PROCEEDINGS

OF THE 14TH INTERNATIONAL SOCIETY FOR MUSIC INFORMATION RETRIEVAL CONFERENCE

November 4-8, 2013
Curitiba, Brazil

EDITED BY
ALCEU DE SOUZA BRITTO JR.
FABIEN GOUYON and
SIMON DIXON

ismir
CURITIBA • BRAZIL

Proceedings of the 14th Conference of the International Society for Music Information Retrieval (ISMIR)

Edited by:
Alceu de Souza Britto Junior
Fabien Gouyon
Simon Dixon

November 4th - 8th, Curitiba, Brazil

http://ismir2013.ismir.net/ismir2013/

ISMIR2013 is organized by the International Society for Music Information Retrieval.
Website: http://ismir2013.ismir.net/ismir2013/

Edited by:
Alceu de Souza Britto Junior
Fabien Gouyon
Simon Dixon

Copies may be ordered from:
Programa de Pós-Graduação em Informática (PPGIa)
Pontifícia Universidade Católica do Paraná
Rua Imaculada Conceição. 1155 - Prado Velho
Curitiba - PR - Brasil
email: secretaria@ppgia.pucpr.br
ISBN 978-0-615-90065-0

Sponsors

Support

Conference Committee

Conference Chairs
Alessandro L. Koerich (Pontifícia Universidade Católica do Paraná, Brazil)
George Tzanetakis (University of Victoria, Canada)

Program Chairs
Alceu S. Britto Jr. (Pontifícia Universidade Católica do Paraná, Brazil)
Fabien Gouyon (Instituto de Engenharia de Sistemas e Computadores do Porto, Portugal)
Simon Dixon (Queen Mary University of London, UK)

Music Chair
Jônatas Manzolli (Universidade de Campinas, Brazil)

Financial and Local Arrangments Chair
Luiz S. Oliveira (Universidade Federal do Paraná, Brazil)

Tutorial Chair
Carlos N. Silla Jr. (Universidade Tecnológica Federal do Paraná, Brazil)

Late-break/Demos Chair
Mohamed Sordo (Universitat Pompeu Fabra, Spain)

Music Committee
Carlos Guedes (New York University Abu Dhabi, UAE)
Mikhail Malt (Institut de Recherche et Coordination Acoustique/Musique, France)
Silvio Ferraz (Universidade de Campinas, Brazil)

Program Committee

Jean-Julien Aucouturier (Institut de Recherche et Coordination Acoustique/Musique, France)
Stephan Baumann (German Research Center for Artificial Intelligence, Germany)
Juan Pablo Bello (New York University, USA)
Darrell Conklin (Universidad del País Vasco, Spain)
Sally Jo Cunningham (University of Waikato, New Zealand)
Matthew Davies (Institute for Systems and Computer Engineering of Porto, Portugal)
J. Stephen Downie (University of Illinois, USA)
Dan Ellis (Columbia University, USA)
Rebecca Fiebrink (Princeton University, USA)
Arthur Flexer (Austrian Research Institute for Artificial Intelligence, Austria)
José Tuti Fornari (University of Campinas, Brazil)
Masataka Goto (National Institute of Advanced Industrial Science and Technology, Japan)
Emilia Gómez (Universitat Pompeu Fabra, Spain)
Perfecto Herrera (Universitat Pompeu Fabra, Spain)
André Holzapfel (Bogazici University, Turkey)
Ozgur Izmirli (Connecticut College, USA)
Anssi Klapuri (Tampere University of Technology, Finland)
Peter Knees (Johannes Kepler University, Austria)
Paul Lamere (Echonest, USA)
Luis Gustavo Martins (Catholic University of Porto, Portugal)
Meinard Mueller (Max-Planck-Institut für Informatik, Germany)
Teresa Nakra (The College of New Jersey, USA)
Nicola Orio (University of Padova, Italy)
Geoffroy Peeters (Institut de Recherche et Coordination Acoustique/Musique, France)
Marcelo Queiroz (University of São Paulo, Brazil)
Andreas Rauber (Vienna University of Technology, Austria)
Robert Rowe (New York University, USA)
Markus Schedl (Johannes Kepler University, Austria)
Erik Schmidt (Pandora Internet Radio, USA)
Xavier Serra (Universitat Pompeu Fabra, Spain)
Joan Serrà (Artificial Intelligence Research Institute, Spain)
Douglas Turnbull (Ithaca College, USA)
Emmanuel Vincent (Institut National de Recherche en Informatique et en Automatique, France)
Anja Volk (Utrecht University, The Netherlands)
Ye Wang (National University of Singapore, Singapore)
Gerhard Widmer (Johannes Kepler University, Austria)
Geraint Wiggins (Queen Mary University of London, UK)

Reviewers

Abdallah, Samer
Abesser, Jakob
Ahonen, Teppo
Alluri, Vinoo
Anagnostopoulou, Christina
Arzt, Andreas
Bainbridge, David
Barbancho, Isabel
Barbancho, Ana Maria
Barthet, Mathieu
Battenberg, Eric
Bay, Mert
Benetos, Emmanouil
Bertin-Mahieux, Thierry
Bimbot, Frédéric
Bohak, Ciril
Bonafonte, Antonio
Bozkurt, Baris
Burgoyne, Ashley
Burred, Juan Jose
Böck, Sebastian
Caceres, Juan Pablo
Caetano, Marcelo
Cambouropoulos, Emilios
Cano, Estefania
Cartwright, Mark
Casey, Michael
Cemgil, Taylan
Chen, Shuo
Cho, Taemin
Christensen, Mads
Chuan, Ching-Hua
Ciglar, Miha
Collecchia, Gina
Collins, Tom
Cont, Arshia
Cotton, Courtenay
Coviello, Emanuele
Crawford, Tim
Cuthbert, Michael Scott
David, Bertrand
de Haas, Bas
Degara, Norberto
Deliege, François
Dessein, Arnaud
Devaney, Johanna
Dittmar, Christian
Dzhambazov, Georgi
Eck, Douglas
Ehmann, Andreas
Emiya, Valentin
Esquef, Paulo
Essid, Slim
Ewert, Sebastian
Fazekas, György
Fillon, Thomas
Fitzgerald, Derry
Flaig, Nicole
Flossmann, Sebastian
Font, Frederic
Franklin, Judy
Fujinaga, Ichiro
Gaertner, Daniel
Gasser, Martin
Gedik, Ali Cenk
Gine, Enric
Gingras, Bruno
Glennon, Aron
Goebl, Werner
Grachten, Maarten
Griffin, Garth
Grill, Thomas
Grunberg, David
Grund, Cynthia
Guaus, Enric
Gulati, Sankalp
Gurevich, Michael
Hamel, Philippe
Hankinson, Andrew
Hanna, Pierre
Harte, Chris
Hegde, Shantala
Hirata, Keiji
Hockman, Jason
Hoffman, Matt
Honing, Henkjan
Honingh, Aline
Hu, Diane
Hu, Xiao
Hu, Yajie
Huang, Anna
Humphrey, Eric
Huron, David
Iazzetta, Fernando
Imbrasaite, Vaiva
Ito, Akinori
Itoyama, Katsutoshi
Janata, Petr
Janer, Jordi
Jang, Jyh-Shing Roger
Jehan, Tristan
Jensen, Kristoffer
Jun, Sanghoon
Kaiser, Florian
Kameoka, Hirokazu
Kashino, Kunio
Katayose, Haruhiro
Keller, Damián
Kepper, Johannes
Kim, Minje
Knopke, Ian
Koduri, Gopala
Koenigstein, Noam
Kon, Fabio
Korzeniowski, Filip
Krebs, Florian
Kurth, Frank
Langlois, Thibault
Laplante, Audrey
Lartillot, Olivier
Laurier, Cyril
Law, Edith
Lee, Honglak
Lee, Kyogu
Lee, Jin Ha
Lehmann, Andreas
Lehtonen, Heidi-Maria
Lemstrom, Kjell
Lewis, David
Liem, Cynthia
Lobato Oliveira, Joao
Loviscach, Jörn
Lukashevich, Hanna
Maia Jr., Adolfo
Mandel, Michael
Marolt, Matija
Marsden, Alan
Martorell, Agustin

Mauch, Matthias
Mayer, Rudolf
McFee, Brian
McKay, Cory
McVicar, Matt
Mesaros, Annamaria
Miron, Marius
Montecchio, Nicola
Moore, Josh
Moroni, Artemis
Murthy, Hema
Mysore, Gautham
Müllensiefen, Daniel
Nagano, Hidehisa
Nakano, Tomoyasu
Nichols, Eric
Nieto, Oriol
Ogihara, Mitsunori
Ohishi, Yasunori
Ono, Nobutaka
Pachet, François
Paiva, Rui Pedro
Papadopoulos, Hélène
Pardo, Bryan
Park, Tae Hong
Paulus, Jouni
Pauwels, Johan
Pauws, Steffen
Percival, Graham
Pertusa, Antonio
Pestana, Pedro
Pikrakis, Aggelos
Pimenta, Marcelo
Prockup, Matthew
Pugin, Laurent
Quinn, Ian
Raczynski, Stanislaw
Rafii, Zafar
Ramirez, Miguel
Ramona, Mathieu
Rao, Preeti
Rebelo, Ana
Reiss, Joshua
Rhodes, Christophe
Rizo, David
Robertson, Andrew
Robine, Matthias
Rocamora, Martin
Rodríguez López, Marcelo
Roland, Perry
Roma, Gerard
Salamon, Justin
Sapp, Craig Stuart
Sargent, Gabriel
Sarroff, Andy
Schirru, Rafael
Schlueter, Jan
Schnitzer, Dominik
Schuller, Bjoern
Scott, Jeff
Sethares, William
Seyerlehner, Klaus
Simon, Ian
Sioros, George
Slaney, Malcolm
Smaragdis, Paris
Smith, Lloyd
Smith, Jordan
Song, Yading
Sonnleitner, Reinhard
Sordo, Mohamed
Srinivasamurthy, Ajay
Stark, Adam
Stober, Sebastian
Stowell, Dan
Sturm, Bob
Temperley, David
Tjoa, Steve
Toiviainen, Petri
Toussaint, Godfried
Tsai, Wei-Ho
Unal, Erdem
Urbano, Julián
Vaizman, Yonatan
Van Balen, Jan
Van Kranenburg, Peter
Veltkamp, Remco
Virtanen, Tuomas
Wang, Xinxi
Wang, Ju-Chiang
Wang, Ge
Weinstein, Eugene
Weiss, Ron
Weninger, Felix
Weyde, Tillman
Whalley, Ian
Wright, Matthew
Yang, Yi-Hsuan
Yoshii, Kazuyoshi
Zagorski-Thomas, Simon
Zapata, Jose Ricardo

Contents

Preface

We would like to welcome all participants of the 14th International Society for Music Information Retrieval Conference (ISMIR 2013) in the Bourbon Curitiba Convention Hotel. It is located in the downtown area of Curitiba, capital of the Parana State and one of the most important cities of Southern Brazil, being a cultural, political and economic center. This year's edition of the conference is organized by the Pontifícia Universidade Católica do Paraná (PUCPR), in partnership with the Universidade Federal do Paraná (UFPR) and the Universidade Tecnológica Federal do Paraná (UTFPR).

The present volume contains the complete manuscripts of all the peer-reviewed papers presented at ISMIR 2013. A total of 252 abstracts were entered in the reviewing system before the deadline, out of which 215 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-blind (i.e., both the authors and the reviewers were anonymous) with a two-tier model of review committee (i.e. including reviewers and Program Committee members). Reviewers and PC members were able to bid for papers. Each paper was assigned to a PC member, who could then recommend reviewers. Reviewers were assigned to papers based on topic preferences, bids and PC member assignments. After the review phase, PC members and reviewers entered a discussion phase aiming at homogenizing acceptance vs. rejection decisions. This discussion phase, which did not provide any further interaction with authors, replaced the rebuttal phase as used in some recent ISMIR conferences.

Final acceptance decisions were based on 855 reviews and meta-reviews written by 282 reviewers and PC members. From the 215 reviewed papers, 99 papers were accepted resulting in an acceptance rate of 46%. One paper was withdrawn as the authors were unable to attend and present the paper, leaving 98 papers in this proceedings. Compared to previous ISMIR conferences, it is possible to observe a slight reduction in the total number of accepted papers, which led to a higher competitiveness. The table shown in the next page summarizes the ISMIR publication statistics over the last years.

The mode of presentation was determined after the accept/reject decisions. From the 98 papers, 31 papers were chosen for oral presentation based on the quality, topic and breadth of appeal of the work, whereas 67 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 2 sessions per day, the same posters being presented in the morning and in the afternoon of a given day. Poster presenters have the opportunity (in 1 minute and 1 slide) to introduce their posters during a plenary session (so-called "poster-craze") and "tease" the whole audience, inviting to a later, more personal, scientific discussion around their poster.

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-break/demo session. Two 3-hour tutorials are presented in parallel during Monday morning, and two in parallel during the afternoon. In the morning, François Pachet and Giordano Cabral present a tutorial on the wonderful facets of "Brazilian guitar", while Mohamed Sordo and Emanuele Coviello cover the challenges in building an automatic music tagger. In the afternoon, Slim Essid presents a tutorial on conditional random fields with application to music analysis, while Philippe Hamel, Eric J. Humphrey, and Erik M. Schmidt present a tutorial on deep learning in MIR.

The ISMIR program also features one distinguished keynote speaker on Tuesday afternoon. François Pachet, from Sony Computer Science Lab (Paris, France), describes a new approach to content authoring tools based on style manipulation as a creative device. The proposed approach is based on letting users explicitly manipulate style, as a computational object.

On Sunday and Tuesday nights, 1-hour concerts will be held in the main venue. The aim of the ISMIR 2013 music program is two-fold: to encourage the use of MIR techniques in the creation of new music and to explore music that can suggest novel ideas for research in the MIR field. A call for participation was issued which resulted in the submission of 10 music pieces. From these pieces, 5 were selected, which are presented in the ISMIR concerts.

On Wednesday afternoon, an Industrial Panel will be held, which the focus is to discuss the relationship between academic MIR research and industry practice. For this, we have invited key representatives of some of the companies that have been actively utilizing MIR techniques in their business. The panelists will discuss how their companies have utilized MIR in the past, what MIR topics are they currently working on, and suggest directions for future research. The plan is to leave enough time for open ended discussion and questions.

Location	Oral	Poster	Total Papers	Total Pages	Total Authors	Unique Authors	Pages/ Paper	Authors/ Paper	U. Authors/ Paper
Plymouth	19	16	35	155	68	63	4.4	1.9	1.8
Indiana	25	16	41	222	100	86	5.4	2.4	2.1
Paris	35	22	57	300	129	117	5.3	2.3	2.1
Baltimore	26	24	50	209	132	111	4.2	2.6	2.2
Barcelona	61	44	105	582	252	214	5.5	2.4	2
London	57	57	114	697	316	233	6.1	2.8	2
Victoria	59	36	95	397	246	198	4.2	2.6	2.1
Vienna	62	65	127	486	361	267	3.8	2.8	2.1
Philadelphia	24	105	105	630	296	253	6	2.8	2.4
Kobe	38	85	123	729	375	292	5.9	3	2.4
Utrecht	24	86	110	656	314	263	6	2.9	2.4
Miami	36	97	133	792	395	322	6	3	2.4
Porto	36	65	101	606	324	264	6	3.2	2.6
Curitiba	**31**	**67**	**98**	**587**	**395**	**236**	**5.9**	**3**	**2.4**

On Friday morning, the plenary and poster sessions of the eighth running of the Music Information Retrieval Evaluation eXchange (MIREX 2013) are presented. It is organized by the International Music Information Retrieval Systems Evaluation Laboratory (IMIRSEL) of the Graduate School of Library and Information Science (GSLIS), University of Illinois at Urbana-Champaign (UIUC).

The last afternoon of the conference is dedicated to the Demos/Late-breaking (D&L) track. The D&L track has become a popular and integral part of ISMIR conferences. Like in the last ISMIR, the format of the D&L is following an "unconference" style: there will not be any formal submission nor peer-review. The D&L program is built by participants, both before the event on an open website (http://ismir2013.wikispaces.com/), and even during the conference. Instead of focusing on individuals reporting in a one-to-many way on their personal accomplished research, D&L focuses on capturing those informal yet often insightful and original ideas that typically emerge in collaborative settings at ISMIR (e.g. at coffee breaks, hallways, etc).

As for social gathering, lunches and coffee will be served every day from Monday to Friday. ISMIR 2013 includes in the registration fee a reception on Sunday at 7pm and a banquet on Thursday night, including "reasonable" amounts of Brazilian beer and "caipirinha".

We are very proud to present you the proceedings of ISMIR 2013. The conference program was made possible thanks to the hard work of many people including the members of the organization, PC members and reviewers. Special thanks go to this year's sponsors and supporting institutions: CAPES, CNPq, Google, PUCPR, UFPR and UTFPR. Last, but not least, the ISMIR 2013 program is possible only thanks to the numerous contributions of the MIR community in response to our call for participation. The biggest acknowledgment therefore goes to you, the authors, researchers and participants of this conference.

Alessandro L. Koerich (PUCPR, Brazil)
George Tzanetakis (UVic, Canada)
Conference Chairs

Alceu S. Britto Jr. (PUCPR, Brazil)
Fabien Gouyon (INESC TEC, Portugal)
Simon Dixon (QMUL, UK)
Program Chairs

Keynote Talk

STYLE MANIPULATION AS A CREATIVE DEVICE

François Pachet
Sony Computer Science Lab
Paris, France

Abstract

Creative artifacts are often obtained by combining ideas, patterns, ways of doing - in a word, styles - to new configurations or situations. We describe a new approach to content authoring tools based on this idea. The approach is based on letting users explicitly manipulate style, as a computational object. In such a context, one key technical problem is to generate sequences that imitate a style, while satisfying arbitrary constraints, coming from the domain of study. We describe a novel sequence generation approach based on an old statistical tool: Markov chains. We show that it is possible to explore the complete set of sequences that a Markov model can generate, using combinatorial optimization techniques. We show that the addition of even simple constraints biases the initial Markov model in interesting ways, which are not fully understood. We describe some recent results in constrained Markov generation which pave the way for novel and exciting style manipulation applications, in music composition, improvisation and literary text writing, and give a few examples developed within the Flow Machines project.

