between those similar chords. Assuming more inversions are "mis-annotated" [8] as root positions than vice versa (which might unfortunately be true), if such inconsistencies abound, classifications will be bias towards the dominating root position chords.

The most noticeable drawback of our systems is the poor performance of all sevenths chords (M7, 7 and m7) compared with Chordino. Chordino has a very nice and balanced chord confusion matrix. Shown in Table 5, almost every chord type has less than 50% confusion with other types. As for our approach, taking CJK-BLSTM as example, the main problem is that both M7 and 7 chords are easily confused with maj, and m7 is easily confused

[7] The numbers in the table are normalized durations. Reference labels are indicated by "(r)"

[8] technically not necessarily a "miss" but let's just use this expression for convenience in this context

	maj	min	maj7	min7	7
maj (r)	0.66	0.03	0.11	0.03	0.13
min (r)	0.10	0.60	0.03	0.20	0.03
maj7 (r)	0.22	0.08	0.62	0.02	0.01
min7 (r)	0.12	0.20	0.01	0.56	0.08
7 (r)	0.30	0.08	0.06	0.06	0.47

Table 5. Chordino's seventh confusion matrix.

	maj	min	maj7	min7	7
maj (r)	0.82	0.05	0.03	0.02	0.03
min (r)	0.21	0.52	0.01	0.17	0.02
maj7 (r)	0.42	0.07	0.39	0.03	0.01
min7 (r)	0.21	0.31	0.02	0.34	0.03
7 (r)	0.67	0.12	0.01	0.05	0.10

Table 6. CJK-BLSTM's seventh confusion matrix

with min (Table 6). The most undesirable case is the confusion between 7 and maj . The main reason behind this, as we try to analyze, is the different distribution of 7s in the training datasets and the test dataset. The Beatles' albums contain a lot of chord progressions that involve 7s, where the bass lines are moving by arpeggio or running as broken chords, but in CJK, there are very few such examples. CJK contains 7s that are mostly bass line static. Thus CJK-BLSTM does not have enough chance to learn 7 in dynamic bass line population, resulting in these poor results. This analysis is to some degree confirmed by the much better scores of 7 after adding dataset U, which contains a lot more 7 chord renderings in dynamic style.

For Sevenths' inversions other than "7/b7", since there are not many examples in all datasets, it is not meaningful for further discussion. Actually, their WCSRs are all relatively low. This fact might in some sense invalidate the necessity to recognize more complicated inversions, but does not invalidate the need to capture inversions in general.

4.2 Jazz Chord Vocabulary Systems Evaluation

Following the MIREX ACE convention, system performance on jazz chord vocabulary should also be evaluated based on WCSR. The WCSR score computing procedure in its fairest/strictest sense should count each chord as it is

	μ	σ^2
Bass - Chord Bass	1	0.1
Treble - Chord Note	1	0.2
Neither bass nor treble	0	0.2
"N" Chord	1	0.2

Table 7. Gaussian model of Jazz-Chordino

systems	WCSR	SQ
Jazz-Chordino	57.99	81.68
Jazz-MLP	61.81	76.18
Jazz-DBN	62.33	80.73
Jazz-BLSTM	**66.41**	80.78

Table 8. WCSRs and SQ (segmentation quality) of jazz chord vocabulary.

without applying any sort of mapping scheme, as happens to SeventhsBass. In the following we evaluate each system in this way. The baseline is an augmented Chordino with jazz vocabulary extension (Jazz-Chordino). The augmentation is done within its Gaussian-HMM engine by applying the jazz chord dictionary to the Gaussian model, whose setting is described in Table 7.

The jazz vocabulary systems have the same system framework as the SeventhsBass systems, but their deep learning models are trained using JazzGuitar99 dataset. All systems are tested using GaryBurton7 dataset [9]. Results are shown in Table 8. Jazz-BLSTM system performs the best, and outperforms Jazz-Chordino by about 10 points. The ranking is very similar to SeventhsBass', but the results are in a sense more convincing, since the test set is not dominated by chords like major and minor. In fact the composition of chords in GaryBurton7 is relatively balanced, though rare chords are still rare. Therefore in this set of results we see clearly the advantage of hybrid Gaussian-HMM-Deep-Learning approach over a pure Gaussian-HMM approach for very large chord vocabulary.

Meanwhile, notice that the SQ of these systems are all relatively high, and these are achieved in pure jazz test audio. All systems use Jazz-Chordino's Gaussian-HMM as segmentation engine. The differences between SQ scores are caused by different merging of consecutive chord boundaries in different systems. Obviously the success of Jazz-BLSTM is based on the success of the Gaussian-HMM segmentation at the beginning; then based on the robust segmentation it performs classifications without taking care of chord progression context. This task is comfortable to deal with by a fixed-length input deep learning model. The advantage may not be obvious under a small chord vocabulary, but is obvious under a large chord vocabulary.