Biography

François Pachet was trained in engineering, computer science and artificial intelligence (University of Paris 6) and is also a jazz musician. His research addresses issues in music interaction and production. He recently developed several jazz generation systems including Virtuoso (a system that generates virtuoso bebop improvisations) and VirtualBand (a system that generates adaptive jazz accompaniments). He recently obtained an ERC Advanced Grant to develop a new generation of music and text generation tools (and is looking for motivated, talented people to join his team in Paris). He was also trained in classical guitar (Ecole Normale de Musique de Paris), Baroque harmony (Conservatoire de Paris X) and jazz (Berklee school). He plays jazz guitar and composes jazz and pop music. He has 2 albums in progress. He learned to play Brazilian guitar by studying with Roland Dyens, and by listening endlessly to albums by Baden Powell.

Tutorials

Tutorial 1 - WHY IS BRAZILIAN GUITAR INTERESTING?

François Pachet
Sony Computer Science Lab
Paris, France

Giordano Cabral
Daccord Music Software and the MusiGames Studio
Brazil

Abstract

Following the ISMIR 2012 tutorial "Why jazz is interesting?" this tutorial will give an overview of the wonderful facets of "Brazilian guitar" for a MIR audience. Both presenters are accomplished Brazilian guitar players, and the tutorial will include many live demonstrations of the concepts addressed. Brazilian guitar is not a genre per se, nor a style, not even a specific instrument. But almost every Brazilian knows how to play guitar in a certain way: rich chords, complex rhythms, harmonious melodies. Almost every Brazilian also knows hundreds of songs and this shared repertoire plays a big role in shaping Brazilian culture in general. Consequently, Brazilian guitar is a key instrument in many genres of so-called "MPB" (Música Popular Brasileira for Brazilian Popular Music), including choros, bossa nova (of course), samba, and all the developments of an incredibly lively musical country. However, it is still a hard task to learn how to play Brazilian guitar and understand its basic principles. Its intrinsic rules are still ill studied, and poorly formalized. This tutorial will attempt to explain why Brazilian guitar is so fascinating. Like the preceding tutorial on jazz, it will be delivered from a "musician" viewpoint, but targeted at a scientifically advanced audience of MIR people. The main assumption of this tutorial (hopefully to become a series) is that in-depth knowledge of specific musical genres is a prerequisite to build the next generation of MIR systems.

Biographies

François Pachet was trained in engineering, computer science and artificial intelligence (University of Paris 6) and is also a jazz musician. His research addresses issues in music interaction and production. He recently developed several jazz generation systems including Virtuoso (a system that generates virtuoso bebop improvisations) and VirtualBand (a system that generates adaptive jazz accompaniments). He recently obtained an ERC Advanced Grant to develop a new generation of music and text generation tools (and is looking for motivated, talented people to join his team in Paris). He was also trained in classical guitar (Ecole Normale de Musique de Paris), Baroque harmony (Conservatoire de Paris X) and jazz (Berklee school). He plays jazz guitar and composes jazz and pop music. He has 2 albums in progress. He learned to play Brazilian guitar by studying with Roland Dyens, and by listening endlessly to albums by Baden Powell.

Giordano Cabral is the founder of Daccord Music Software and the MusiGames Studio. Daccord was recently acknowledged by FINEP as one of the most innovative companies in Brazil. The company has released a dozen game titles and musical apps, including some on the Top Charts of Apple's App Store. Giordano was also nominated by INFO Exame magazine for the entrepreneur of the year 2011 award. He is a computer music researcher, with PhD from the Université Pierre et Marie Curie (Paris VI). He is an associate professor at UFRPE, integrates the Special Committee of Computer Music of the Brazilian Computer Science Society and coordinates the Research Group of Music, Technology, Interaction and Creativity (MUSTIC). He is also an accomplished musician (guitar, drums, singing) and performs regularly in Brazil.

Tutorial 2 - MUSIC AUTOTAGGING

Mohamed Sordo
Music Technology Group
Universitat Pompeu Fabra
Barcelona, Spain

Emanuele Coviello
University of California, San Diego
Dept. of Electrical and Computer Engineering
USA

Abstract

Technology is revolutionizing the way in which music is distributed and consumed. As a result, millions of songs are instantly available to millions of people, on the Internet. This has created the need for novel music search and discovery technologies, to help users find a "mellow Beatles song" on a nostalgic night, "scary Halloween music" on October 31st, or address a sudden desire for "romantic jazz with saxophone and deep male vocals", without knowing an appropriate artist or song title. One important task in the realization of a music search engine is the automatic annotation of music with descriptive keywords, or tags, based on the audio content of the song. Music annotations can be used for a variety of purposes, such as searching for songs exhibiting specific qualities (e.g., jazz songs with female vocals and saxophone), or retrieval of semantically similar songs (e.g., for generating playlists). In this tutorial we look at the current state-of-the-art in content-based automatic music tagging. We cover the challenges in building an automatic music tagger, ranging from ground truth collection, statistical modeling, inference, and evaluation. We examine past and recent work, and discuss advantages and disadvantages of the various solutions. We finally put emphasis on the most recent discoveries.

Biographies

Dr. Mohamed Sordo is a Post-doctoral researcher at the Music Technology Group of the Universitat Pompeu Fabra in Barcelona, Spain. He obtained his PhD at the Music Technology Group in 2012, with a thesis entitled "Semantic annotation of music collections: a computational approach" [5], under the supervision of Dr. Xavier Serra and Dr. Oscar Celma, including a research stay at INESC Porto under the supervision of Dr. Fabien Gouyon. His thesis is mainly devoted to the topic of automatic music tagging. Mohamed's research areas involve music text/web mining, music information retrieval and machine learning. Mohamed has participated in a number of European funded projects, including Variazioni, Pharos and CompMusic, where he is currently involved, developing systems to extract semantically and musically meaningful information from web data.

Emanuele Coviello is Candidate in Philosopy in Electrical and Computer Engineering, at the University of California at San Diego (UCSD). Emanuele is part of Prof. Gert Lanckriet's Computer Audition Laboratory (CALab), where he researches about machine learning and applies his discoveries to music information retrieval. Emanuele received the "Premio Guglielmo Marconi Junior 2009" award, from the Guglielmo Marconi Foundation (Italy), and won the "2010 Yahoo! Key Scientific Challenge Program", sponsored by Yahoo!, and during Summer 2012 was an intern at Microsoft Research in Silicon Valley. Emanuele received the "Laurea Triennale" degree in information engineering and the "Laurea Specialistica" degree in telecommunication Engineering from the Università degli Studi di Padova, Italy, in 2006 and 2008, respectively.

Tutorial 3 - DEEP LEARNING IN MIR - DEMYSTIFYING THE DARK ART

Philippe Hamel
National Institute of Advanced Industrial Science and Technology (AIST)
Japan

Eric J. Humphrey
Music and Audio Research Lab
New York University
USA

Erik M. Schmidt
MET-lab
Drexel University
USA

Abstract
Following recent developments both within and beyond the music information retrieval (Music-IR) community, the popularity of deep learning as an approach to solving machine perception problems is noticeably growing. Despite this increasing interest, the prevailing sentiment seems to be that these methods are not well understood, uninterpretable, and generally impractical. As three successful practitioners of deep learning in Music-IR, we would like to offer a comprehensive tutorial featuring three complementary perspectives in an effort to begin addressing these issues.

Biographies
Philippe Hamel obtained his Ph.D. in computer science at Université de Montréal in 2012. His main research interests are machine learning and its applications to music information retrieval. One of his goals is to find ways to obtain better and richer representations of music audio. His recent work has been focused on artificial neural networks, deep learning and signal processing. He is interested in music information retrieval problems such as automatic tagging, music classification, music similarity and music recommendation. Prior to his work in computer science, Philippe studied physics at Université de Montreal where he obtained his masters degree in theoretical physics (2006) and his bachelor degree in mathematics and physics (2004). He is currently working at AIST, Japan as a postdoctoral researcher.

Eric J. Humphrey is a Ph.D candidate (ABD) in music technology at New York University, with an expected graduation in 2014. In the broadest sense, his research is ultimately motivated by the hunt for better artificial intelligence; currently, this takes the form of exploring machine learning, and particularly data-driven methods, in the context of music signal processing. Parallel to this ongoing work, he is also an advocate of feature learning and deep architectures in music informatics, both of which can be viewed as extensions to traditional approaches in the field. Though interested in general approaches more than specific applications, his research has focused on deep learning in the areas of timbre similarity and chord recognition. Before attending NYU, Eric earned a masters of science in Music Engineering Technology at the University of Miami (2009) and bachelors of science in Electrical Engineering at Syracuse University (2007). His masters thesis explored the application of signal processing in music therapy by developing systems to produce improved playlists for motor entrainment during recreational running. In addition to his academic pursuits, he has also held multiple independent consulting positions, is named as an inventor in several patent applications, and serves as student

member on the ISMIR Board.

Erik M. Schmidt is a Post-Doctoral Researcher in the Music and Entertainment Technology Laboratory (MET-lab) at Drexel University in Philadelphia, PA. He received his Ph.D. in Electrical Engineering from Drexel University in 2012, and also holds an MSEE from Drexel University and BSEE from Temple University. During his undergraduate career, Erik also worked for Aviom, Inc., a company in the market of digital audio networking technologies. Erik's research focuses on the automatic prediction of emotional content in musical signals. In seeking to understand the complex relationship between acoustic data and emotion space representations, his work focuses on the development of graphical models and deep learning techniques for identifying these relationships, and how they evolve over time in synchrony with one another. Using deep learning approaches, his work has developed multiple methods for identifying deep structured connections between the affective and acoustic domains. These networks can be used for emotion prediction, feature extraction, and perhaps even for providing a new understanding of human emotion in general.

Tutorial 4 - CONDITIONAL RANDOM FIELDS WITH APPLICATION TO MUSIC ANALYSIS

Slim Essid
Telecom ParisTech
France

Abstract

Conditional Random Fields (CRF) (Lafferty, 2001) 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. While they have been receiving a booming interest from researchers in text and natural language processing (Taskar, 2002; Settles, 2004; Lavergne, 2010), computer vision (He, 2004, Wang, 2006; Liu, 2008; Chang, 2009; Mori, 2009; Gao, 2012) and speech analysis (Gunawardana, 2005; Reiter, 2007; Morris, 2008; Hong, 2010), they remain to date rarely used in the MIR community despite a few remarkable contributions (Corey, 2007; Joder, 2011; Schmidt, 2011; Joder, 2013). Hence, this tutorial aims at introducing the CRF framework to MIR researchers, providing the theoretical foundations, methods and algorithms for CRF-based modelling, training and inference; while focusing on the aspects that are particularly relevant to music analysis applications and discussing concrete case-studies.

Biography

Slim Essid is an Associate Professor at the Department of Image and Signal Processing of Telecom ParisTech with the Audio & Waves group. He received the state engineering degree from the École Nationale d'Ingénieurs de Tunis in 2001; the M.Sc. (D.E.A.) degree in digital communication systems from the École Nationale Supérieure des Télécommunications, Paris, France, in 2002; and the Ph.D. degree from the Université Pierre et Marie Curie (Paris 6), in 2005, after completing a thesis on automatic audio classification. His research interests are in machine learning for multimodal signal analysis with applications to music information retrieval, audio-visual content analysis, and human behavior and activity analysis. He has been involved in various French and European research projects among which are Quaero, Infom@gic, Networks of Excellence Kspace and 3DLife, and collaborative projects REVERIE and VERVE. Over the past 3 years, he has graduated 3 PhD students and is currently supervising 4 other students and collaborating with 2 post-docs. He has published over 60 peer-reviewed conference and journal papers with more than 50 distinct co-authors. He serves on a regular basis as a reviewer for various audio and multimedia conferences and journals, for instance various IEEE transactions, and as an expert for research funding agencies. In 2013 he is co-chairing the 14th edition of the International Workshop on Image and Audio Analysis for Multimedia Interactive Services (WIAMIS). More information on http://www.telecom-paristech.fr/ essid.

Panel Session

Industrial Panel

Chair

George Tzanetakis (University of Victoria, Canada)

Panelists

Mickael Le Goff (Native Instruments)
Andreas Ehmann (Pandora)
Matthew Hoffman (Adobe)
Philippe Hamel (Google)

The focus of this panel is to discuss the relationship between academic MIR research and industry practice. For this, we have invited key representative of some of the companies that have been actively utilizing MIR techniques in their business. The panelists will discuss how their companies have utilized MIR in the past, what MIR topics they are currently working on, and suggest directions for future research. The plan is to leave enough time for open ended discussion and questions.

Oral Session 1:

Representation and Learning

MULTISCALE APPROACHES TO MUSIC AUDIO FEATURE LEARNING

Sander Dieleman and Benjamin Schrauwen
Electronics and Information Systems department, Ghent University
{sander.dieleman, benjamin.schrauwen}@ugent.be

ABSTRACT

Content-based music information retrieval tasks are typically solved with a two-stage approach: features are extracted from music audio signals, and are then used as input to a regressor or classifier. These features can be engineered or learned from data. Although the former approach was dominant in the past, feature learning has started to receive more attention from the MIR community in recent years. Recent results in feature learning indicate that simple algorithms such as K-means can be very effective, sometimes surpassing more complicated approaches based on restricted Boltzmann machines, autoencoders or sparse coding. Furthermore, there has been increased interest in multiscale representations of music audio recently. Such representations are more versatile because music audio exhibits structure on multiple timescales, which are relevant for different MIR tasks to varying degrees. We develop and compare three approaches to multiscale audio feature learning using the spherical K-means algorithm. We evaluate them in an automatic tagging task and a similarity metric learning task on the Magnatagatune dataset.

1. INTRODUCTION

Content-based *music information retrieval* (MIR) techniques can be used to solve a variety of different problems, such as genre classification, artist recognition and music recommendation. They typically have a two-stage architecture: first, features are extracted from music audio signals to transform them into a more meaningful representation. These features are then used as input to a regressor or a classifier, which is trained to perform the task at hand.

Although machine learning techniques such as support vector machines and neural networks have traditionally been popular for the second stage, the features extracted from the audio are typically engineered rather than learned. Feature engineering is a complex and time-consuming process. Furthermore, these features are usually designed with a particular task in mind and are likely not optimally suited for other tasks. A prime example of this are the popular *mel-frequency cepstral coefficients* (MFCCs), which were originally designed for speech processing. MFCCs mainly encode timbre and discard pitch information, which is often undesirable when working with music audio. Despite this, they are still very commonly used for this purpose.

In recent years, feature learning has started to receive more attention from the MIR community. This can be attributed at least in part to a surge of interest in feature learning in general, ever since the emergence of *deep learning* in the mid-2000s [12]. Simply put, the idea of deep learning is to learn a hierarchy of features, organized in layers that correspond to different levels of abstraction. Higher-level features are defined in terms of lower-level features.

A similar evolution has taken place in computer vision and speech processing, where approaches based on deep learning are now commonplace, after improving on the previous state of the art by a large margin [6, 14]. In light of this, Humphrey et al. [13] advocate the use of deep architectures to solve MIR problems. However, recent results in feature learning indicate that simple, shallow (single-layer) feature learning techniques such as the K-means algorithm can also be quite competitive, sometimes surpassing more complicated approaches based on restricted Boltzmann machines, autoencoders and sparse coding [5]. These models are typically much more difficult to tune than the K-means algorithm, which only has one parameter (the number of means). They are often an order of magnitude slower to train as well.

Another recent development in MIR research is the increased interest in *multiscale architectures* [1,8,10]. Music audio exhibits structure on many different timescales: at the lowest level, signal periodicity gives rise to pitch. Periodicity at longer timescales emerges from rhythmic structure, repeated motifs and musical form. These timescales are relevant for different tasks to varying degrees.

Some researchers have explored architectures that are both deep and multiscale: for example, in convolutional neural networks, subsampling layers are often inserted between the convolutional layers, so that the features at each successive level take a larger part of the input into account [7, 16]. Indeed, it is not unreasonable to assume that more complex features will typically span longer timescales. However, these concepts need not necessarily be intertwined; for example in multiresolution deep belief networks [19], both hierarchies are separated.

In this paper, we endeavor to build a versatile feature learning architecture for music audio that operates on multiple timescales, using the spherical K-means algorithm. We compare three multiscale architectures and evaluate the resulting features in an automatic tagging task and a similarity metric learning task on the Magnatagatune dataset

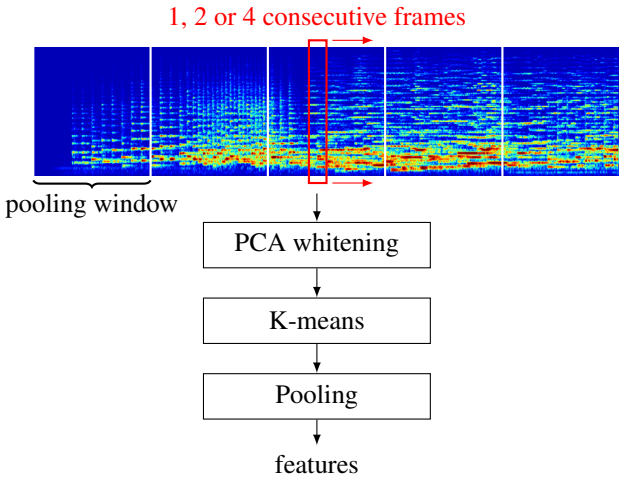


Figure 1: Schematic overview of the feature extraction process at a single timescale. The mel-spectrogram is divided into large pooling windows. Smaller windows consisting of 1, 2 or 4 consecutive frames are extracted convolutionally and PCA-whitened, and then K-means features are extracted. The features are pooled by taking the maximum across time over the pooling windows.

[15]. The rest of the paper is structured as follows: in Section 2 we outline the feature learning algorithm we used. A description of the multiscale architectures we investigated follows in Section 3. Our experiments and evaluation methods are detailed in Section 4 and the results are discussed in Section 5.

2. FEATURE LEARNING

To learn features from music audio, it is typically converted into a time-frequency representation first, such as a mel-spectrogram (see Section 4.1 for details). Feature learning algorithms can then be applied to individual spectrogram frames or windows of consecutive frames. We used the following feature extraction pipeline, which is visualized schematically in Figure 1: first, the mel-spectrograms are divided into large pooling windows, several seconds in length. Smaller windows consisting of 1, 2 or 4 consecutive frames are then extracted convolutionally and PCA-whitened, and K-means features are extracted from the whitened windows. The features are pooled by taking the maximum across time over the pooling windows. Each part of the pipeline is decribed in more detail below. For now, we will assume a typical single-timescale setting. In Section 3, we extend our approach to multiscale time-frequency representations.

2.1 PCA whitening

First, we randomly sample a set of windows from the data, and whiten them with PCA. We keep enough components to retain 99% of the variance. The whitened windows are then used to learn the dictionary. It has been shown that this whitening step considerably improves the features learned by the K-means algorithm [5].

2.2 Spherical K-means

We then use spherical K-means to learn features from the whitened windows. The spherical K-means algorithm differs from more traditional variants in the fact that the means are constrained to have a unit L2 norm (they must lie on the unit sphere). This is achieved by adding a normalization step in every iteration after the means are updated. For a detailed overview of the algorithm, we refer to Coates et al. [4].

K-means has often been used as a dictionary learning algorithm in the past, but it has only recently been shown to be competitive with more advanced techniques such as sparse coding. The one-of-K coding (i.e., each example being assigned to a single mean) is beneficial during learning, but it turns out to be less suitable for the encoding phase [3]. By replacing the encoding procedure, the features become significantly more useful. For spherical K-means, it turns out that using a linear encoding scheme works well: the feature representation of a data point is obtained by multiplying it with the dictionary matrix. To extract features from mel-spectrograms with a sliding window, we have to whiten the windows and extract features, which can be implemented as a single convolution with the product of the whitening matrix and the dictionary matrix.

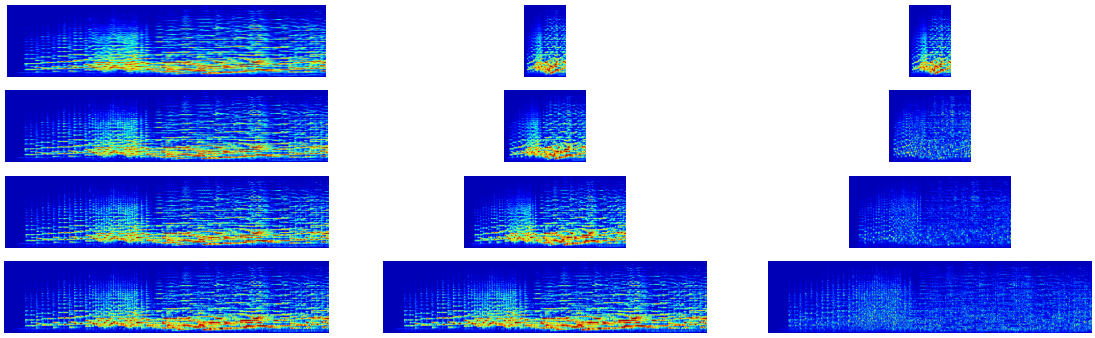
We can also skip the K-means step altogether and use the PCA components obtained after whitening as features directly. These features were referred to as principal mel-spectrum components (PMSCs) by Hamel et al. [11]. They are in fact very similar to MFCCs: if the windows consist of single frames, replacing the PCA whitening step with a discrete cosine transform (DCT) results in MFCC vectors. Both transformations serve to decorrelate the input.

2.3 Pooling

The representation we obtain by extracting features convolutionally from mel-spectrograms is not yet suitable as input to a regressor or classifier. It needs to be summarized over the time dimension first. We pool the features across large time windows several seconds in length. Although a combination of the mean, variance, minimum and maximum across time has been found to work well for this purpose [11], we use only the maximum, because it was found to be the best performing single pooling function. This reduces the size of the feature representation fourfold, which speeds up experiments significantly while having only a limited impact on performance. We used non-overlapping pooling windows for convenience, but our approach does not preclude the use of overlapping windows. Figure 1 shows a schematic overview of the feature extraction process.

3. MULTISCALE APPROACHES

We explore three approaches to obtain multiscale time-frequency representations from mel-spectrograms: multiresolution spectrograms, Gaussian pyramids and Laplacian pyramids [2]. An example of each is given in Figure



(a) Multiresolution spectrograms (b) Gaussian pyramid (c) Laplacian pyramid

Figure 2: Three different 4-level multiscale time-frequency representations of music audio. Level 0 is at the bottom, higher levels correspond to coarser timescales. This figure is best viewed in color.

2. To arrive at a multiscale feature representation, we simply apply the pipeline described in Section 2 to each level separately, and concatenate the resulting feature vectors.

3.1 Multiresolution spectrograms

The most straightforward approach to learning music audio features at multiple timescales is to extract mel-spectrograms with different window sizes. This approach was previously explored by Hamel et al. [10]. For each consecutive level, we simply double the spectrogram window size of the previous level. Although we could also double the hop size to reduce the size of the higher level representations, this was found to decrease performance considerably, so it is kept the same for all levels. Figure 2a shows an example of a set of multiresolution spectrograms.

3.2 Gaussian pyramid

The Gaussian pyramid is a very popular multiscale representation in image processing. Each consecutive level of a Gaussian pyramid is obtained by smoothing and then subsampling the previous level by a factor of 2 in the time dimension. To our knowledge, it has not previously been applied to time-frequency representations of music audio. In this case, mel-spectrograms are extracted only at the finest timescale and all higher levels can be obtained from them. Higher-level representations in a Gaussian pyramid will be smaller in size because of the subsampling step. This means that the pooling windows used in our feature extraction pipeline must be shrunk accordingly. An example of a Gaussian pyramid is shown in Figure 2b. Note that the lowest level of the Gaussian pyramid is identical to that of the multiresolution spectrograms.

3.3 Laplacian pyramid

The Laplacian pyramid can be derived from the Gaussian pyramid by taking each level and subtracting an upsampled version of the level above it. The top level remains the same. The result is that the representations at finer timescales will not contain any information that is already represented at coarser timescales. This reduces redundancy between the levels of the pyramid. An example of a Laplacian pyramid is shown in Figure 2c.

3.4 Modeling local temporal structure

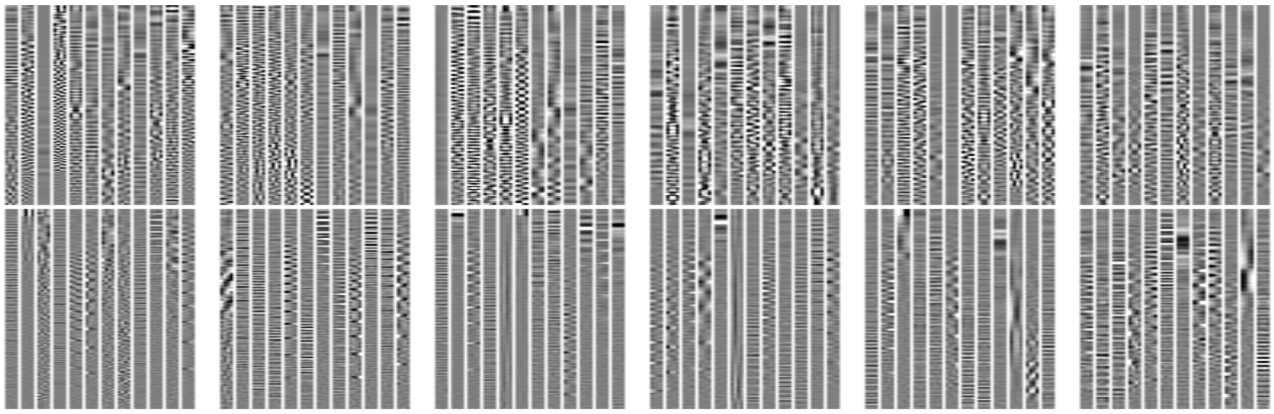
Frames taken from multiresolution spectrograms will automatically model longer-range temporal structure at higher levels, because the spectrogram window size is increased. For the Gaussian and Laplacian pyramids however, this is not the case: any temporal structure present at longer timescales is lost by the downsampling operation that is required to construct the higher levels of the pyramid. Although frames at higher levels of the pyramid are affected by a larger region of the input, they do not reflect temporal structure within this region. To allow for this structure to be taken into account by the feature learning algorithm, we can instead use windows of a small number of consecutive frames as input. This is not useful for multiresolution spectrograms because the temporal resolution is the same at all levels, i.e. higher levels do not have coarser temporal resolutions as is the case for the pyramid representations.

Figure 3 shows a random selection of features learned from windows of 4 consecutive mel-spectrogram frames at different levels in a 6-level Gaussian pyramid, with PCA whitening (top) and with spherical K-means (bottom). Some features are stationary across the time dimension, others resemble percussive events and changes in pitch. Many of the K-means features seem to reflect harmonic structure. This is especially the case for level 1 where the features span roughly half a second, which is around the average duration of a musical note. This type of structure is less pronounced in the PCA features.

4. EXPERIMENTS

4.1 Dataset

We used the Magnatagatune dataset [15], which contains 25863 29-second audio clips taken from songs by 230 artists sampled at 16 kHz, along with metadata and tags. It comes in 16 parts, of which we used the first 12 for training, the 13th for validation and the remaining 3 for testing. We extracted log-scaled mel-spectrograms with 200 components, with a window size of 1024 frames (corresponding to 64 ms) and a hop size of 512 frames (32 ms). For the multiresolution spectrogram representation, the spectrogram window size was doubled for each con-



(a) level 0 (b) level 1 (c) level 2 (d) level 3 (e) level 4 (f) level 5

Figure 3: A random selection of features learned with PCA whitening (top) and spherical K-means (bottom) from windows of 4 consecutive mel-spectrogram frames, at different levels in a Gaussian pyramid. The frequency increases from bottom to top.

secutive level. We used 6 levels for all three multiscale representations, numbered 0 to 5. This means that frames at the highest level span 2 seconds and are spaced 1 second apart. We used pooling windows of 128 spectrogram frames at the lowest level (about 4 seconds).

4.2 Tag prediction

The main task we used to evaluate the proposed feature learning architectures is a tag prediction task. This allows us to evaluate the versatility of the representations, because tags describe a variety of different aspects of the music: genre, presence (or absence) of an instrument, tempo and dynamics, among others. All clips in the dataset are annotated with tags from a set of 188. We only used the 50 most frequently occurring tags for our experiments, to ensure that enough training data is available for each of them.

We trained a multilayer perceptron (MLP) on the proposed multiscale feature representations to predict the presence of the tags, with a hidden layer consisting of 1000 rectified linear units [17]. We used minibatch gradient descent with weight decay to minimize the cross entropy between the predictions and the true tag assignments, and stopped training when the performance on the validation set was no longer increasing. Hyperparameters such as the learning rate and the weight decay constant were optimized by performing a random search on the validation set. We trained the MLP on feature vectors obtained from pooling windows. Tag predictions for an entire clip were computed by taking the average of the predictions across all pooling windows. We computed the area under the ROC-curve (AUC) for all tags individually and then took the average across all tags to get a measure of the predictive performance of the trained models.

We evaluated four different feature learning setups: PCA whitening, and spherical K-means with 200, 500 and 1000 means. We also compared 7 different multiscale approaches: Gaussian and Laplacian pyramids with windows of 1, 2 and 4 consecutive frames, and multiresolution spectrograms. This yields 28 different architectures in total.

4.3 Similarity metric learning

We also used the features to learn an audio-based music similarity metric, to further assess their versatility. Using Neighborhood Components Analysis (NCA) [9], we learned a linear projection of the features into a 50-dimensional space, such that similar clips are close together in terms of Euclidean distance. Each clip is mapped into this space by projecting the feature vectors corresponding to each pooling window and then taking the mean across all pooling windows. The linear projection matrix is then optimized with minibatch gradient descent to project clips by a given artist into the same region. This approach to learning a music similarity metric was previously explored by Slaney et al. [18].

NCA is based on a probabilistic version of K-nearest neighbor classification, where neighbors are selected probabilistically proportionally to their distance and each data point inherits the class of its selected neighbor. The objective is then to maximize the probability of correct classification. We report this probability on the test set.

5. RESULTS

5.1 Architectures

The results for the tag prediction task obtained with each of the 28 different architectures are shown in Figure 4. All reported results are averaged across 10 MLP training runs with different initializations. Unfortunately we cannot directly compare our results with those of Hamel et al. [10], because they used a different version of the dataset.

The first thing to note is that using features learned with spherical K-means almost always yields increased performance, although the difference between using 500 or 1000 means is usually small. Interestingly, the best performing architecture uses a Laplacian pyramid, with features learned from single frames. This is somewhat unexpected, because it implies that grouping consecutive frames into windows so that the feature learning algorithm can capture temporal structure is not necessary for this type of multi-

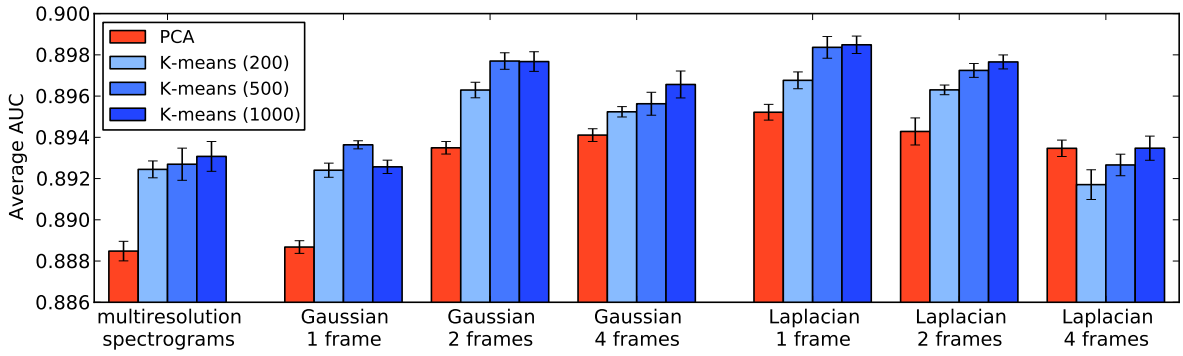


Figure 4: Results for the tag prediction task, for 28 different multiscale feature learning architectures. All reported results are averaged across 10 MLP training runs with different initializations. Error bars indicate the standard deviation across these 10 runs.

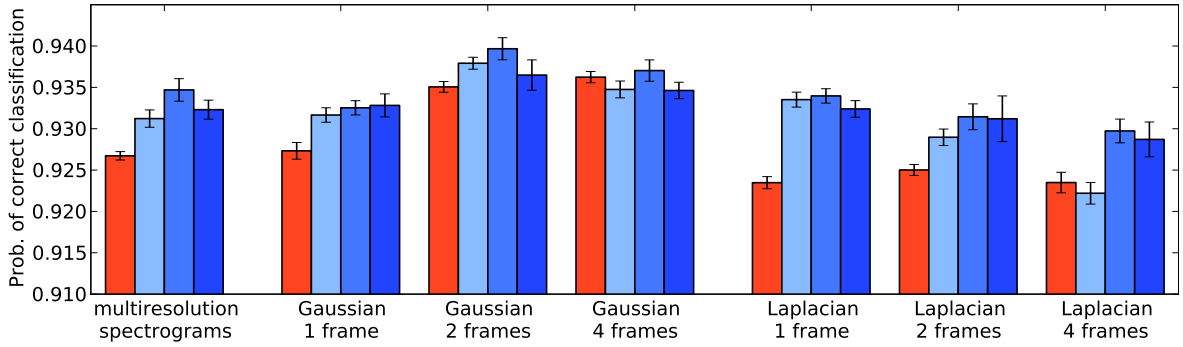


Figure 5: Results for the similarity metric learning task, for 28 different multiscale feature learning architectures. All reported results are averaged across 10 training runs with different initializations. Error bars indicate the standard deviation across these 10 runs.

scale representation. This does seem to help when using a Gaussian pyramid, however.

Results for the similarity metric task are shown in Figure 5. Here, a Gaussian pyramid with windows of 2 consecutive frames works best. Using 1000 means is noticeably worse than using 500 means. This can be attributed to the fact that the NCA objective seems to be very prone to overfitting when using a large amount of input features (6000 in this case, for 6 levels), despite our use of weight decay and early stopping for regularization.

5.2 Relevant timescales

To assess which timescales are relevant for different tags, we took the best architecture from the previous experiment and tried to predict tags from each level individually. Although a combination of all levels performs best for all tags, it is not always obvious precisely which timescales are the most relevant ones for a given tag. Figure 6 shows a selection of tags where some patterns can be identified.

Two tags describe the tempo of the music: *slow* and *fast*. As expected, the highest level seems to be the most important one for *slow*. For *fast*, level 1 (corresponding to a frame size of 128 ms) performs best. Both tags benefit quite a lot from the multiscale representation: a combination of all levels performs much better than any level individually. Tags describing dynamics, such as *loud*, *quiet* and *soft*, seem to rely mostly on the top level, corresponding to the coarsest timescale. This may also be because the top level is the only level in the Laplacian pyramid that is not a difference of two levels in the Gaussian pyramid.

Tags related to vocals can be predicted most accurately from intermediate levels, as evidenced by the results for *vocal*, *female*, *singing* and *vocals*. Of these, *female* is the easiest to predict, being the most specific tag. Finally, the *flute* tag is somewhat atypical among the tags describing instruments, in that it is the only one that relies mostly on the coarsest timescale (results for other instruments are not shown). A possible reason for this could be that the instrument lends itself well to playing longer, drawn out notes. A quick examination of the dataset reveals that many examples tagged *flute* in the dataset feature such notes.

6. CONCLUSION AND FUTURE WORK

We have proposed three approaches to building multiscale feature learning architectures for music audio, and we have evaluated them using two tasks designed to demonstrate the versatility of the learned features. Although learning features with the spherical K-means algorithm consistently improves results over just using PCA components, there is no clear winner among the proposed multiscale architectures. However, it is clear that learning features at multiple timescales improves performance over single-timescale approaches. We have also shown that different kinds of tags tend to rely on different timescales. Finally, we have

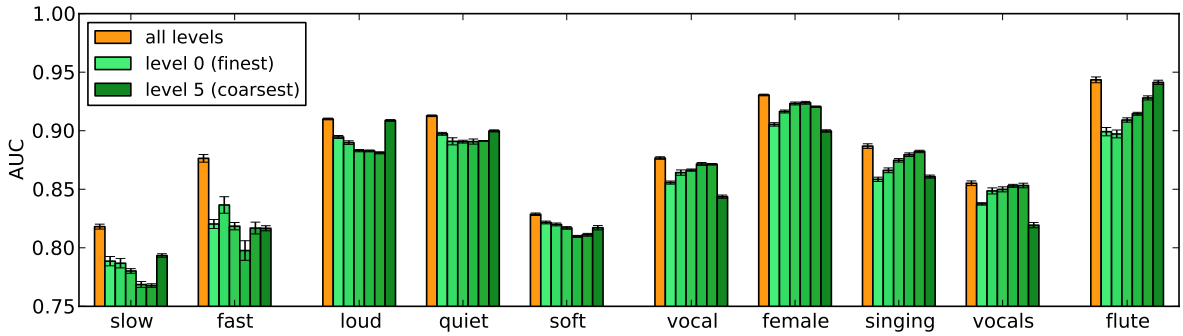


Figure 6: Results for a selection of individual tags, when using features from all levels combined and when using features from each level individually. The reported AUCs are for K-means with 500 means on the Laplacian pyramid with features learned from single frames.

observed that special care has to be taken to prevent overfitting when using large feature representations, which is almost unavoidable if a multiscale representation is desired.