5. CONCLUSION

In this paper we propose a hybrid Gaussian-HMM-Deep-Learning approach towards SeventhsBass and jazz vocabulary automatic chord estimation. Based on a Chordino-like segmentation engine, the approach applies two types of deep learning models, i.e., DBN and BLSTM-RNN, for chord classifications.

For SeventhsBass implementation, we train several models of each type using four datasets in an incremental way. The systems are tested using another dataset, and compared with Chordino. Results show that the best system variant, CJKU-BLSTM obviously outperforms Chordino in both Sevenths and SeventhsBass, but is slightly outperformed by Chordino in MajMin and MajMinBass. We find that our system tends to make more bass confusion but less seventh confusion compared with Chordino. The major success of our systems is in triads, while the major drawbacks are in sevenths chords. The trends within the results along incremental training data sizes may indicate a possible data annotation inconsistency issue that conceivably leads to diminishing return effect.

For jazz vocabulary implementation, we train one model for each type using JazzGuitar99 dataset, test them using GaryBurton7 dataset, and compare them with a Chordino-like system augmented with jazz chord vocabulary (Jazz-Chordino). Results show a similar system ranking as in SeventhsBass' results, with high segmentation qualities. The best system, Jazz-BLSTM, outscores Jazz-Chordino obviously. Given that GaryBurton7 is a relatively chord balanced dataset, the results demonstrate more clearly the advantage of hybrid Gaussian-HMM-Deep-Learning approach over pure Gaussian-HMM approach, which might not be so obvious with much smaller chord vocabulary.

Generally speaking, Chordino is an elegant music knowledge driven expert system that generally recognizes chords very well. But at times it fails also because of its simplicity, which fails to capture chords rendered in abnormal ways. On the other hand, our approach is data driven. The success or non-success of it depends highly on the chord balancing, distribution and population of training data. While performances on some dominating chords benefit much from the data, other performances suffer a lot from data insufficiency or inconsistency.

There are a few concerns to be addressed. The first concern is about the manually engineered segmentation engine. The Gaussian-HMM segmentation engine is good indeed, but for scientific interest, we are also very curious about whether by doing a deep training on huge amount of data can one system learn that transformation. Preliminary researches are ongoing, but none of our attempts have achieved that level yet. We believe this can be achieved gradually by deeper models and more data. A separate training for segmentation only might be beneficial. The second concern is about datasets. A better training based system asks for more ground truth annotations, especially those of skew classes, so as to train a more balanced system and to avoid the main contribution of performance being dominated by a few classes. Generally more data will lead to more examples of skew classes, but due to annotation inconsistency issue, simply "more data" may not be the final solution at all, which leaves much more works to be done in this area. Finally there is a concern of vocabulary size (seems contradictory to the previous concern), which asks for gradually exploring ACE systems' capabilities on more complex vocabularies as it is the way to approach the ultimate goal of ACE, which is to match human expert's ability of doing chord recognition.

[9] Composition of chords in GaryBurton7: maj:0.09; min7:0.13; 7:0.22; min7b5:0.12; 7b9:0.06; min:0.1; maj:0.14; others:0.14.

6. REFERENCES

[1] Gary Burton Jazz Improvisation Course. https://www.coursera.org/learn/jazz-improvisation/. Accessed: 2016-02-16.

[2] Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[3] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Audio chord recognition with recurrent neural networks. In *ISMIR*, pages 335–340, 2013.

[4] J Ashley Burgoyne, W Bas de Haas, and Johan Pauwels. On comparative statistics for labelling tasks: What can we learn from mirex ace 2013. In *Proceedings of the 15th Conference of the International Society for Music Information Retrieval (ISMIR 2014)*, pages 525–530, 2014.

[5] Junqi Deng and Yu-Kwong Kwok. MIREX 2015 submission: Automatic chord estimation with chord correction using neural network, 2015.

[6] Junqi Deng and Yu-Kwong Kwok. Automatic chord estimation on seventhsbass chord vocabulary using deep neural network. In *Proceedings of the 41th International Conference on Acoustics, Speech and Signal Processing (ICASSP 2016). Shanghai, China*, 2016.

[7] Alex Graves. *Supervised sequence labelling*. Springer, 2012.

[8] Geoffrey Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 9(1):926, 2010.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[10] Eric J Humphrey and Juan P Bello. Rethinking automatic chord recognition with convolutional neural networks. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volume 2, pages 357–362. IEEE, 2012.

[11] Eric J Humphrey and Juan P Bello. Four timely insights on automatic chord estimation. In *Proceedings of the 16th Conference of the International Society for Music Information Retrieval (ISMIR 2015)*, 2015.

[12] Matthias Mauch. *Automatic chord transcription from audio using computational models of musical context*. PhD thesis, School of Electronic Engineering and Computer Science Queen Mary, University of London, 2010.

[13] Matthias Mauch and Simon Dixon. Approximate note transcription for the improved identification of difficult chords. In *ISMIR*, pages 135–140, 2010.