In future work, we would like to improve the feature learning pipeline, by learning multiple layers of features for each timescale and investigating other encoding schemes. We would also like to evaluate the proposed architectures on an extended range of tasks, including content-based music recommendation and artist recognition, and on multiple datasets.

7. REFERENCES

[1] Joakim Andén and Stéphane Mallat. Multiscale scattering for audio classification. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[2] P.J. Burt and E.H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31:532–540, 1983.

[3] Adam Coates and Andrew Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011.

[4] Adam Coates and Andrew Y. Ng. Learning feature representations with k-means. *Neural Networks: Tricks of the Trade, Reloaded*, 2012.

[5] Adam Coates, Andrew Y. Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. *Journal of Machine Learning Research - Proceedings Track*, 15:215–223, 2011.

[6] G. E. Dahl, Dong Yu, Li Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42, January 2012.

[7] Sander Dieleman, Philémon Brakel, and Benjamin Schrauwen. Audio-based music classification with a pretrained convolutional network. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[8] Rémi Foucard, Slim Essid, Mathieu Lagrange, Gaël Richard, et al. Multi-scale temporal fusion by boosting for music classification. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[9] Jacob Goldberger, Sam Roweis, Geoff Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520, 2004.

[10] Philippe Hamel, Yoshua Bengio, and Douglas Eck. Building musically-relevant audio features through multiple timescale representations. In *Proceedings of the 13th International Conference on Music Information Retrieval (ISMIR)*, 2012.

[11] 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 Conference on Music Information Retrieval (ISMIR)*, 2011.

[12] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[13] Eric J. Humphrey, Juan P. Bello, and Yann LeCun. Moving beyond feature design: Deep architectures and automatic feature learning in music informatics. In *Proceedings of the 13th International Conference on Music Information Retrieval (ISMIR)*, 2012.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, 2012.

[15] Edith Law and Luis von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the 27th international conference on Human factors in computing systems*, 2009.

[16] Honglak Lee, Peter Pham, Yan Largman, and Andrew Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems 22*. 2009.

[17] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010.

[18] Malcolm Slaney, Kilian Q. Weinberger, and William White. Learning a metric for music similarity. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, 2008.

[19] Yichuan Tang and Abdel rahman Mohamed. Multiresolution deep belief networks. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

TRANSFER LEARNING IN MIR: SHARING LEARNED LATENT REPRESENTATIONS FOR MUSIC AUDIO CLASSIFICATION AND SIMILARITY

Philippe Hamel, Matthew E. P. Davies, Kazuyoshi Yoshii and Masataka Goto
National Institute of Advanced Industrial Science and Technology (AIST), Japan
hamelphi@google.com, {matthew.davies, k.yoshii, m.goto}@aist.go.jp

ABSTRACT

This paper discusses the concept of transfer learning and its potential applications to MIR tasks such as music audio classification and similarity.

In a traditional supervised machine learning setting, a system can only use labeled data from a single dataset to solve a given task. The labels associated with the dataset define the nature of the task to solve. A key advantage of transfer learning is in leveraging knowledge from related tasks to improve performance on a given target task. One way to transfer knowledge is to learn a shared latent representation across related tasks. This method has shown to be beneficial in many domains of machine learning, but has yet to be explored in MIR.

Many MIR datasets for audio classification present a semantic overlap in their labels. Furthermore, these datasets often contain relatively few songs. Thus, there is a strong case for exploring methods to share knowledge between these datasets towards a more general and robust understanding of high level musical concepts such as genre and similarity.

Our results show that shared representations can improve classification accuracy. We also show how transfer learning can improve performance for music similarity.

1. INTRODUCTION

As human beings, we are constantly learning to solve new tasks every day. The way we learn to perform new tasks is influenced by what we know about similar tasks [17].

For instance, let's think of a pianist that wants to learn to play guitar. The musician already has some knowledge of music theory, and knows how to use his motor skills to play the piano. When he learns to play guitar, he will not start from scratch but rather use his prior knowledge on music and motor skills and build on top of it. We can see it as if the musician transfers knowledge between tasks by sharing a common abstract internal representation of music.

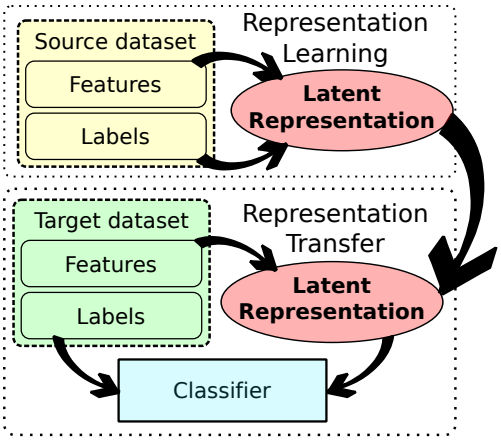


Figure 1: Schema of our transfer learning approach. In the first step, we learn a latent representation in a supervised way using a source dataset. In the second step, we solve the target task by first mapping the features to the learned latent space. In this example, the target task is a classification task.

The equivalent concept in machine learning is called *transfer learning*. It has been applied successfully in many domains such as visual object recognition [13] and webpage classification [8].

The performance of a supervised machine learning system is limited by the quantity and the quality of available labeled data. Obtaining such data can be expensive. As a consequence, many datasets in the MIR community have a relatively small number of labeled examples. Some of these datasets have been built to solve the same task, or similar tasks. For example, there exist many datasets for genre classification, and these datasets exhibit semantic overlap in their labels. However, each individual dataset contains a relatively small number of examples. In this context, it would make sense to try to leverage the information from all these datasets to improve the overall performance. Transfer learning might allow us to do just that.

In this paper, we investigate how transfer learning applied to genre classification, automatic tag annotation and music similarity can be beneficial. We hypothesize that transferring latent representations learned on related tasks can improve the performance of a given task when compared with the original features. Our intuition is that the learned representation will retain some knowledge of the

original task and that this knowledge should make the given task easier to solve.

The paper is divided as follows. We begin with an overview of transfer learning in Section 2. We describe the different MIR tasks that are relevant to our experiments in Section 3. In Section 4 we give details about how we handle our features. The representation learning algorithm is presented in Section 5. We describe our experimental results in Section 6. Finally, we conclude in Section 7.

2. TRANSFER LEARNING

Transfer learning is a machine learning problem that focuses on reusing knowledge learned on one problem in order to help solve another. More formally, we will distinguish between the *target task*, which is the task that we ultimately want to solve, and the *source task* which is the related task that will help us in solving the target task. It is worth noting that there could be more than one source or target task.

Transfer learning is an active field of research, and many approaches have been proposed [2, 8, 13]. Pan et al. [6] describe four transfer learning approaches: i) *instance transfer*, ii) *feature representation transfer*, iii) *parameter transfer* and iv) *relational knowledge transfer*. In this work, we will focus on the feature representation transfer approach, which consists of learning a common feature space between the source and target tasks. More specifically, we will use a supervised approach to construct a feature space using labeled data from a source task, and then use this feature space to help solve the target task. This transfer learning process is illustrated in Figure 1.

Although transfer learning has been applied successfully in many domains, only a few applications can be found in the MIR domain. In [8], self-taught learning, which is an extension of semi-supervised learning, is applied to many tasks, including a 7-way music genre classification. However, very few details are provided on the nature of the music data. In [3], a deep representation learned on genre classification is used for automatic tag annotation. Although the transferred representation is compared to a set of audio features, there is no comparison to the original spectral features that were used to build the deep representation. Thus, it is difficult to assess the impact of the transfer of representation. In [10], a learned automatic tag annotation system is used to produce features to help solve a music similarity task. In [15], a method that attempts to capture the semantic similarities between audio features, tags, and artists names is presented. This multitask approach consists of embedding the different concepts in a common low-dimensional space. This learned space can then be used to solve many MIR related tasks. In our work, we use a similar approach to build a shared latent representation.

3. TASKS AND DATASETS

In this paper, we investigate transfer learning over three related MIR tasks: genre classification, music similarity estimation and automatic tag annotation. Even though these task all use music audio as input data, they differ in their goal and in the way performance is evaluated.

Table 1: Characteristics of the genre classification and automatic tag annotation datasets.

Dataset	# of excerpts	# of classes	Audio length
1517-Artists [11]	3180	19	full
GTZAN [14]	1000	10	30s
Homburg [4]	1886	9	10s
Unique [12]	3115	14	30s
Magnatagatune [5]	22787	160	30s

Table 2: Genre classes for the datasets. In bold are the terms which are also tags in the Magnatagatune dataset [5].

1517-Artists	GTZAN	Homburg	Unique
Alternative & **Punk**	**blues**	alternative	**blues**
Blues	**classical**	**blues**	**country**
Childrens's	**country**	**electronic**	**dance**
Classical	disco	**folkcountry**	**electronica**
Comedy & Spoken Word	**hiphop**	**funk**soulrnb	**hip-hop**
Country	**jazz**	**jazz**	**jazz**
Easy Listening & **Vocals**	**metal**	**pop**	**klassik**
Electronic & **Dance**	**pop**	**raphiphop**	reggae
Folk	reggae	**rock**	**rock**
Hip-Hop	**rock**		schlager
Jazz			soul_rnb
Latin			volksmusik
New Age			**world**
R&B & Soul			wort
Reggae			
Religious			
Rock & **Pop**			
Soundtracks & More			
World			

3.1 Genre Classification

Genre classification consists of choosing the genre that best describes an audio excerpt given a set of genre labels. We consider 4 different datasets for genre classification: 1517-Artists [11], GTZAN [14] Homburg [4], and Unique [12]. These datasets each contain between 1000 and 3180 audio excerpts, from 10 seconds in length to full songs, classified in 9 to 19 genres. In the case where full songs are provided, we use only the first 30 seconds of each song. Further details about the datasets are in Table 1. The genre labels have strong semantic overlap across datasets as can be seen in Table 2. For simplicity, we will sometimes refer to the 1517-Artists dataset as *artists*.

To evaluate the performance of a genre classification system, we use the classification accuracy, which is simply the percentage of correctly classified excerpts in the test set.

3.2 Music Similarity

Music similarity systems seek to obtain a measure of similarity between audio excerpts.

One issue with this task is that the meaning of *similarity* is ill-defined. What is considered similar by one listener might not be the same for another. Another issue is that

similarity is a pair-wise relative measure. Thus, it is complicated and costly to obtain enough ground truth information from human listeners to fully evaluate music similarity systems. In order to circumvent these issues, music similarity systems often use genre labels as a proxy for similarity evaluation [7, 9, 12]. In this context, we consider that two excerpts within the same genre must be more similar than two excerpts from different genres. On this basis, we will use the same datasets as for genre classification in our music similarity experiments.

Even though genre classification and music similarity use the same data, the tasks differ on how we use the data and on how we evaluate performance. Typically, in the music similarity literature [7, 9, 12], the labels are not used for training. Thus, the task must be solved by signal processing, unsupervised learning, or, in our case, by transferring supervised learning from external datasets.

The evaluation of music similarity systems typically use *precision at* k as a performance measure. *Precision at* k gives the ratio of excerpts of the same class in the k nearest neighbors of a given excerpt. In this work we use $k = 10$.

Approaches to solve this task typically consist of measuring distances in a feature space to obtain a distance matrix. The type of features and the distance measure used can vary. In [7], distance is computed using the Jensen-Shannon divergence on a Gaussian representation of the features. In [12], an L_1-distance is computed over aggregated block-level features. In [9], an L_1-distance is computed on features extracted in an unsupervised fashion.

In this work, we use the L_1-distance on our different feature sets in order to obtain a similarity matrix. We also tested Euclidian distance and Cosine distance and obtained similar results.

3.3 Tag annotation

The automatic tag annotation task consists of assigning words to describe an audio excerpt. It is a multi-label problem, meaning that many labels can be applied to a single example. In this paper, we use the Magnatagatune dataset [5] which contains more than 22,000 30-seconds excerpts and 160 tags. Tag labels include musical genre (rock, blues, jazz), instrumentation (guitar, piano, vocals), mood (sad, mellow), other descriptors (fast, airy, beat), etc. There is high semantic overlap with the genre labels from the four genre datasets. We illustrate this, in Table 2, by putting in bold the genres which are also tags in the Magnatagatune dataset.

4. AUDIO FEATURES

In our experiments, we extract Mel-spectrum features from audio. We compute the Discrete Fourier Transform (DFT) on frames of 46ms (1024 samples at 22kHz sampling rate) with half frame overlap. We then pass the magnitude spectrum through 200 triangle Mel-scaled filters and take the log-amplitude to obtain the Mel-spectrum features. These are what we will refer to as *frame-level* features.

However, frame level features have been shown to be suboptimal for genre classification [1]. To obtain a better classification performance, we aggregate features on windows of 64 frames (about 1.5s), computing the mean, variance, maximum and minimum of each feature. We can apply this aggregation process to the Mel-spectrum features as well as to the frame-level latent representations. We will refer to aggregated features as *window-level* features.

5. LEARNING A LATENT REPRESENTATION

In order to transfer knowledge between tasks, we aim to learn a latent representation that will be shared across tasks. To learn this representation, we use the linear embedding method described in [16]. This method consists of embedding both the features and the labels via linear transformations in a common space. This algorithm is built to handle a large number of labels in a multi-label problem, such as in the case of automatic tag annotation. However, the model can trivially be adapted to multi-class problems with a small number of classes such as genre recognition. The model has also been extended to multi-task learning in MIR in [15].

The algorithm seeks to map both the features and the labels in a common latent space, as illustrated in Figure 2. Given a feature representation $x \in \mathbb{R}^d$ and a set of labels $i \in \mathcal{Y} = \{1, ..., Y\}$, we seek to jointly learn a *feature embedding transform* that will map the feature space to a semantic space $\mathbb{R}^D$

$$\Phi_x(x) : \mathbb{R}^d \rightarrow \mathbb{R}^D$$

and a *label embedding transform* that will map labels to the same semantic space

$$\Phi_y(i) : \{1, ..., Y\} \rightarrow \mathbb{R}^D.$$

Thus, in this latent space, it is possible to measure distances between different concepts such as between two feature vectors, a feature vector and a label, or between two labels.

Since we use linear maps, we have $\Phi_x(x) = Vx$ where V is a $D \times d$ matrix and $\Phi_y(i) = W_i$ where W_i is the i-th column of a $D \times Y$ matrix. We can obtain an affinity measure between a feature vector and a given label with

$$f_i(x) = \Phi_y(i)^\top \Phi_x(x) = W_i^\top Vx.$$

Each training example has positive and negative labels associated to it. Given a feature vector, an optimal representation would yield high affinities for positive labels and low affinities for negative labels. In other words, if we rank the affinities of the labels to the feature vector, the positive labels should be ranked low (i.e. in the first few positions), and the negative labels should be ranked high. Computing the exact ranking of the labels becomes expensive when there are many labels. Thus, following [16] we use a stochastic method that allows us to compute an approximate ranking.

The training procedure is as follows. For a given training example x', we randomly pick a positive label j. Then,

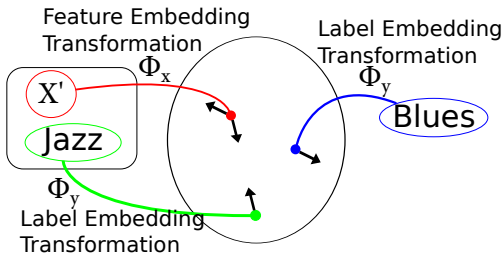


Figure 2: Illustration of the learning of the latent representation. Audio features and labels are mapped to a common embedding space via the transformations Φ_x and Φ_y. In this example, excerpt X' has *jazz* as a positive label and *blues* as a negative example. The black arrows illustrate how the learning gradient will push the negative label embedding and the feature embedding away from each other, while pulling the positive example embedding and the feature embedding together.

we iterate randomly through the negative labels until we find a label j' for which $f_{j'}(x') > f_j(x') - 1$. If we do not find such a negative label, we move to the next training example. If we only need a few iterations to find such a negative label, chances are that the rank of the positive label is high, we thus try to correct this by boosting the loss. On the contrary, if we need many iterations to find such a negative label, the rank of the positive label is probably quite low, so we do not need to change the representation as much. We then minimize the loss given by

$$\mathcal{L} = L(r)|1 - f_j(x') + f_{j'}(x')|$$

where $L(r) = \sum_{k=1}^{r} 1/k$ and r is the approximated rank of the label j and is given by

$$r = \left\lfloor \frac{Y-1}{N} \right\rfloor$$

where N is the number of iterations needed to find j', and $\lfloor \cdot \rfloor$ is the floor function. The loss $\mathcal{L}$ is known as the Weighted Approximate-Rank Pairwise loss, or WARP loss [16]. The L term increases as the approximate rank r grows. The second term in the loss can be seen as a kind of hinge loss, which tries to maximize the margin. For a more in depth description of the algorithm, see [16] and [15] .

In our experiments we used a batch method, meaning that we average the gradient over a batch before updating the parameters. We use 100 examples per batch. For the dimensionality of our latent space, we followed [16] and [15] and chose $D = 100$ as the latent dimensionality for all our experiments.

To extend the model to a multi-dataset setting, we simply alternate between datasets after each batch. The feature embedding transformation is shared across all datasets, but the label embedding transformations are independent. In this way, we do not assume any semantic similarity between similar classes across datasets. In Section 6.1, we show that the model naturally learns these semantic similarities.

6. EXPERIMENTS

We conduct several experiments to assess if transferring knowledge across datasets and task can be beneficial. First, we qualitatively evaluate the semantic similarity in a multi-dataset genre embedding. Then, we compare genre classification performance between tag embedding, genre embedding and the base features. Finally, we use these feature spaces for the music similarity task.

6.1 Semantic similarity

In our first experiment, we learn an embedding jointly on the four genre datasets. The combination of the four label sets gives us a total 52 labels. We then look at the nearest neighbours of the class embedding and make a qualitative evaluation. If the embedding process learns semantic information about the classes as expected, similar classes across datasets should be close to each other.

To do this, we compute a distance matrix using an L_1-distance on the embeddings of all the classes. Then, for each class, we look at which classes are the closest and perform a qualitative evaluation. Some typical examples are presented in Table 3. In general the similar classes across datasets tend to be close to one another. For example, in Table 3, we see that the *jazz* classes all end up near one another in the embedding space. However, there are also some problematic classes. For instance, the *blues* classes do not appear to all be clustered together. From these results, we can say that the embedding space indeed learns some kind of semantic knowledge about the classes.

6.2 Genre Classification

For this experiment, we consider three sets of features for each genre dataset: base features, genre embedding and tag embedding. The base features are the window-level aggregated Mel-spectrum features described in Section 4.

For a given genre dataset, the genre embedding is learned jointly on the 3 other genre datasets. It is learned on frame-level Mel-spectrum features. The frame-level embedded features are then aggregated in a similar fashion as the base features to obtain window-level features.

The tag embedding is learned on the Magnatagatune dataset. Again, the embedding is learned on frame-level features and these are then aggregated to obtain window-level features. We then train a simple linear regression classifier on the window-level features. Finally to classify a song, we average the output of the classifier on the whole song and pick the class with the highest output.

One of the key strengths of transfer learning compared to standard learning is the ability to improve performance using fewer training examples [8]. To test this hypothesis, we measure the accuracy of the classifier across a range of training examples per class in the target dataset. Since the number of examples per class is unbalanced in some datasets, there are cases where there are fewer examples for the less frequent classes. We ran a 10-fold cross-validation for each experiment. The results are shown in Figure 3.

Table 3: Nearest neighbouring classes in the genre embedding space for a few examples.

Seed	5 Nearest neighbours (in order)
Hip-Hop(artists)	hip-hop(unique), raphiphop(homburg), schlager(unique), hiphop(gtzan), Electronic & Dance(artists)
Rock & Pop(artists)	rock(unique) rock(homburg) Alternative & Punk(artists) metal(gtzan) alternative(homburg)
Electronic & Dance(artists)	raphiphop (homburg) reggae(unique) electronica(unique) pop(gtzan) dance(unique)
country(gtzan)	country(unique), folkcountry(homburg), rock(unique), Country(artists), Religious(artists)
jazz(homburg)	jazz(unique), jazz(gtzan), Jazz(artists), world(unique), dance(unique)
blues(unique)	alternative(homburg), Alternative & Punk(artists), blues(gtzan), funksoulrnb(homburg), rock(homburg)

Table 4: Classification accuracy and standard error on the full training set using a 10-fold cross-validation.

Dataset	Base Features	Genre Embedding	Tag Embedding
Artists	0.323 +/- 0.010	0.310 +/- 0.005	0.338 +/- 0.007
GTZAN	0.748 +/- 0.010	0.671 +/- 0.014	0.754 +/- 0.015
Homburg	0.580 +/- 0.012	0.561 +/- 0.009	0.584 +/- 0.008
Unique	0.651 +/- 0.006	0.634 +/- 0.005	0.666 +/- 0.006

Table 5: Precision at 10 for the music similarity task on different feature spaces. The genre embedding is learned using the 3 other genre datasets. The tag embedding is learned on the Magnatagatune dataset.

Dataset	Base Features	Genre Embedding	Tag Embedding
Artists	0.15	0.19	0.19
GTZAN	0.48	0.52	0.53
Homburg	0.36	0.41	0.40
Unique	0.53	0.52	0.54

These results show that the tag embedding often significantly outperforms the base features. This confirms our hypothesis. However, the genre embedding does not perform as well, obtaining better accuracy only for the *Homburg* dataset.

We then measured the accuracy of the three feature sets on the full training dataset. The results are in Table 4. We see that the tag embedding tends to give slightly better results.

6.3 Music similarity

For this task, we used the same 3 feature sets as in Section 6.2. We use precision at 10 as the performance measure. Results are shown in Table 5. We see that both the genre and tag embedding features perform better than the base features, except for the Unique dataset where the three feature sets perform about as well.

7. CONCLUSION

In this paper, we conducted experiments on sharing a learned latent representation between related MIR tasks. We showed that jointly learning a representation on many genre datasets naturally learns semantic similarity between genre classes. In the context of genre classification, we saw that transferring a representation between tasks can significantly improve classification accuracy when the number of training examples is limited. In the context of music similarity, we saw that the similarity space obtained by embedding features using genre and tag labels allows better precision.

The fact that the genre embedding performed worse than the base features for the genre classification task goes against our hypothesis that classification accuracy should be improved by such a representation. This might be due to the fact that the genre datasets are rather small, and thus there was not enough data to learn a robust representation. Another reason might be that some of the semantic knowledge that was learned ended up in the label embedding transform rather than the feature embedding transform. Since we did not use the label embedding transform in the classification task experiment, some of the learned knowledge might have been lost in the transfer. To address this problem in future work, we could try to impose a more severe regularization on the label embedding transformation in the learning process. This could help to force the semantic knowledge to go in the feature embedding transformation.

In this work, to focus on the simplest case first, we limited ourselves to basic feature aggregation, a linear embedding method, and a linear classifier. Each of these elements could be improved further. Thus the performance measures presented in this paper might not reflect the full power of transfer learning. For the features, more complex block-level features as described in [12] could be constructed from the learned frame-level representation. For the representation learning, non-linear mappings could be used to obtain a more powerful representation. Finally, more complex classifiers, such as support vector machines or neural networks could be used to improve classification accuracy on the learned features.

This work presents a first analysis of the potential of transfer learning in MIR. We hope that the results presented here will stimulate more research in the field and motivate the application of transfer learning in future MIR applications.

8. ACKNOWLEDGMENTS

This work was supported by OngaCREST, CREST, JST.

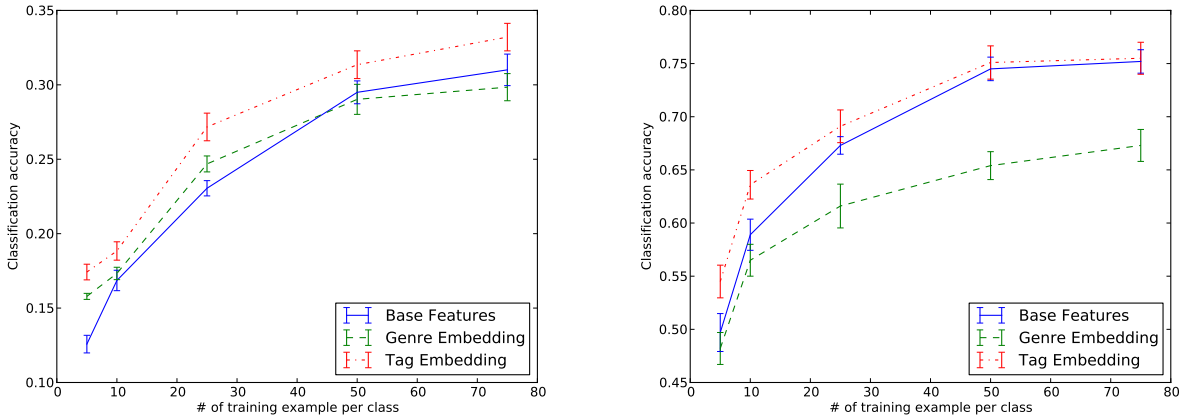
9. REFERENCES

[1] J. Bergstra. Algorithms for Classifying Recorded Music by Genre. Masters thesis, Université de Montréal, 2006.

[2] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997.

[3] P. Hamel and D. Eck. Learning features from music audio with deep belief networks. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 339–344, 2010.

[4] H. Homburg, I. Mierswa, B. Mller, K. Morik, and M. Wurst. A benchmark dataset for audio classification and clustering. In *Proceedings of the 6th International Society for Music Information Retrieval Conference (ISMIR)*, pages 528–531, 2005.



(a) Artists

(b) GTZAN

(c) Homburg

(d) Unique

Figure 3: Comparison of base features (baseline) to genre embedding and tag embedding for the genre classification task. The genre embedding and tag embedding representations are obtained through our proposed transfer learning method. The error bars correspond to the standard error across the 10 folds.

[5] E. Law and L. von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the International Conference on Human factors in computing systems*, pages 1197–1206, 2009.

[6] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.

[7] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 525–530, Kobe, Japan, 2009.

[8] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the Twenty-Fourth International Machine Learning Conference (ICML 2007)*, pages 759–766, Corvallis, Oregon, USA, 2007.

[9] J. Schlüter and C. Osendorfer. Music Similarity Estimation with the Mean-Covariance Restricted Boltzmann Machine. In *Proceedings of the 10th International Conference on Machine Learning and Applications (ICMLA 2011)*, pages 118–123, Honolulu, USA, 2011.

[10] K. Seyerlehner, R. Sonnleitner, Schedl, D. M., Hauger, and B. Ionescu. From improved auto-taggers to improved music similarity measures. In *Proceedings of the 10th International Workshop on Adaptive Multimedia Retrieval (AMR 2012)*, Copenhagen, Denmark, 2012.

[11] K. Seyerlehner, G. Widmer, and P. Knees. Frame-level audio similarity - a codebook approach. In *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx-2008)*, pages 349–356, Espoo, Finland, 2008.

[12] K. Seyerlehner, G. Widmer, and T. Pohle. Fusing block-level features for music similarity estimation. In *Proc. of the 13th Int. Conference on Digital Audio Effects (DAFx-2010)*, pages 528–531, Graz, Austria, 2010.

[13] T. Tommasi, N. Quadrianto, B. Caputo, and C. H. Lampert. Beyond dataset bias: Multi-task unaligned shared knowledge transfer. In *Proc. of the 11th Asian Conference on Computer Vision (ACCV)*, pages 1–15, Daejeon, Korea, 2012.

[14] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[15] J. Weston, S. Bengio, and P. Hamel. Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval. *Journal of New Music Research*, 40(4):337–348, 2011.

[16] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI*, volume 3, pages 2764–2770, 2011.

[17] R. S. Woodworth and E. L. Thorndike. The influence of improvement in one mental function upon the efficiency of other functions. *Psychological Review*, 8(3):247–261, May 1901.

A DISTRIBUTED MODEL FOR MULTIPLE-VIEWPOINT MELODIC PREDICTION

Srikanth Cherla[1,2], Tillman Weyde[1,2], Artur d'Avila Garcez[2] and Marcus Pearce[3]
[1]Music Informatics Research Group, Department of Computer Science, City University London
[2] Machine Learning Group, Department of Computer Science, City University London
[3]Centre for Digital Music, Queen Mary University of London
{srikanth.cherla.1, t.e.weyde, a.garcez}@city.ac.uk
marcus.pearce@eecs.qmul.ac.uk

ABSTRACT

The analysis of sequences is important for extracting information from music owing to its fundamentally temporal nature. In this paper, we present a distributed model based on the Restricted Boltzmann Machine (RBM) for melodic sequences. The model is similar to a previous successful neural network model for natural language [2]. It is first trained to predict the next pitch in a given pitch sequence, and then extended to also make use of information in sequences of note-durations in monophonic melodies on the same task. In doing so, we also propose an efficient way of representing this additional information that takes advantage of the RBM's structure. In our evaluation, this RBM-based prediction model performs slightly better than previously evaluated n-gram models in most cases. Results on a corpus of chorale and folk melodies showed that it is able to make use of information present in longer contexts more effectively than n-gram models, while scaling linearly in the number of free parameters required.

1. INTRODUCTION

Sequential structure in music influences our notions of musical style, similarity and the emotions we associate with it. The analysis of sequences in musical scores and equivalent symbolic representations of music is an integral part of Music Information Retrieval, with applications such as music classification [6], computational musicology [26], music creation [19], and music source separation [10]. In the past, this analysis has often been carried out using music generation systems [1, 4, 8, 13, 18].

The present research is based around previous work that adopted ideas proposed in information theory to music [7]. There, *Multiple-viewpoint Systems for Music Prediction* were introduced as a detailed re-interpretation of the key ideas of information theory [22] in music, through an analogy between language and musical style. In that work and what followed [21], Markov models were employed for learning melodic subsequences. While this is a reasonable choice, Markov models are often faced with a problem related to data sparsity known as the *curse of dimensionality*. This refers to the exponential rise in the number of model parameters with the length of the modelled subsequences. Recent research in language modelling has demonstrated that neural networks can be a suitable alternative to more widely used n-gram and variable-order Markov models [2, 5, 17]. There have been some initial results on the success of such models in music [3, 24].

In this paper, we present a model for melody prediction based on one such neural network — the Restricted Boltzmann Machine (RBM) [23]. The choice is motivated by the following. Firstly, the inherent non-linearity of the RBM makes it a suitable candidate for learning complex structures in data, such as those occurring in musical sequences. There exist efficient algorithms for training this model [11, 25]. The RBM, with its straightforward extensibility to deep networks [12], has become a vital building block for creating models that are capable of learning features from the data at multiple levels of abstraction.

We describe here a model for fixed-length subsequences of musical pitch, which compares favourably to n-gram models that were previously evaluated with a prediction task on a corpus of monophonic MIDI melodies [21]. This pitch-only version of the model is then adapted to also make use of note-durations in the melodies, on the same pitch-prediction task. In doing so, we also propose an efficient way to represent this additional information, which takes advantage of the RBM's structure and thus limits model complexity. The structure of the proposed model ensures that it scales only linearly with the length of subsequences to be learned and with the number of symbols in the data. We demonstrate an improvement of results by combining the two models in a manner similar to [7] using the arithmetic mean of their individual probability estimates. An implementation of the model in Python, along with scripts used to generate the results in this paper, are available upon request.

The remainder of this paper is organized as follows. The next section introduces music prediction and multiple viewpoint systems as a framework for music prediction. Section 3 explains the RBM and its discriminative inter-

pretation which make up the basis for the model proposed in this paper. This is followed by a description of the model itself in Section 4. An evaluation of the the model and its comparison with previously evaluated n-gram models is presented in Section 5, followed by discussion on possible directions for future research in Section 6.

2. MUSIC PREDICTION WITH MULTIPLE-VIEWPOINT SYSTEMS

In order to explain music prediction with multiple viewpoints, the analogy to natural language is used here. In statistical language modelling, the goal is to build a model that can estimate the joint probability distribution of subsequences of words occurring in a language L. A statistical language model (SLM) can be represented by the conditional probability of the next word w_T given all the previous ones $[w_1, \ldots, w_{(T-1)}]$ (written here as $w_1^{(T-1)}$), as

$$P(w_1^T) = \prod_{t=1}^{T} P(w_t | w_1^{(t-1)}) \; . \tag{1}$$

The most commonly used SLMs are *n-gram* models, which rely on the simplifying assumption that the probability of a word in a sequence depends only on the immediately preceding $(n-1)$ words [16]. This is known as the Markov assumption, and reduces (1) to

$$P(w_1^T) = \prod_{t=1}^{T} P(w_t | w_{(t-n+1)}^{(t-1)}) \; . \tag{2}$$

Following this approach, musical styles can be interpreted as vast and complex languages [7]. In music prediction, one is interested in learning the joint distribution of *musical event* sequences s_1^T in a *musical language* S. Much in the same way as an SLM, a system for music prediction models the conditional distribution $p(s_t | s_1^{(t-1)})$, or under the Markov assumption $p(s_t | s_{(t-n+1)}^{(t-1)})$. For each prediction, context information is obtained from the events $s_{(t-n+1)}^{(t-1)}$ immediately preceding s_t. Musical events have a rich internal structure and can be expressed in terms of directly observable or derived musical features such as pitch, note duration, inter-onset interval, or a combination of two or more such features. The framework of multiple-viewpoint systems for music prediction [7] was proposed in order to efficiently handle this rich internal structure of music by exploiting information contained in these different musical feature sequences, while at the same time limiting the dimensionality of the models using these features. In the interest of brevity, we limit ourselves to an informal discussion of multiple-viewpoint systems for monophonic music prediction and refer the reader to [7] for the underlying mathematical formulation.

A musical event s refers to the occurrence of a note in a melody. A *viewpoint type* (henceforth written as *type*) τ refers to any of a set of musical features that describe an event. The domain of a *type*, denoted by $|\tau|$ is the set of possible values of that type. A *basic type* is a directly observable or given feature such as *pitch, note duration, key-signature* or *time-signature*. A *derived type* can be derived from any of the basic types or other derived types. A *linked viewpoint type* is created by taking the Cartesian product over two or more types, thus "linking" them.

A *multiple-viewpoint system* (MVS) is a set of models, each of which is trained on subsequences of one *type*, whose individual predictions are combined in some way to influence the prediction of the next event in a given event sequence. Given a context $s_{(t-n+1)}^{(t-1)}$ and an event s_t, each viewpoint τ in an MVS must compute the probability $p_\tau(s_t | s_{(t-n+1)}^{(t-1)})$. While originally n-gram models were proposed to be used with the multiple viewpoints framework, we demonstrate how a distributed model such as the RBM used here can serve as a scalable alternative.

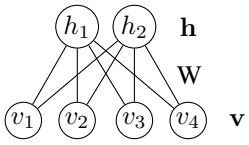


Figure 1. A simple Restricted Boltzmann Machine with four visible, two hidden, and no bias units.

3. RESTRICTED BOLTZMANN MACHINE

The Restricted Boltzmann Machine (RBM) is an undirected graphical model consisting of a set of r visible units $\mathbf{v}$ and a set of q hidden units $\mathbf{h}$. These make up the visible and hidden layers of the RBM respectively. The two layers are fully inter-connected but there exist no connections between any two hidden units, or any two visible units. In its original form, the RBM has binary, logistic units in both layers. Additionally, the units of each layer are connected to a bias unit whose value is always 1.

The edge between the i^{th} visible node and the j^{th} hidden node is associated with a weight w_{ji}. All these weights are together represented in a *weight matrix* $\mathbf{W}$ of size $q \times r$. The weights of connections between visible units and the bias unit are contained in an r-dimensional *visible bias* vector $\mathbf{b}$. Likewise, for the hidden units there is a q-dimensional *hidden bias* vector $\mathbf{c}$. The RBM is fully characterized by the parameters $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{c}$. Figure 1 shows a simple RBM with four visible and two hidden units, without the bias unit to better illustrate its bipartite structure.