[14] Johan Pauwels and Geoffroy Peeters. Evaluating automatically estimated chord sequences. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 749–753. IEEE, 2013.

[15] Jeff Schroedl. *Hal Leonard Guitar Method - Jazz Guitar: Hal Leonard Guitar Method Stylistic Supplement Bk/online audio*. Hal Leonard, 2003.

[16] Siddharth Sigtia, Nicolas Boulanger-Lewandowski, and Simon Dixon. Audio chord recognition with a hybrid recurrent neural network. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR 2015). Malaga, Spain*, 2015.

[17] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071. ACM, 2008.

[18] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[19] Xinquan Zhou and Alexander Lerch. Chord detection using deep learning. In *Proceedings of the 16th ISMIR Conference*, volume 53, 2015.

MELODY EXTRACTION ON VOCAL SEGMENTS USING MULTI-COLUMN DEEP NEURAL NETWORKS

Sangeun Kum, Changheun Oh, Juhan Nam
Graduate School of Culture Technology
Korea Advanced Institute of Science and Technology
{keums, thecow, juhannam}@kaist.ac.kr

ABSTRACT

Singing melody extraction is a task that tracks pitch contour of singing voice in polyphonic music. While the majority of melody extraction algorithms are based on computing a saliency function of pitch candidates or separating the melody source from the mixture, data-driven approaches based on classification have been rarely explored. In this paper, we present a classification-based approach for melody extraction on vocal segments using multi-column deep neural networks. In the proposed model, each of neural networks is trained to predict a pitch label of singing voice from spectrogram, but their outputs have different pitch resolutions. The final melody contour is inferred by combining the outputs of the networks and post-processing it with a hidden Markov model. In order to take advantage of the data-driven approach, we also augment training data by pitch-shifting the audio content and modifying the pitch label accordingly. We use the RWC dataset and vocal tracks of the MedleyDB dataset for training the model and evaluate it on the ADC 2004, MIREX 2005 and MIR-1k datasets. Through several settings of experiments, we show incremental improvements of the melody prediction. Lastly, we compare our best result to those of previous state-of-the-arts.

1. INTRODUCTION

Melody is a pitch sequence with which one might hum or whistle a piece of polyphonic music in an identifiable manner [10]. Among others, singing voice has been used as a main source of the melody, particularly in popular music. Thus, extracting melodies from singing voice can be used for not only music retrieval, for example, query-by-humming [5] or cover song identification [16] but also voice separation as a guide to inform the voice source.

A number of melody extraction algorithms, which can be applied for singing voice with an additional voice detection step, have been proposed so far and they are well summarized in [13]. The majority of the algorithms are based on computing a saliency function of pitch candidates or separating the melody source from the mixture. They typically return melody as a continuous pitch stream. On the other hand, data-driven approaches based on classification, which categorizes melody into a finite set of pitch labels, have rarely been explored. An early work by Ellis and Poliner used a support vector machine classifier to predict a pitch label from spectrogram [7]. Recently, Bittner et. al. proposed a method using a random forest classifier that predicts a pitch contour from highly hand-crafted features [3]. To the best of our knowledge, no other attempts have been made so far.

This scarcity of classification-based approach might be attributed to the following limitations. First, the extracted melody is supposed to be quantized by the pitch categorization (e.g. semitone unit in [7]). While this discrete outcome may be useful for some applications that require a MIDI-level pitch notation, it loses detailed information about singing styles, for example, vibrato or note-to-note transition patterns. Second, the data-driven approach typically requires sufficient labeled training data to achieve good performance. Finer pitch resolutions may need even more training data and possibly more complicated classifiers that can handle it.

In this paper, we address these limitations of the classification-based approach using multi-column deep neural networks (MCDNN). In the proposed model, each of DNN is trained to predict a pitch label of singing voice with different pitch resolutions. The outputs of the networks are combined and post-processed with a hidden Markov model to produce the final melody contour. Given a single DNN and training data, we observed that performance is inversely proportional to pitch resolutions. By combining the multiple DNNs, we show that the model can achieve higher pitch resolutions and better performance at the same time. In addition, we augment the training data by pitch-shifting the audio content and modifying the pitch label accordingly. We show that this is an effective technique to improve classification performance of the model.

2. RELATED WORK

The MCDNN was originally devised as an ensemble method to improve the performance of DNN for image classification [4]. In this model, each column (or single DNN) share the same network configuration and training

© Sangeun Kum, Changheun Oh, Juhan Nam. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Sangeun Kum, Changheun Oh, Juhan Nam. "Melody Extraction on Vocal Segments Using Multi-Column Deep Neural Networks", 17th International Society for Music Information Retrieval Conference, 2016.

data. However, they are randomly initialized, and the input data may be preprocessed in different ways for each column. The predictions from all columns are averaged to produce the final output. The multi-column approach was applied to image denoising as well [1]. In this approach, each column is trained on a different type of noise, and the outputs are adaptively weighted to handle a variety of noise types. Our proposed model may pose half-way between these two approaches. Each column is trained to conduct a different role, having a different number of outputs. However, we combine the outputs with even weights as they are the same pitch quantity with different resolutions.