The activation probabilities of the units in the hidden layer given the visible layer (and vice versa) are given by the logistic sigmoid function as $p(h_j = 1|\mathbf{v}) = \sigma(c_j + W_{j\cdot}\mathbf{v})$, and $p(v_i = 1|\mathbf{h}) = \sigma(b_i + W'_{i\cdot}\mathbf{h})$ respectively. Due to the RBM's bipartite structure, the activation probabilities of the nodes within one of the layers are independent, if the activation of the other layer is given, i.e.

$$p(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{q} p(h_j|\mathbf{v}) \tag{3}$$

$$p(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{r} p(v_i|\mathbf{h}) \; . \tag{4}$$

The RBM is a special case of the Boltzmann Machine, which is an energy-based model for representing probability distributions [15]. In such energy-based models, probability is expressed in terms of an energy function. In the case of the RBM, this function is expressed as

$$Energy(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^\top \mathbf{v} - \mathbf{c}^\top \mathbf{h} - \mathbf{h}^\top \mathbf{W} \mathbf{v} \,. \quad (5)$$

Learning in energy-based models can be carried out in a *generative* fashion, by updating the weights and biases in order to minimize the overall energy of the system with respect to the training data. This amounts to maximizing the log-likelihood function of the joint probability distribution $p(\mathbf{v})$, which is given by

$$p(\mathbf{v}) = \frac{e^{-FreeEnergy(\mathbf{v})}}{Z}, \quad (6)$$

with $Z = \sum_{\mathbf{v}} e^{-FreeEnergy(\mathbf{v})}$, where

$$FreeEnergy(\mathbf{v}) = -\log \sum_{\mathbf{h}} e^{-Energy(\mathbf{v},\mathbf{h})} \,. \quad (7)$$

While computing the exact gradient of the log-likelihood function for $p(\mathbf{v})$ is not tractable, an approximation of this gradient called the Contrastive Divergence (CD) gradient has been found to be a successful update rule for training RBMs [11]. With the CD update, the RBM can be trained efficiently.

The RBM described above models the joint probability $p(\mathbf{v})$ of the set of visible units $\mathbf{v}$. However, as described in Section 2, we are interested in a conditional distribution of the form $p(y|\mathbf{x})$. It has been demonstrated in [14] how an RBM can be used for a *discriminative* task such as classification. The posterior class probability distribution of such an RBM has the form

$$p(\mathbf{y} = \mathbf{e}_c|\mathbf{x}) = \sum_{\mathbf{h}} p(\mathbf{y} = \mathbf{e}_c, \mathbf{h}|\mathbf{x}) \quad (8)$$

$$= \frac{e^{-FreeEnergy(\mathbf{x},\mathbf{e_c})}}{\sum_{c'=1\ldots C} e^{-FreeEnergy(\mathbf{x},\mathbf{e}_{c'})}} \quad (9)$$

where $\mathbf{x}$ is the input vector, and $\mathbf{y}$ is a vector that is a *1-of-C* representation of the class (also known as *one-hot* encoding), with C being the number of classes. If $\mathbf{x}$ belongs to a class c, then $\mathbf{y} = \mathbf{e}_c$, where $\mathbf{e}_c$ is a vector with all values set to 0 except at position c. With respect to the RBM, $\mathbf{x}$ and $\mathbf{y}$ together make up the visible layer $\mathbf{v}$.

Assuming a training set $\mathcal{D}_{train} = \{(\mathbf{x}_i, y_i)\}$ where $\mathbf{x}_i$ and $y_i \in \{1, \ldots, C\}$ are the i-th input vector and target class respectively, training the RBM generatively involves minimizing the negative log-likelihood

$$\mathcal{L}_{gen}(\mathcal{D}_{train}) = -\sum_{i=1}^{|\mathcal{D}_{train}|} \log p(\mathbf{x}_i, \mathbf{y}_i) \,. \quad (10)$$

The RBM thus used in a discriminative manner, forms the basis of the prediction model described in the next section.

4. A DISTRIBUTED MODEL FOR USE WITH MULTIPLE VIEWPOINTS

The prediction model we present in this paper models the conditional distribution $p(s_t|s_{(t-n+1)}^{(t-1)})$. It places no restrictions on the *types* associated with events in the context $s_{(t-n+1)}^{(n-1)}$ (*input type*), or the predicted event s_t (*target type*). In the simplest case, both are the same. In the case where they are different, the performance of the model depends on how informative the input types are of the target type. In the present work, we demonstrate this model with two cases where (1) both the input and target viewpoint types are musical pitch, and (2) the input types are pitch and duration, and the target type pitch. The choice of the additional input type in the second case was motivated by simplicity and to lay emphasis on the representation.

For each monophonic melody (in MIDI format) in a given dataset, sequences of the relevant input and target types are first extracted using the MIDI Toolbox [9]. These values are encoded as binary 1-of-$|\tau|$ vectors, where $|\tau|$ is the size of the domain of type τ. In the case where more than one input type exists, their corresponding vectors are simply concatenated. Such an idea is similar to that of the linked viewpoint type proposed in [7]. There are however, two important distinctions between the two. Firstly, the input and target types must be identical in the case of the n-gram models originally proposed for use with multiple-viewpoint systems, whereas this is not a requirement for the RBM model. Secondly, a linked viewpoint between two arbitrary types τ_1 and τ_2 of domain sizes $|\tau_1|$ and $|\tau_2|$ respectively, would have a domain of size $|\tau_1| \times |\tau_2|$ in the case of the n-gram models. Thus, for subsequences of length n, the number of free parameters to be estimated are $(|\tau_1| \times |\tau_2|)^n$ in the worst case. In contrast, the number to be estimated in case of the RBM model, with q hidden units and r visible units, is $(q \times r) + q + r$, where $r = (n-1) \times [(|\tau_1| + 1) + (|\tau_2| + 1)] + |\tau_3|$, and τ_3 the target type. The additional visible unit added to the representation of each of the input types τ_1 and τ_2 in the context is 1 when the corresponding event is absent at the start of a melody. Such a model only scales linearly with the length of the learned subsequences as well as the domain size of each of the involved viewpoint types (assuming q is constant). Its structure is depicted in Figure 2. Here we considered only those cases with a single target type.

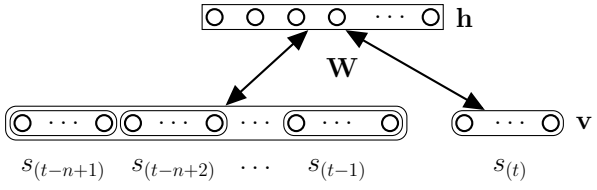


Figure 2. The structure of the prediction model. The set of nodes in the visible layer grouped together on the left make up the context $s_{(t-n+1)}^{(t-1)}$ of the input type(s). The set of nodes $s_{(t)}$ to the far right corresponds to the target type.

To train the model generatively, a subsequence $s^t_{(t-n+1)}$ is clamped to all the nodes in the visible layer. Training is done using the first instantiation of the Contrastive Divergence learning algorithm (CD-1). This simply means that the model parameters are updated after a single step of Gibbs sampling [11]. During prediction, the probability of each of the possible pitches in the prediction space is determined using (9). The distribution generated in this way does not require any kind of smoothing operation for unseen subsequences unlike n-gram models, where in [21] an empirical evaluation of different smoothing techniques was found necessary to establish the most reliable one.

5. EVALUATION

In order to evaluate the proposed prediction model, we make a comparison to a previous study of n-gram models for music prediction in [21]. There, *cross-entropy* was used to measure the information content of the models. This quantity is related to *entropy*, which is defined as

$$H(p) = -\sum_{s \in S} p(s) \log_2 p(s) \; . \tag{11}$$

where $p(s \in S) = p(\chi = s)$ is the probability mass function of a random variable χ distributed over a discrete alphabet $S = \{s_1, \ldots, s_k\}$ such that the individual probabilities are independent and sum to 1. The value of entropy, with reference to a prediction model, is a measure of the uncertainty of its predictions. A higher value reflects greater uncertainty. In practice, one rarely knows the true probability distribution of the stochastic process and uses a model to approximate the probabilities in (11). An estimate of the goodness of this approximation can be measured using cross-entropy (H_c) which represents the divergence between the entropy calculated from the estimated probabilities and the source model. This quantity can be computed over all the subsequences of length n in the test data $\mathcal{D}_{test}$, as

$$H_c(p_{mod}, \mathcal{D}_{test}) = \frac{-\sum_{s_1^n \in \mathcal{D}_{test}} \log_2 p_{mod}(s_n | s_1^{(n-1)})}{|\mathcal{D}_{test}|} \tag{12}$$

where p_{mod} is the probability assigned by the model to the last pitch in the subsequence given its preceding context. Cross-entropy approaches the true entropy as the number of test samples ($|\mathcal{D}_{test}|$) increases.

Evaluation was carried out on a corpus of monophonic MIDI melodies that cover a range of musical styles. The corpus is a collection of 8 datasets containing a total of $54,308$ musical events and was also used to evaluate n-gram models for music prediction in [21]. There, two different models were evaluated both individually and in combination. The first of these was a Long-Term Model (LTM), that was governed by structure and statistics induced from a large corpus of sequences from the same musical style. And the other was a Short-Term Model (STM) which relied on structure and statistics particular to the melody being predicted. The prediction model presented here deals only with long-term effects that are induced from a corpus, and is thus compared with the two best performing LTMs in [21] of unbounded order (labelled there as *C*I*) and order bound 2 respectively. To facilitate a direct comparison between the two approaches, the melodies are not transposed to a default key.

For the RBM model, different hyperparameters were evaluated through a grid search over the learning rate $\lambda = \{0.01, 0.05\}$, the number of hidden units $n_{hid} = \{100, 200, 400\}$, and the weight-cost $w_{cost} = \{0.0001, 0.0005\}$. Each model was trained using mini-batch gradient descent over 500 epochs with a batch size of 100 samples. The momentum μ, was set to 0.5 during the first five epochs and then increased to 0.9 for the rest of the training. Each model was evaluated with 10-fold cross-validation.

We carry out three types of evaluation. The first measures the information content of the pitch-only version of the proposed model using cross-entropy, and compares it to the n-gram models of [21]. It was observed that the RBM model compares favourably with the best of the n-gram models by making better use of information in longer contexts. In the second evaluation, we compare a variant of the model with input types pitch and duration and target type pitch to its pitch-only counterpart. And lastly, we combine these two models using mixture-of-experts and demonstrate how this can further improve the model performance in comparison to the individual models.

The first evaluation is carried out with cross-validation separately for each of the individual datasets. The context length is varied between 1 and 8. It was found that the RBM models with context length greater than 2 perform better than corresponding n-gram models on average. This is illustrated in Figure 3. An RBM model of suitable context length perform marginally better than the best-performing n-gram model — that of unbounded order. The same is the case with the best bounded-order n-gram model (of context length 2) and the RBM model of the same context length. While it was found that the performance of bounded order n-gram models tends to worsen on further increasing the context length, the performance of RBM models continues to improve until a context length of 4. The value of n where the RBM model performs better than the n-gram models of unbounded order is different on different datasets, and typically occurs between $n = 3$ and $n = 7$. The best average model cross-entropy of 2.819 is reached for a context length of 4. For models using longer contexts an increase in training performance was accompanied by a slight worsening of test performance, indicating overfitting. We suspect that the overall performance of the RBM models can be further improved with an optimized grid-search strategy in the hyper-parameter space, but leave this to be explored in the future. The optimal number of hidden units in our search was 100 across all datasets for almost all context lengths, leading to a linear increase in model size with context length.

In the second evaluation, we compared the cross-entropies of the single and multiple input type models (pitch and pitch with duration respectively) using the same target type (pitch), on the Bach chorale subset of the corpus. The re-

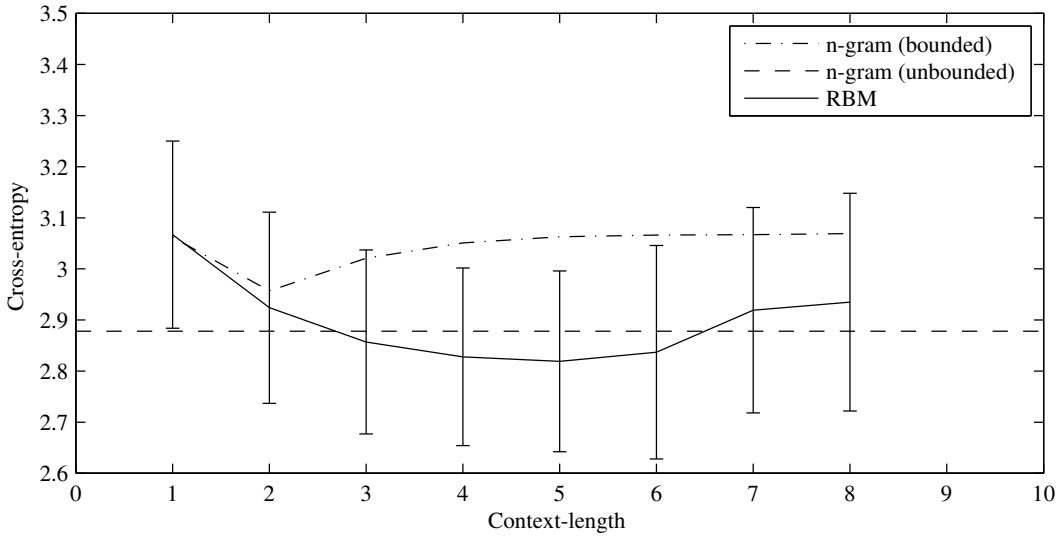


Figure 3. Variation in average cross-entropy of the prediction models with context length l (with standard deviation across folds for the RBM model). The cross-entropy of the RBM models progressively decreases until $l = 4$, while that of the n-gram models evaluated in [21] is minimal at $l = 2$ and increases thereafter. The performance of the n-gram model of unbounded order is indicated by the dashed line.

sults are shown in Table 1. The choice of adding duration was motivated by simplicity but the results show that it was not ideal for improving predictions. This conclusion is also supported by a similar trend observed with n-gram models, where a small deterioration in performance was observed on adding duration. The RBM model shows small performance improvements for some context lengths. This indicates that the representation for multiple input types proposed in Section 4 as an alternative to the linked viewpoints may indeed be effective.

l	1	2	3	4
n-gram (p)	2.737	2.565	2.505	2.473
n-gram (p + d)	2.761	2.562	2.522	2.502
RBM (p)	2.698	2.530	2.490	2.470
RBM(p + d)	2.660	2.512	2.481	2.519
RBM (combined)	2.663	2.486	2.462	2.413

Table 1. Cross-entropies of the single (pitch) and multiple (pitch, duration) input type RBM models and their combination over a range of context lengths l on the Bach chorales dataset. The individual RBM models compare favourably with corresponding n-gram models.

To illustrate the application of the proposed RBM model to multiple viewpoints for music prediction, we combine the pitch-only and the pitch & duration models. We use a simple mixture-of-experts model, i.e., take the arithmetic mean of the distributions each of the two models predicts for pitch. The results of this are listed in the third row of Table 1 and show an improvement over individual models.

6. CONCLUSIONS & FUTURE WORK

We presented a distributed model based on the Restricted Boltzmann Machine for multiple-viewpoint music prediction. It was demonstrated how such a model can be a scalable alternative to n-gram models for simultaneously modelling sequences of multiple musical features. The proposed model was evaluated in comparison with n-gram models and was found to compare favourably with them. It is able to make better use of information in longer event contexts than n-gram models, and also scales linearly with context length.

In the future, we would first like to address some of the issues left open in the present research. These include experiments with more promising viewpoint-type combinations as reported in [7] and [20], the use of alternative data fusion techniques like the weighted mixture- and product-of-experts [20], and further optimization of the existing model parameters. Previous research suggests that combining the LTM and STM improves prediction performance [7, 20] and, in fact, the combined n-gram model reported in [20] (mean cross-entropy: 2.479 for all datasets; 2.342 for the chorale dataset) outperforms the long-term RBMs examined here. Given the improved performance of these long-term RBMs, we expect adding a short-term component will yield the best prediction performance yet observed for this corpus. Extensions of the present model to handle polyphony and higher-level musical structure will also be explored. We would also like to apply the prediction model described here to some of the MIR tasks listed in Section 1. The present model can be potentially extended into a deep network, as demonstrated in [11], which is expected to improve its performance further.

7. ACKNOWLEDGEMENTS

Srikanth Cherla is supported by a Ph.D. studentship from City University London. The authors would like to thank Son Tran for many useful discussions on RBMs, and the reviewers for their valuable feedback on the paper.

8. REFERENCES

[1] Charles Ames. The Markov Process as a Compositional Model: A Survey and Tutorial. *Leonardo*, 22(2):175–187, 1989.

[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

[3] Greg Bickerman, Sam Bosley, Peter Swire, and Robert Keller. Learning to Create Jazz Melodies using Deep Belief Nets. In *International Conference On Computational Creativity*, 2010.

[4] John Biles. Genjam: A genetic algorithm for generating jazz solos. In *Proceedings of the International Computer Music Conference*, pages 131–131, 1994.

[5] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[6] Darrell Conklin. Multiple viewpoint systems for music classification. *Journal of New Music Research*, 42(1):19–26, 2013.

[7] Darrell Conklin and Ian H Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.

[8] David Cope. *Experiments in musical intelligence*, volume 12. AR Editions Madison, WI, 1996.

[9] Tuomas Eerola and Petri Toiviainen. MIR in Matlab: The Midi Toolbox. In *Proceedings of the International Conference on Music Information Retrieval*, pages 22–27. Universitat Pompeu Fabra Barcelona, 2004.

[10] Joachim Ganseman, Paul Scheunders, Gautham J Mysore, and Jonathan S Abel. Evaluation of a Score-informed Source Separation System. In *International Society for Music Information Retrieval Conference (ISMIR)*, pages 219–224, 2010.

[11] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[12] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18:1527–1554, 2006.

[13] Robert M Keller and David R Morrison. A Grammatical Approach to Automatic Improvisation. In *Sound and Music Computing Conference*, pages 11–13, 2007.

[14] Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted Boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 536–543. ACM Press, 2008.

[15] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting Structured Data*, 2006.

[16] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[17] Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2008.

[18] Michael C Mozer. Connectionist music composition based on melodic, stylistic and psychophysical constraints. *Music and connectionism*, pages 195–211, 1991.

[19] Francois Pachet. The continuator: Musical interaction with style. *Journal of New Music Research*, 32(3):333–341, 2003.

[20] Marcus Pearce. *The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Composition*. PhD thesis, 2005.

[21] Marcus Pearce and Geraint Wiggins. Improved methods for statistical modelling of monophonic music. *Journal of New Music Research*, 33(4):367–385, 2004.

[22] Claude E. Shannon. A Mathematical Theory of Communication. *The Bell System Techincal Journal*, 27(July):379–423, 623–656, 1948. Reprinted in *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[23] Paul Smolensky. Parallel distributed processing: explorations in the microstructure of cognition, vol. 1. chapter Information processing in dynamical systems: foundations of harmony theory, pages 194–281. MIT Press, Cambridge, MA, USA, 1986.