As aforementioned, classification-based melody extraction is rarely attempted. Among them, our proposed model is similar to the SVM approach by Ellis and Poliner [7] in that both of them predict a pitch label from spectrogram using a classifier and a hidden Markov model for post-processing. However, our model produces a finer pitch resolution. Also, we take advantage of deep neural networks, which recently has proved to be capable of having great performance with sufficient labeled data and computing power.

3. PROPOSED METHODS

3.1 Multi-Column Deep Neural Networks

Our architecture of the MCDNN is illustrated in Figure 1. Each of the DNN columns takes an odd-numbered spectrogram frames as input to capture contextual information from neighboring frames and predicts a pitch label at the center position of the context window. The DNNs are configured with three hidden layers and ReLUs for the non-linear function in common, but the output layers predict a pitch label with different resolutions. The lowest resolution is semitone, corresponding to the leftmost one. The next ones progressively have higher resolutions by two times (e.g. 0.5 semitones, 0.25 semitones, ...), thereby having as much pitch labels as the increased resolutions. Given the outputs of the columns, we compute the combined posterior as follows:

$$y^N_{MCDNN} = \prod_{i=1}^{N} (y^i_{DNN} + \epsilon) \quad (1)$$

where y^i_{DNN} corresponds to the prediction from i^{th} column DNN, and N corresponds to the number of total columns. We use multiplication in a maximum-likelihood sense, assuming that the column DNNs are independent. We add a small value, ϵ to prevent numerical underflow. Note that, before combining the predictions, those with lower resolutions are actually expanded by locally replicating each element so that the output sizes are the same for all columns. For example, the leftmost DNN in Figure 1, which predicts pitch in semitone, expands the output vector by a factor of 4. As a result, the merged posterior maintains the highest pitch resolution.

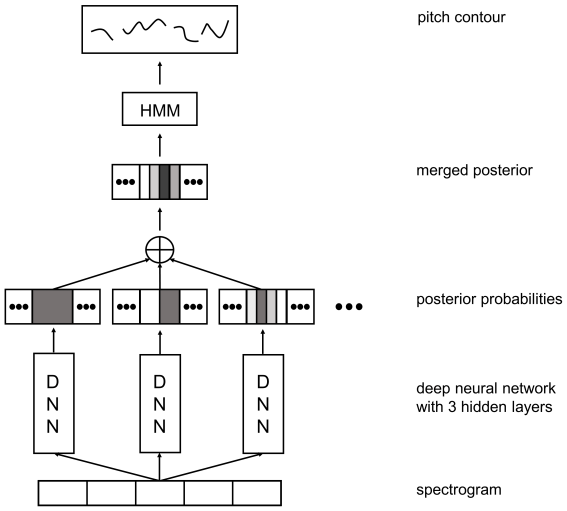


Figure 1: Block diagram of our proposed multi-column deep neural networks for singing melody extraction

3.2 Data Augmentation

Recent advances in deep learning are attributed to the availability of large-scale labeled data among others. Considering that melody-labeled public datasets are not much available, and manual labeling is laborious, it is desirable to augment existing datasets. In our experiments, we augment our training set by changing the global pitch of the audio content. Instead of pitch-shifting by resampling [10], which carries out time-stretching at the same time, we use a phase-vocoder method approach to achieve more natural transposition [9]. Pitch shifting proved to be an effective method of data augmentation for singing voice detection [15]. We will show that it works for singing melody extraction as well. On top of this, we also augment the training data by simply using an extra dataset that covers more music genres, as melody characteristics are quite discriminative over different music genres [14].

3.3 Temporal Smoothing by HMM

Although the MCDNN is trained to capture contextual information by taking multiple frames as input data, this may be limited to learn long-term temporal dependencies that appear on the pitch contours of singing voices. Also, the prediction is performed independently every time step. In order to incorporate the sequential structure further, we conduct temporal smoothing for the combined output of the MCDNN using HMM. We implemented the HMM, following the procedure in [7].

3.4 Singing Voice Detection

The MCDNN is trained with only voiced frames for pitch classification. Therefore, a separate singing voice detection step is necessary for the test phase. However, since singing voice detection itself is a challenging task and not

our main concern in this paper, we evaluate the test data using two scenarios. In the first scenario, we assume that a perfect singing voice detector is available so that we focus on the performance of our model only on voiced frames. In the second scenario, we use a simple energy-based singing voice detector introduced in [7]. The detector sums spectral energy between 200 Hz and 1800 Hz where the singing voice is likely to have a higher level than background music. The sum is normalized by the median energy in the band, and a threshold is used to determine the presence of singing voice. We expect that the performance of our model will range between the results from the two scenarios if a better singing voice detector is available.