[24] Athina Spiliopoulou and Amos Storkey. Comparing probabilistic models for melodic sequences. In *Machine Learning and Knowledge Discovery in Databases*, pages 289–304. 2011.

[25] 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.

[26] Raymond Whorley, Christophe Rhodes, Geraint Wiggins, and Marcus Pearce. Harmonising melodies: Why do we add the bass line first? In *International Conference on Computational Creativity*, pages 79–86, 2013.

LEARNING RHYTHM AND MELODY FEATURES WITH DEEP BELIEF NETWORKS

Erik M. Schmidt and Youngmoo E. Kim
Music and Entertainment Technology Laboratory (MET-lab)
Electrical and Computer Engineering, Drexel University
{eschmidt, ykim}@drexel.edu

ABSTRACT

Deep learning techniques provide powerful methods for the development of deep structured projections connecting multiple domains of data. But the fine-tuning of such networks for supervised problems is challenging, and many current approaches are therefore heavily reliant on pre-training, which consists of unsupervised processing on the input observation data. In previous work, we have investigated using magnitude spectra as the network observations, finding reasonable improvements over standard acoustic representations. However, in necessarily supervised problems such as music emotion recognition, there is no guarantee that the starting points for optimization are anywhere near optimal, as emotion is unlikely to be the most dominant aspect of the data. In this new work, we develop input representations using harmonic/percussive source separation designed to inform rhythm and melodic contour. These representations are beat synchronous, providing an event-driven representation, and potentially the ability to learn emotion informative representations from pre-training alone. In order to provide a large dataset for our pre-training experiments, we select a subset of 50,000 songs from the Million Song Dataset, and employ their 30-60 second preview clips from 7digital to compute our custom feature representations.

1. INTRODUCTION

Deep learning is rapidly becoming one of the most popular topics in the machine learning community, and such approaches offer powerful methods for finding deep structured connections in data. But the success of these methods often hinges on pre-training, or unsupervised methods that are used to provide a starting point to perform gradient descent optimization. As there is no guarantee of convexity in these problems, finding a useful initial starting point is paramount, as the best case scenario is generally limited to finding a good local minima.

In previous work, we have looked into deep learning methods for the prediction of musical emotion [1–3]. Deep belief networks (DBNs) were trained on magnitude spectra observations with the goal of predicting Arousal-Valence (A-V) coordinates, where valence indicates positive versus negative emotion, and arousal indicates emotional intensity. Using these models, the individual layers were treated as basis functions for feature extraction, and the learned representations were shown to outperform standard music information retrieval (Music-IR) features (e.g., mel-frequency cepstral coefficients).

But in looking to further improve these approaches, many questions remain in the pre-training methodology. Unsupervised methods such as restricted Boltzman machine (RBM) pre-training reduce the dimensionality of data based on the most prominent aspects. For instance, very compelling results have been shown on text data, where reducing text documents to two dimensions related directly to document type [4]. If the goal is to build a document type classifier, then this approach will yield an excellent starting position, but if it is document emotion we wish to model, then such a starting point may be no better than random. The same is true in music; if we cannot have the expectation of learning useful domains from unsupervised pre-training, then we should have low expectations for the supervised fine-tuning.

In this new work, we develop DBN input representations specifically designed to allow the model to learn about rhythm and melodic contour in an unsupervised fashion. Learning to understand these phenomena necessarily requires the ability to parse musical events, and we therefore begin with beat tracking, such that the models can be provided with a history of feature snapshots at musically relevant time points. In all of our feature extraction, we utilize harmonic/percussive source separation (HPSS) [5], allowing us to deconstruct the spectrum, separating out melody and harmonic sources from drums and percussive sources.

With the percussive spectra, we compute a beat synchronous percussion timbre feature, allowing us to parse different drum sounds and construct rhythm models by analyzing a history of samples. For the harmonic spectra, we compute a 48-dimensional beat synchronous chroma representation that allows the ability to track melodic contour over multiple octaves. In addition, we investigate the use of the 2-d FFT of beat synchronous chroma over four beat

segments, providing a shift (transposition) invariant feature for melodic contour that has been shown to be successful in cover song recognition [6].

In order to provide a reasonable dataset for pre-training we employ a set of 50,000, 30-60 second audio clips from 7digital that were randomly selected from the Million Song Dataset [7]. The DBNs are fine-tuned for predicting musical emotion using a publicly available dataset of time-varying musical emotion data [8].

2. BACKGROUND

Deep learning and DBN based feature learning is a topic of expanding attention in the machine listening community [9]. Lee *et al.* was one of the first to apply deep belief networks to acoustic signals, employing an unsupervised convolutional approach [10]. Their system employed PCA to provide a dimensionality reduced representation of the magnitude spectrum as input to the DBN and showed slight improvement over MFCCs for speaker, gender, and phoneme detection.

Hamel and Eck applied DBNs to the problems of musical genre identification and autotagging [11]. Their approach used raw magnitude spectra as the input to their DBNs, which were constructed from three layers and employed fifty units at each layer. The system was trained using greedy-wise pre-training and fine-tuned on a genre classification dataset, consisting of 1000, 30-second clips. The learned representations showed reasonable increases in performance over standard feature representations on both genre recognition and autotagging. The authors have also found significant improvement in moving to multi-timescale representations [12, 13].

Battenberg and Wessel applied conditional DBNs in modeling drum patterns in recent work, which incorporated an autoregressive time-varying restricted Boltzman machine model that can be used for generating sequences [14]. One downside of the conditional RBM for the application discussed in this new work is that the input history (past samples) only contributes to the bias term between the visible and hidden layer, and therefore the full information about rhythm may not be available in the upper hidden model layers.

3. DATA COLLECTION

In this paper, we use a universal background model style pre-training, initializing our models on a dataset of 50,000 songs, followed by fine-tuning on a 240 song labeled dataset of 15-second clips annotated with A-V emotion.

3.1 Unsupervised Pre-Training Data

For the unsupervised pre-training phase we seek to employ a large dataset in order to expose our model to a wide distribution of musical data. As such, we select a subset of 50,000 tracks from the Million Song Dataset (MSD). As the MSD includes only proprietary features, and we seek to handcraft original domains, we employ their 30-60 second preview clips from the 7digital API [1]. In order to ensure quality audio, we first download clips for the entire MSD and filter out any songs with less than 128 kbps MP3 bitrate, lower than 22050 Hz sampling rate, clips shorter than 30 seconds, clips that were found to be silent, and ones that had bad frames or file corruption issues.

3.2 Supervised Fine-Tuning Dataset

For the supervised musical emotion training dataset, we employ a corpus annotated in previous work using Amazon's Mechanical Turk (MTurk) [8]. The dataset contains 240, 15-second song clips that were sampled from a larger corpus that was annotated at 1-second intervals using a game-based approach. Each song clip was selected to ensure a uniform distribution was provided across the four quadrants of the A-V space. The goals of the MTurk activity were to assess the effectiveness of the game and to determine any biases created though collaborative labeling. Overall, the datasets were shown to be highly correlated, with arousal $r = 0.712$, and valence $r = 0.846$. The MTurk dataset is available to the research community [2] and is densely annotated, containing $4,064$ label sequences in total (16.93 ± 2.690 ratings per song).

4. ACOUSTIC REPRESENTATIONS

As previously discussed, learning to understand musical attributes, such as rhythm and melody, necessarily requires the ability to parse musical events. As such, the success of these methods hinges on our ability to accurately beat track music audio. All acoustic representations developed in this work employ harmonic/percussive source separation. With beat tracking, HPSS allows us to find the best onsets possible using percussive spectra. With rhythm features, it allows us to isolate just percussion (or percussive contributions of other pitched instruments), and to create features based on the timbre of percussion on the beat. Finally, with pitch features, it allows us to isolate harmonic (pitched) sources when creating our chroma representations. Figure 1 shows the feature extraction process for each stage in our processing. Beat tracking is shown in the center, and percussion and pitch features are on the left and right, respectively.

4.1 Harmonic/Percussive Source Separation

As the time/frequency considerations are different for each of our feature extraction chains (i.e., beat tracking, pitch, percussion timbre), we must perform HPSS three times for each of our 50,000 pre-training songs. As a result, we elect to use an efficient median filtering based approach [5]. The general idea of HPSS is that harmonic signals correspond to horizontal lines in the spectrogram (i.e., Fourier series) and percussive signals correspond to vertical lines (i.e., impulses). In Figure 2, we show a simple audio example of a guitar and drums mix.

[1] http://developer.7digital.net/

[2] http://music.ece.drexel.edu/research/emotion/moodswingsturk

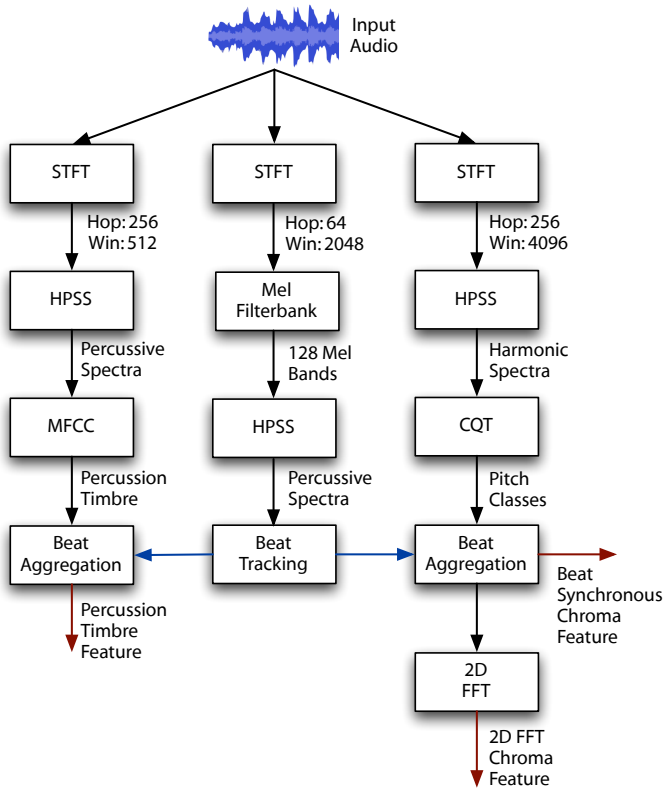


Figure 1. Feature extraction process for percussion timbre (left), beat detection (center), and pitch chroma representations (right).

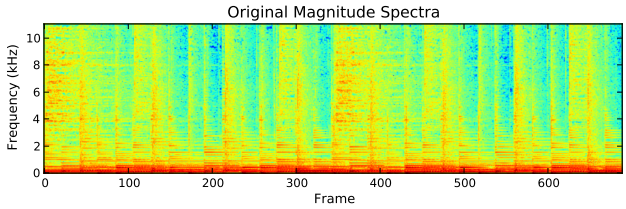


Figure 2. Original audio input spectra.

Median filtering based HPSS performs two passes of median filtering (vertically and horizontally) in order to generate spectral masks, and is therefore extremely efficient. Figure 3 shows the HPSS separation for the spectrogram shown in Figure 2.

4.2 Beat Tracking

Our beat tracking approach begins with an STFT with a 64 frame hop ($\sim$ 3msec) to provide maximal time resolution, followed by the application of a 128 bin mel-spaced filter bank that provides vertical smoothing in the spectrum, thus making percussive onsets more prominent. Following the filter bank, the onset profile is computed via a multidimensional Laplace filter, and the filter means are used in an Ellis-style beat tracker using librosa [3] [15].

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

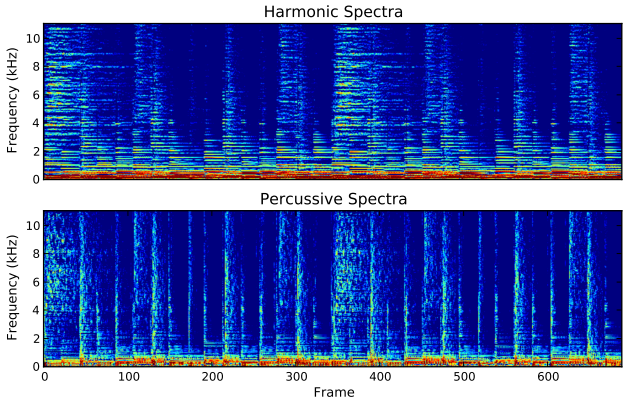


Figure 3. Harmonic percussive source separation.

4.3 Percussion Timbre

In order to train a model to understand rhythm, we extract a percussion timbre feature. This feature is shown in the left column of Figure 1, where we first extract the STFT with a window size of 512 samples ($\sim$ 23msec) and hop size of 256 ($\sim$ 11.6msec). We then compute HPSS, followed by MFCCs of the percussive spectra, providing a percussion timbre feature (e.g., to differentiate the boom sound of a bass drum vs. the hit of a snare). We then beat aggregate this feature such that the DBN is provided with event-driven feature updates to learn rhythmic styles. As shown in Figure 4, we can parse rhythm visually.

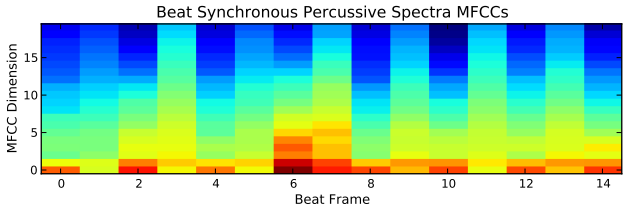


Figure 4. Beat synchronous aggregation of mel-frequency cepstral coefficients computed from the percussive spectrogram (percussion timbre).

4.4 Pitch Chroma

Following a similar pattern to the percussion timbre, we begin our chroma representation with HPSS as well, but with a much larger STFT window size. Here we use a 4096 ($\sim$ 186msec) window in order to provide reasonable frequency precision on bass frequencies. Next, we apply a magnitude constant-Q transform (CQT) filter bank starting at G_2 (98Hz), the first CQT filter that fits comfortably into our STFT representation, and spanning four octaves up to $F\sharp_6/G\flat_6$ (1479.98). Figure 5 displays our 48-dimensional chroma representation.

To learn a model of how the chroma evolve, we will need to present the DBN with multiple frames, and we therefore elect to center those event around beats, as we can have a reasonable expectation of a correlation with note onsets. This also greatly reduces the number of train-

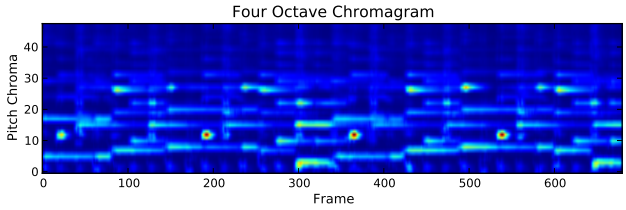


Figure 5. Four octave pitch chroma.

ing frames in our dataset, making the approach more computationally feasible. Figure 6 shows the beat aggregation of our chroma feature.

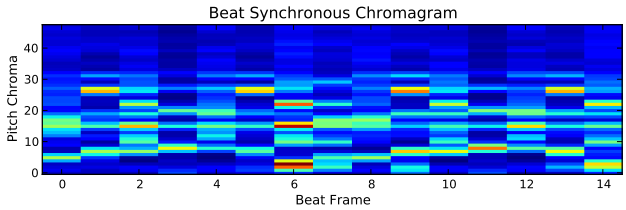


Figure 6. Beat synchronous chroma.

4.5 Chroma 2-d FFT

For our DBN input representation, we investigate using the magnitude 2-d FFT of our beat synchronous chroma representation. This feature was previously investigated in the realm of cover song detection, where it was found to perform well using 75-beat patches, providing shift (transposition) and time invariant properties for melody within a song [6]. Here we shorten this observation to just four beats, with the goal of obtaining shift/transposition invariance, but still retaining time information. Figure 7 shows this feature, where it is computed for each shift of a four beat window.

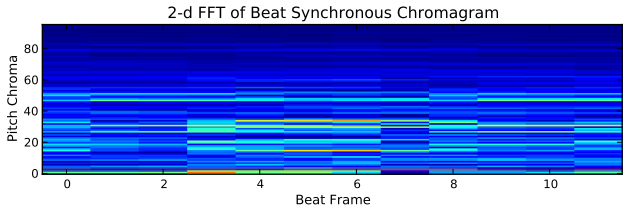


Figure 7. 2-d FFT of beat synchronous chroma.

5. DEEP BELIEF NETWORKS

A trained deep belief network shares an identical topology to a neural network, though they offer a far-superior training procedure, which begins with an unsupervised pre-training that models the hidden layers as restricted Boltzman machines (RBMs) [4, 16, 17] . A graphical depiction of our first layer RBM is shown in Figure 8, which uses four beat synchronous frames of observations in the input. An RBM is a generative model that contains only a single hidden layer, and in simplistic terms they can be thought of as two sets of basis vectors, one which reduces the dimensionality of the data and the other that reconstructs it.

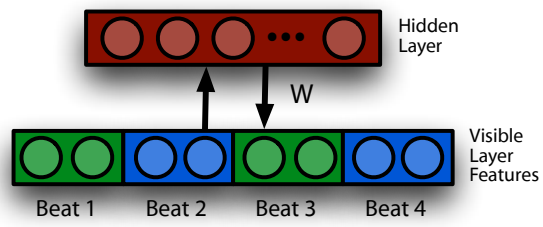


Figure 8. A restricted Boltzman machine with multiple beat observations.

RBMs are Markov random fields (MRFs) with hidden units, in a two layer architecture where we have visible units $\mathbf{v}$ and hidden units $\mathbf{h}$. During pre-training, we learn RBMs "greedily," where we learn them one at a time from the bottom up. That is, after we learn the first RBM we retain only the forward weights, and use them to create the input for training the next RBM layer.

For our first layer RBM we employ a Gaussian-binomial RBM, where the visible data is represented as a Gaussian distribution. The advantage of this representation over the standard binomial-binomial is that a sigmoid function is not applied during inference when estimating the visible layer from the hidden. With the standard binomial units, most visible values are forced to 0 or 1,

$$p(v_i = 1|\mathbf{h}) = \sigma(b_i + \sum_j w_{ij} h_j), \tag{1}$$

where the visible layer is $\mathbf{v} \in \mathbb{R}^{1\times I}$, the hidden layer $\mathbf{h} \in \mathbb{R}^{1\times J}$, and the model has parameters $\mathbf{W} \in \mathbb{R}^{I\times J}$, with biases $\mathbf{c} \in \mathbb{R}^{1\times J}$ and $\mathbf{b} \in \mathbb{R}^{1\times I}$.