4. DATASETS

4.1 Training Datasets

We use the RWC pop music database as our main training set [8]. It contains 100 popular songs with singing voice melody annotations. We divide the database into two splits, 85 songs for training and the remaining 15 songs for validation. In order to avoid bias by gender and the number of singers, we select the songs such that male/female singers and solo/chorus singing are evenly distributed over the training and validation sets. We also prevent the same singer's songs from being split over the two sets so that singer voices in the validation stage are never heard. In order to train the MCDNN more effectively, we augment the training set by applying pitch-shifting by $\pm 1, 2$ semitones. This increases the amount of the training set by five times. Also, we modify the corresponding pitch label accordingly.

Since the RWC database includes only pop music, the model trained on the set may not work well for other genres. We thus increase the size of training set and genre diversity by using 60 vocal tracks of the MedleyDB dataset as an additional training set [3].

4.2 Test Datasets

We examine our proposed model with three publicly available datasets: ADC2004, MIREX05, and MIR1k. Due to the limited accessibility to the datasets [1] and the limitation of our model that can handle singing voice only, we test them with several options. Specifically, the ADC2004 dataset includes some instrumental pieces where the melody is played by saxophones or other musical instruments. The MIREX05 dataset we obtained has only 13 out of the total 25 songs. Furthermore, only 9 of the 13 songs contain singing voice. For these reasons, we evaluate our model on all songs and those with singing voices separately for the two sets.

We report various evaluation metrics for melody extraction, including overall accuracy, raw pitch accuracy, raw chroma accuracy, voicing detection rate and voicing false alarm rate. We compute them using *mir_eval* [11], a Python library designed for objective evaluation in MIR tasks.

[1] We downloaded the ADC2004 and MIREX05 datasets from http://labrosa.ee.columbia.edu/projects/melody/ and the MIR1k dataset from https://sites.google.com/site/unvoicedsoundseparation/mir-1k

4.3 Preprocessing

We resample the audio files to 8 kHz and merge stereo channels into mono. We then compute spectrogram with Hann window of 1024 samples and hop size of 80 samples, and finally compress the magnitude by a log scale. Following the strategy in [7], we use only 256 bins from 0 Hz to 2000 Hz where the human singing voices have a relatively greater level than in other frequency bands with regard to background music.

5. EXPERIMENTS

Given the MCDNN model and training data, we conduct several experiments to figure out the effect of different settings in the model. In the followings, we describe options in training the MCDNN and the experiments.

5.1 DNN Training

We configure the DNN to have three hidden layers, each with 512, 512 and 256 units, and ReLUs for the nonlinear function. For the output layer, we use the sigmoid function instead of the softmax function, which is a typical choice in the categorical classification, because the sigmoid slightly worked better in our experiments. Thus, we use binary cross-entropy between the output layer and the one-hot representation of pitch labels as an objective function to minimize. The pitch labels cover from D2 to F#5 in semitone unit. The label vectors are expanded as pitch resolution increases. We initialize the weights with random values from the uniform distribution and optimize the objective function using RMSprop and 20% dropout for all hidden layers to avoid overfitting to the training set. For fast computing, we run the code using Keras [2], a deep learning library in Python, on a computer with two GPUs.

5.2 Context Size

Our model takes multiple frames of spectrogram as input to take contextual information into account. Our first experiment is to figure out an optimal size of the input for different pitch resolutions. For this experiment, we train a single-column DNN using one million examples from the RWC training set. Every training iteration, we randomly select a subset from the pool. We then verify classification accuracy using only voiced frames on the RWC validation set. Figure 2 shows the classification accuracy for a varying size of the spectrogram input. We experimented with multi-frame as inputs of DNN where the input data were taken from N neighbor spectrogram frames. The accuracy progressively increases up to 7 or 9 frames and then converge to a certain level. This is expected because pitch contours of singing voices usually have continuous curve patterns and this temporal features can be captured better by

[2] https://github.com/fchollet/keras

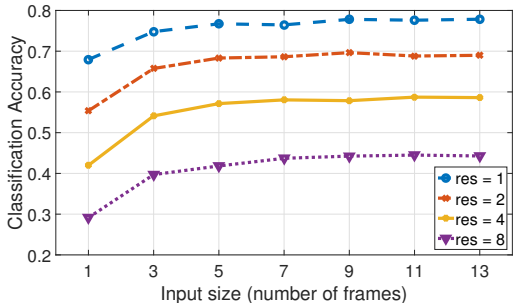


Figure 2: Classification accuracy on the validation set. "res=1" indicates pitch resolution in semitone unit. "res=2", "res=4", and "res=8" indicate progressively higher resolutions than semitone by a factor of 2.

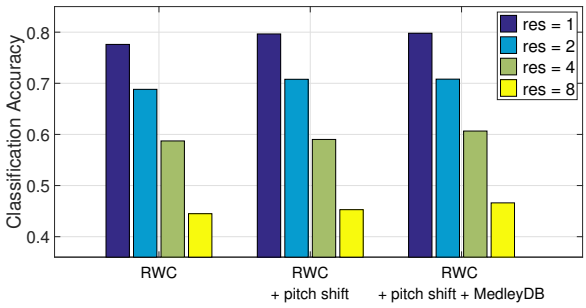


Figure 3: Classification accuracy on the validation set when the pitch-shifted versions of the RWC dataset and 60 vocal songs of the MedleyDB dataset are added in turn to the training set.

taking multiple frames. The result also shows that the validation accuracy is inversely proportional to the pitch resolution. That is, as the resolution increases, the accuracy drops quite significantly. This is also expected because the number of input data per label will decrease given the same training condition and also the accuracy criterion becomes more strict (i.e., slight missing between neighboring pitch labels could have been regarded as a correct prediction). For the following experiments, we fix the input size to 11 frames.

5.3 Data Augmentation

As described in Section 4.1, we augment the training set in two folds. One is by expanding the existing training set using pitch shifting and the other is by making up with another dataset, i.e., 60 songs including singing voices among the MedleyDB dataset. For this experiment, we train a single-column DNN using the increased training pool, specifically, six million examples from the augmented RWC training set and additional 200,000 examples or so from the MedleyDB songs. Again, we verify classification accuracy using only voiced frames on the RWC validation set.

Figure 3 shows the classification accuracy for a varying size of pitch resolution when the pitch-shifted RWC data and MedleyDB data are added to the training data pool in turn. Overall, the accuracy increases by 2 to 3 % with the additional sets. An interesting result is that, with the pitch-shifted data, the accuracy increases more when pitch resolution is low (1 or 2) and, with the additional MedleyDB songs, the accuracy increases more when pitch resolution is high (4 or 8). This is probably because the RWC data is pitch-shifted in semitone units and so technically increases data with low pitch resolutions whereas strong vibrato voices in the opera songs included in the MedleyDB dataset increase data with high pitch resolutions relatively more.

5.4 Single-column vs. Multi-column

As shown in the previous experiments, the classification accuracy is inversely proportional to the pitch resolution in the single-column DNN (SCDNN). That is, as the resolution becomes finer, the classification accuracy decreases, and vice versa. The MCDNN was devised from this empirical result, hoping to achieve both high accuracy and high pitch resolution simultaneously by using the SCDNN with different pitch resolutions together. In this experiment, we validate the idea by comparing the SCDNN and two different combinations of MCDNN. In particular, we evaluate them on the three test sets (ADC2004, MIREX05 and MIR1k), assuming the voiced frames are perfectly detected (the first singing voice detection scenario in Section 3.4).

Figure 4 displays the raw pitch accuracy (RPA) and raw chroma accuracy (RCA). Note that we evaluate the models on the ADC2004 and MIREX05 datasets separately for all songs including instrumental pieces and a subset excluding them (for the latter, the dataset name is suffixed with "vocal"). Overall, the MCDNN improves the melody extraction accuracies. An interesting result is that the MCDNN increases the accuracies on the sets with singing voices quite significantly (about 5 % in RPA and RCA on the MIREX05-vocal) whereas it can be even worse than the SCDNN when instrumental pieces are included. This is actually expected because our model is trained only using voiced frames. This indicates that our model is a specialized melody extraction algorithm that works only on music including singing voices. Comparing the two MCDNN models, there is no significant difference in performance. Thus, the simpler model (the 1-2-4 MCDNN) seems to be a better choice.

5.5 HMM-based Postprocessing

We conduct the Viterbi decoding based on a HMM model for temporal smoothing of the combined prediction. We estimate the prior probabilities and transition matrix from ground-truth of the training set. We then use the prediction of whole tracks as posterior probabilities. Table 1 shows the results as performance increments after applying the Viterbi decoding for the 1-2-4 MCDNN on the test sets.

5.6 A Case Example of Singing Melody Extraction

Our proposed model is capable of predicting temporally smooth pitch contours by using multi-resolution pitch la-

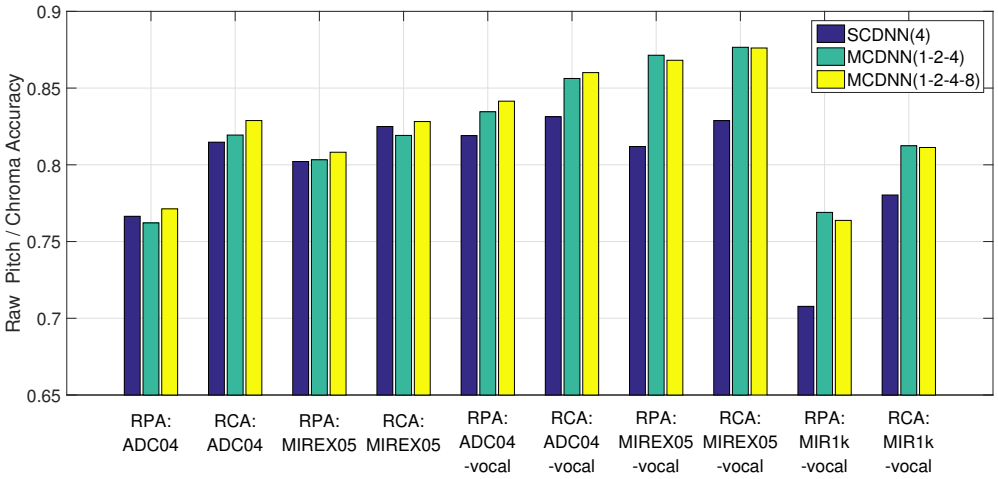


Figure 4: Raw pitch accuracy (RPA) and raw chroma accuracy (RCA) on the ADC2004, MIREX2005 and MIR1K dataset that compare one SCDNN and two different MCDNNs. The "vocal" suffix indicates their subsets that include songs with vocals. Here we assume that we have a perfect voice detector to focus on accuracy on voiced frames.

	without HMM		with HMM	
Dataset	**RPA**	**RCA**	**RPA**	**RCA**
ADC2004	0.749	0.806	0.762	0.816
ADC2004-vocal	0.827	0.852	0.835	0.856
MIREX05	0.801	0.817	0.803	0.817
MIREX05-vocal	0.869	0.875	0.871	0.877
MIR1k	0.766	0.813	0.769	0.813

Table 1: Performance increment by HMM-based smoothing on the 1-2-4 MCDNN.

bels, and the capability is supported more by the augmented datasets. Here we verify it by illustrating an example of singing melody extraction. We selected an opera song from the ADC2004 dataset because the singing voices have dynamic pitch motions such as strong vibrato. Figure 5 shows the results from three different melody extraction models. The left one is from the SCDNN with a pitch resolution of 4 (i.e. 1/4 semitone) and trained only with the RWC dataset. The middle one is from the same SCDNN but trained with additional pitch-shifted RWC dataset and MedleyDB dataset. The right one is from the 1-2-4 MCDNN that has the three pitch resolutions. Comparing the first two models, the additional songs help tracking the vibrato but the second model still misses the whole excursion. With the additional resolutions, the MCDNN makes further improvement, tracking the pitch contours quite precisely.

5.7 Comparison to State-of-the-art Methods

We compare our proposed method with state-of-the-art algorithms on the three test datasets in Table 2. The compared algorithms are all based on pitch saliency [2, 6, 12]. The evaluation metrics include overall accuracy (OA), raw pitch accuracy (RPA), raw chroma accuracy (RCA), voice recall (VR) and voice false alarm (VFA). As mentioned

Algorithm	**OA**	**RPA**	**RCA**	**VR**	**VFA**
Arora [2]	0.690	0.814	0.859	0.765	0.235
Dressler [6]	0.853	0.883	0.889	0.901	0.158
Salamon [12]	0.735	0.763	0.787	0.805	0.151
MCDNN(all)	0.655	0.703	0.759	0.874	0.469
MCDNN(vocal)	0.731	0.758	0.783	0.889	0.412

(a) ADC2004

Algorithm	**OA**	**RPA**	**RCA**	**VR**	**VFA**
Arora [2]	0.634	0.692	0.765	0.810	0.344
Dressler [6]	0.715	0.770	0.806	0.831	0.300
Salamon [12]	0.657	0.676	0.762	0.773	0.263
MCDNN(all)	0.616	0.733	0.752	0.894	0.585
MCDNN(vocal)	0.684	0.776	0.786	0.870	0.490

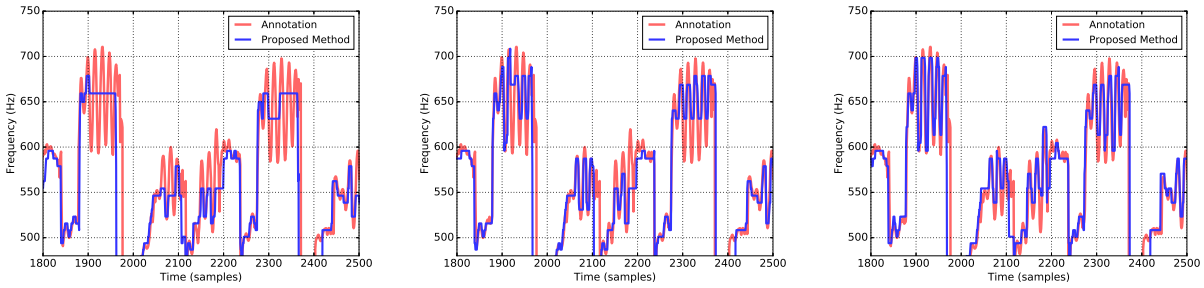
(b) MIREX05

Algorithm	**OA**	**RPA**	**RCA**	**VR**	**VFA**
MCDNN(vocal)	0.613	0.726	0.770	0.934	0.658

(c) MIR-1K

Table 2: Melody extraction results on three test datasets. In this evaluation, we used a simple energy-based voice detector for fair comparison.

in Section 4.2, we have only 13 songs in the MIREX05 dataset. Since we use a simple energy-based voice detector (the second singing voice detection scenario in Section 3.4), the results of our model were not very impressive. However, even with it, the accuracies are quite comparable to some of the algorithms when the test sets include singing vocals. Also, from Figure 4, we can see the RPA and RCA when we have a perfect voice detector. This shows that the accuracies significantly increase, being comparable to the top-notch one.



(a) SCDNN with a pitch resolution of 4 trained with the RWC dataset

(b) SCDNN with a pitch resolution of 4 trained with the additional pitch-shifted RWC dataset and MedleyDB dataset

(c) 1-2-4 MCDNN trained with the additional pitch-shifted RWC dataset and MedleyDB dataset

Figure 5: A case example of melody extraction on an opera song using different models and training data

6. CONCLUSIONS

In this paper, we proposed a novel classification-based melody extraction algorithm on vocal segments using the multi-column deep neural networks. We showed how the data-driven approach can be improved by different settings of the model such as input size, data augmentation, use of multi-column DNN with different pitch resolutions and HMM-based smoothing. The limitation of this model is that it works well only for singing voice because we trained it only with songs where vocals lead the melody. However, this also indicates that our model can be improved to a general melody extractor if a sufficient amount of instrumental pieces are included in the training sets. We compared our model to previous state-of-the-arts. Since we used a simple energy-based singing voice detector, the performance of our model has limitations. However, the results show that, with a better voice detector, our model can be improved further.

7. ACKNOWLEDGMENT

This work was supported by Korea Advanced Institute of Science and Technology (Project No. G04140049), National Research Foundation of Korea (Project No. N01150671) and BK21 Plus Postgraduate Organization for Content Science.

8. REFERENCES

[1] Forest Agostinelli, Michael R Anderson, and Honglak Lee. Adaptive multi-column deep neural networks with application to robust image denoising. In *Advances in Neural Information Processing Systems 26*, pages 1493–1501, 2013.

[2] Vipul Arora and Laxmidhar Behera. On-line melody extraction from polyphonic audio using harmonic cluster tracking. *Audio, Speech, and Language Processing, IEEE Transactions on*, 21(3):520–530, 2013.

[3] Rachel M Bittner, Justin Salamon, Slim Essid, and Juan P Bello. Melody extraction by contour classification. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR*, 2015.

[4] Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.

[5] Roger B Dannenberg, William P Birmingham, George Tzanetakis, Colin Meek, Ning Hu, and Bryan Pardo. The musart testbed for query-by-humming evaluation. *Computer Music Journal*, 28(2):34–48, 2004.

[6] Karin Dressler. An auditory streaming approach for melody extraction from polyphonic music. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR*, pages 19–24, 2011.

[7] Daniel P. W. Ellis and Graham E Poliner. Classification-based melody transcription. *Machine Learning*, 65(2):439–456, 2006.

[8] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. Rwc music database: Popular, classical, and jazz music databases. In *Proceedings of the 3rd International Conference on Music Information Retrieval, ISMIR*, pages 287–288, 2002.

[9] Jean Laroche. *Applications of Digital Signal Processing to Audio and Acoustics*, chapter Time and Pitch Scale Modification of Audio Signals, pages 279–309. Springer US, Boston, MA, 2002.

[10] Graham E Poliner, Daniel P. W. Ellis, Andreas F Ehmann, Emilia Gómez, Sebastian Streich, and Beesuan Ong. Melody transcription from music audio:

Approaches and evaluation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(4):1247–1256, 2007.

[11] Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis. mir eval : A transparent implementation of common mir metrics. In *Proceedings of the 15th International Conference on Music Information Retrieval, ISMIR*, 2014.

[12] Justin Salamon and Emilia Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(6):1759–1770, 2012.

[13] Justin Salamon, Eva Gomez, Daniel P. W. Ellis, and Gael Richard. Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *Signal Processing Magazine, IEEE*, 31(2):118–134, 2014.

[14] Justin Salamon, Bruno Rocha, and Emilia Gómez. Musical genre classification using melody features extracted from polyphonic music signals. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[15] Jan Schlüter and Thomas Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR*, pages 121–126, 2015.

[16] Joan Serra, Emilia Gómez, and Perfecto Herrera. Audio cover song identification and similarity: background, approaches, evaluation, and beyond. In *Advances in Music Information Retrieval*, pages 307–332. Springer, 2010.

Author Index