The Gaussian-binomial RBM allows a more continuous range,

$$p(v_i|\mathbf{h}) = \mathcal{N}(b_i + \sum_j w_{ij} h_j, 1). \tag{2}$$

During our greedy-wise pre-training we use Gaussian-binomial RBMs at the first layer, which presents continuous data, but all subsequent layers use standard binomial-binomial RBMs.

For the Gaussian-binomial RBM, we have an energy distribution of the form,

$$E(\mathbf{v}, \mathbf{h}) = \sum_{i \in \text{visible}} \frac{(v_i - b_i)^2}{2} - \sum_{j \in \text{hidden}} c_j h_j - \sum_{i,j} v_i h_j w_{ij}, \tag{3}$$

and the standard binomial-binomial RBM has an energy function of the form,

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i \in \text{visible}} b_i v_i - \sum_{j \in \text{hidden}} c_j h_j - \sum_{i,j} v_i h_j w_{ij}. \tag{4}$$

As in the typical approach to deep learning, after pre-training we form a multi-layer perceptron using only the forward weights of the RBM layers. As our goal is to learn feature detectors for a regression problem, we lastly attach a linear regression layer and report the prediction error for fine-tuning as the mean squared error of the estimators. We

trained our DBNs using Theano, [4] a Python-based package for symbolic math compilation.

6. EXPERIMENTS AND RESULTS

In the following experiments, we investigate employing DBNs for rhythm and melody feature learning. For each DBN, we use shrinking layer sizes, where layer 0 contains 75 nodes, layer 1 contains 50 nodes, and layer 2 contains 25 nodes. The goal with this approach is to best take advantage of the dimensionality reduction power of RBMs.

For our pre-training dataset, we use the 50,000 7digital preview clips described in Section 3.1, with beat synchronous features that are represented by taking every shift of a 4 beat window, and vectorizing the input as shown in Figure 8. For each feature type, we show the pre-training visible data dimensionality and total number of examples below in Table 1.

DBN Model Domain	Input Dimensionality	Number of Pre-Training Examples
Rhythm	80	$4,645,595$
Pitch Chroma	192	$4,645,526$
2-d FFT Chroma	384	$4,495,526$

Table 1. DBN pre-training Data

Pre-training epochs of 5, 10, 20, and 50 are investigated for all feature types with a learning rate of 10^{-5}. The best validation scores were found for 10 epochs with both the chroma and 2-d FFT of chroma, and 50 epochs with the percussion timbre representation. For each pre-trained model, we perform gradient descent back propagation fine-tuning for each fold, where for each input example $\mathbf{x}_i$, we train the model to produce the emotion space parameter vector $\mathbf{y}_i$,

$$\mathbf{y}_i = [\mu_a,\ \mu_v]. \tag{5}$$

In performing fine-tuning we note that our DBNs are beat-synchronous, but our labeled data is annotated at one-second intervals. In order to fine-tune our DBNs to predict emotion, we use linear interpolation to estimate the values of emotion on the beat. However, since we seek to compare this method to that of previous work, it necessarily must be evaluated on the second-by-second data test set. Therefore, after DBNs are trained and layer-wise features are computed, we then aggregate DBN features over the past 1-second, as is done with the standard feature domains, providing features at the same rate as the original labels.

We evaluate these learned representations in the context of multiple linear regression (MLR), as we have investigated in prior work [18–20], where we develop regressors to predict the parameterization vector $\mathbf{y}_i$ of a two-dimensional Gaussian in A-V space,

$$\mathbf{y}_i = [\mu_a,\ \mu_v,\ \sigma^2_{aa},\ \sigma^2_{vv},\ \sigma^2_{av}]. \tag{6}$$

[4] http://deeplearning.net/software/theano/

In all supervised experiments, the model training is cross-validated 5 times, dividing the dataset into 50% training, 20% verification, and 30% testing. To avoid the well-known album-effect, we ensured that any songs that were recorded on the same album were either placed entirely in the training or testing set. Note, for three songs in the dataset, the beat tracker returned less than four beats in the labeled portion of the song, and as a result they had to be removed from the sets. Those songs are IDs: 2996, 5232, 6258. We post updated results for standard features over previous work in Table 2, and note that their removal leaves the results nearly unchanged [3].

As in previous approaches, we use Euclidean distances to evaluate our A-V mean predictions in a normalized space (i.e., axes bound between -0.5 and 0.5), and Kullback-Liebler divergences to analyze our Gaussian predictions. We note a slight difference in the KL divergence calculation from our previous work,

$$\mathrm{KL}(p||q) = \frac{1}{2}\Big(\log\frac{|\Sigma_q|}{|\Sigma_p|} + \mathrm{tr}(\Sigma_q^{-1}\Sigma_p) + (\mu_q-\mu_p)^T\Sigma_q^{-1}(\mu_q-\mu_p) - d\Big), \tag{7}$$

where in previous work we had omitted the $\frac{1}{2}$ multiplier term (see [20]).

Feature Type	Average Mean Distance	Average KL Divergence
MFCC	0.140 ± 0.004	0.642 ± 0.069
Chroma	0.182 ± 0.005	1.654 ± 0.143
Spectral Shape	0.153 ± 0.005	0.755 ± 0.074
Spectral Contrast	$\mathbf{0.139 \pm 0.005}$	$\mathbf{0.647 \pm 0.072}$
ENT	0.151 ± 0.005	0.700 ± 0.079

Table 2. Emotion regression results from previous work for fifteen second clips.

Results for the different DBN feature types are shown in Table 3. For each learned feature type, we investigate that feature alone, as well as that feature in combination with the others. As we pull the spectrum apart with HPSS to learn the different DBN features, it makes sense that we should put the two domains back together for prediction. The rhythm feature, which uses the percussion timbre as input, is the best single performing feature at a mean error of 0.128, and the best result overall is when combining the pitch and 2-d FFT of chroma at 0.113 mean error.

7. DISCUSSION AND FUTURE WORK

This work presented a novel approach for training deep belief networks for understanding rhythm and melody. The fine-tuned DBN features easily outperformed any other singular existing representation, and the combination of the rhythm and melody DBN features outperformed any other system previously tested on this dataset.

In moving forward with deep learning approaches that require pre-training, we believe that it should be based around input observations from which high level musical

DBN Layer	DBN Feature Type	Pre-training Model Error		Fine-tuning Model Error	
		Mean Distance	KL Divergence	Mean Distance	KL Divergence
Layer 0	Rhythm	$\mathbf{0.146 \pm 0.007}$	$\mathbf{0.681 \pm 0.083}$	0.139 ± 0.009	0.652 ± 0.085
Layer 1	Rhythm	0.148 ± 0.007	0.708 ± 0.086	0.132 ± 0.013	0.598 ± 0.094
Layer 2	Rhythm	0.156 ± 0.006	0.754 ± 0.086	$\mathbf{0.128 \pm 0.016}$	$\mathbf{0.582 \pm 0.104}$
Layer 0	Pitch	$\mathbf{0.160 \pm 0.004}$	$\mathbf{0.786 \pm 0.105}$	0.143 ± 0.015	0.678 ± 0.122
Layer 1	Pitch	0.161 ± 0.005	0.792 ± 0.086	0.131 ± 0.022	0.618 ± 0.149
Layer 2	Pitch	0.165 ± 0.006	0.815 ± 0.095	$\mathbf{0.129 \pm 0.024}$	$\mathbf{0.608 \pm 0.153}$
Layer 0	2-d FFT Pitch	$\mathbf{0.171 \pm 0.007}$	$\mathbf{0.889 \pm 0.103}$	0.148 ± 0.014	0.716 ± 0.132
Layer 1	2-d FFT Pitch	0.175 ± 0.007	0.926 ± 0.116	0.137 ± 0.022	0.669 ± 0.166
Layer 2	2-d FFT Pitch	0.175 ± 0.006	0.915 ± 0.112	$\mathbf{0.129 \pm 0.024}$	$\mathbf{0.620 \pm 0.170}$
Layer 0	Rhythm+Pitch	$\mathbf{0.145 \pm 0.006}$	$\mathbf{0.685 \pm 0.082}$	0.129 ± 0.012	0.604 ± 0.091
Layer 1	Rhythm+Pitch	0.147 ± 0.007	0.709 ± 0.078	0.117 ± 0.019	0.534 ± 0.122
Layer 2	Rhythm+Pitch	0.153 ± 0.007	0.743 ± 0.089	$\mathbf{0.114 \pm 0.022}$	$\mathbf{0.514 \pm 0.125}$
Layer 0	Rhythm+2-d FFT Pitch	$\mathbf{0.147 \pm 0.005}$	$\mathbf{0.707 \pm 0.075}$	0.131 ± 0.013	0.606 ± 0.098
Layer 1	Rhythm+2-d FFT Pitch	0.151 ± 0.006	0.739 ± 0.077	0.119 ± 0.019	0.552 ± 0.128
Layer 2	Rhythm+2-d FFT Pitch	0.156 ± 0.007	0.763 ± 0.082	$\mathbf{0.113 \pm 0.022}$	$\mathbf{0.514 \pm 0.131}$

Table 3. Emotion regression results for Mechanical Turk annotated clips. Rhythm features use percussion timbre as input, pitch features use beat synchronous chroma, and 2-d FFT pitch features use our four beat 2-d FFT of chroma representation. Feature combination results are all early fusion based (concatenation of dimensions).

ideas like rhythm, melody, and harmony can easily be extracted. Furthermore, as understanding these ideas necessarily requires the presentation of time-series data, future approaches should further investigate the best way to present this information to the first DBN layer.

In continuing this work, we wish to further analyze the optimal number of beat synchronous frames to present to the DBN input, as well as investigating smaller units of musical events, such as eighth or sixteenth note feature updates. It would also be interesting to apply these learned features in the context of a graphical model such as a conditional random field, as investigated in prior work [21].

8. ACKNOWLEDGMENT

This work is supported by National Science Foundation awards IIS-0644151 and CNS-0960061.

9. REFERENCES

[1] E. M. Schmidt and Y. E. Kim, "Learning emotion-based acoustic features with deep belief networks," in *IEEE WASPAA*, New Paltz, NY, 2011.

[2] ——, "Modeling the acoustic structure of musical emotion with deep belief networks," in *NIPS Workshop on Music and Machine Learning*, 2011.

[3] E. M. Schmidt, J. Scott, and Y. E. Kim, "Feature learning in dynamic environments: Modeling the acoustic structure of musical emotion," in *ISMIR*, Porto, Portugal, October 2012.

[4] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, July 2006.

[5] D. FitzGerald, "Harmonic/percussive separation using median filtering," in *DAFx*, Graz, Austria, September 2010.

[6] T. Bertin-Mahieux and D. P. W. Ellis, "Large-scale cover song recognition using the 2d fourier transform magnitude," in *ISMIR*, Porto, Portugal, October 2012.

[7] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, "The Million Song Dataset," in *ISMIR*, Miami, FL, October 2011.

[8] J. A. Speck, E. M. Schmidt, B. G. Morton, and Y. E. Kim, "A comparative study of collaborative vs. traditional annotation methods," in *ISMIR*, Miami, Florida, 2011.

[9] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics," in *ISMIR*, Porto, Portugal, October 2012.

[10] H. Lee, Y. Largman, P. Pham, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *NIPS*, 2009.

[11] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks," in *ISMIR*, Utrecht, Netherlands, 2010.

[12] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck, "Temporal pooling and multiscale learning for automatic annotation and ranking of music audio," in *ISMIR*, Miami, FL, October 2011.

[13] P. Hamel, Y. Bengio, and D. Eck, "Building musically-relevant audio features through multiple timescale representations," in *ISMIR*, Porto, Portugal, October 2012.

[14] E. Battenberg and D. Wessel, "Analyzing drum patterns using conditional deep belief networks," in *ISMIR*, Porto, Portugal, October 2012.

[15] D. P. W. Ellis, "Beat tracking by dynamic programming," *JNMR*, vol. 36, no. 1, pp. 51–60, March 2007.

[16] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[17] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *NIPS*, 2007.

[18] E. M. Schmidt, D. Turnbull, and Y. E. Kim, "Feature selection for content-based, time-varying musical emotion regression," in *ACM MIR*, Philadelphia, PA, 2010.

[19] E. M. Schmidt and Y. E. Kim, "Prediction of time-varying musical mood distributions from audio," in *ISMIR*, Utrecht, Netherlands, 2010.

[20] ——, "Prediction of time-varying musical mood distributions using Kalman filtering," in *IEEE ICMLA*, Washington, D.C., 2010.

[21] ——, "Modeling musical emotion dynamics with conditional random fields," in *ISMIR*, Miami, FL, 2011.

Poster Session 1

A COMPARATIVE STUDY OF INDIAN AND WESTERN MUSIC FORMS

Parul Agarwal[1], **Harish Karnick**[2]
Indian Institute of Technology, Kanpur, India
parulagarwal89@gmail.com[1]
hk@cse.iitk.ac.in[2]

Bhiksha Raj
Carnegie Mellon University, USA
bhiksha@cs.cmu.edu

ABSTRACT

Music in India has very ancient roots. Indian classical music is considered to be one of the oldest musical traditions in the world but compared to Western music very little work has been done in the areas of genre recognition, classification, automatic tagging, comparative studies etc. In this work, we investigate the structural differences between Indian and Western music forms and compare the two forms of music in terms of harmony, rhythm, microtones, timbre and other spectral features. To capture the temporal and static structure of the spectrogram, we form a set of global and local frame-wise features for 5- genres of each music form. We then apply Adaboost classification and GMM based Hidden Markov Models for four types of feature sets and observe that Indian Music performs better as compared to Western Music. We have achieved a best accuracy of 98.0% and 77.5% for Indian and Western musical genres respectively. Our comparative analysis indicates that features that work well with one form of music may not necessarily perform well with the other form. The results obtained on Indian Music Genres are better than the previous state-of-the-art.

1. INTRODUCTION

Due to technology advances the size of digital music collections have increased making it difficult to navigate such collections. Music content is very often described by its genre [1], though the definition of genre may differ based on culture, musicians and other factors. Considerable analysis has been done on western music for genre classification [2] [3] and content analysis [4].

Although, Indian music, especially classical music, is considered to be one of the oldest musical traditions in the world, not much computational analytical work has been done in this area. Indian music can be divided into two broad categories *classical* and *popular*. Classical music has two main variants *Hindustani classical* prevalent largely in north and central India and *Carnatic classical* prevalent largely in the south of India. Each variety has multiple genres. For example *Hindustani* music has *Dhrupad, Khayal, Tarana* as classical genres and *Thumri, Dadra, Tappa, Bhajan* as semi-classical genres - amongst others. Popular music has multiple folk genres based on region, film music and *adhunik* or modern music which is influenced by multiple Indian and western genres.

At the base of classical and semi-classical Indian music is a *raga(s)* which is described as a mood or sentiment expressed by a microtonal scale form [28]. A *raga* can be sung in many Indian genres like Dhrupad, Khayal, Thumri etc with its corresponding microtonal scale based on a natural harmonic series. Popular Indian music, on the other hand, is not necessarily based on *ragas* and can have tonal elements that can differ from the traditionally accepted classical norms. Almost all Indian music is based on melody - single notes played in a given order. Western music has strong harmonic content i.e. a group of notes called chords played simultaneously. Unlike Indian tonal system, the Western tonal system is divided into twelve equal intervals. Each Indian genre has its own well-defined structure which makes it different from other forms of music. To capture the inherent structure of Indian music and use it for genre classification is a challenging problem. Through our work we try to explore this area and focus on the following research questions:

- What are the structural differences between Indian and Western musical forms?
- How well can the audio features used for genre classification in Western music, capture the characteristics of Indian genres?
- Is there a feature set that can be used for cross-cultural music genre classification? In this case, for Indian and Western music.

2. RELEVANT WORK

2.1 Study on Genre classification by humans

Research has been done on the human ability to classify music genres. Perrot et al. [16](1999) reported that humans with little musical training were able to classify genres with an accuracy of about 72% on a ten-genre dataset of a music company, based on only 300 milliseconds of audio. Soltau (1997) conducted a Turing-test in which he trained 37 people to distinguish rock from pop music and found

that his automatic music classifier was similar to their performance on the same dataset. Lippens et al. [15](2004) conducted an experiment with six genres and observed that people were able to classify with an accuracy of around 90%. Considerable work has been done in the area of automatic music genre classification and with increasing improvement in classification methods there is a distinct possibility that they may outperform humans in the near future.

2.2 Genre classification of Western Music

Many supervised and unsupervised methods have been used to automatically classify recordings into genres. Commonly used supervised classification methods are k-nearest Neighbours (KNN) [5] [17], Support Vector Machines (SVMs) [19] [17], Gaussian Mixture Models (GMMs) [5] [20] [17], Linear Discriminant Analysis (LDA) [17] [18], non-negative Matrix Factorization (NMF) [2], Artificial Neural Networks (ANNs) and Hidden Markov Models (HMMs). In the unsupervised domain, there is work using the k-means approach and hierarchical clustering.

To the best of our knowledge, there is no work in the literature that analyses which features are best for music genre classification. There is work using Fourier analysis, cepstral analysis and wavelet analysis [17] [24]. Li et al. [17] and J. Bergstra et al. [3] worked with a set of parameters like spectral centroid, spectral rolloff, spectral flux, zero crossings and low energy. Y.Panagakis et al. [25] used sparse representation-based classifiers on audio temporal modulations of songs for genre classification. Salamon et al. [26] used a combination of low level features with high level melodic features derived from pitch contours and showed that the combination outperforms the results from using only low level features.

Tzanetakis et al. [5] proposed to classify musical genres using K -Nearest Neighbours and GMMs based on timbral texture and rhythmic features. E. Benetos et al. [2] used non-negative tensor factorization (NTF) with GTZAN data set [5]. J. Bergstra et al. [3] used GMM based Adaboost classification with aggregate features dividing frames into segments. Hidden Markov Models were used by T. Langlois et al. [10] for genre classification based on timbre information.

2.3 Genre classification of Indian Music

Compared to western music very little work has been done in Indian musical genre classification. S.Jothilakshmi et al. [12] used kNN and GMM on spectral and perceptual features and achieved an accuracy of 91.23% with 5 Indian genres- Hindustani, Carnatic, Ghazal, Folk and Indian western. P. Chordia et al. [13] used pitch profiles and Pitch-Class Distributions for *raga* identification. A. Vidwans et al. [14] extracted melodic contours to classify Indian classical vocals. They also performed a listening test in which around 85% samples were correctly identified by people without any training.

2.4 Contributions of the paper

There is no study, to the best of our knowledge, that compares classification of Indian musical genres with Western musical genres. Our work makes the following contributions: first, this paper introduces a new research problem on cross-cultural music genre classification by doing a comparative study on Indian and Western musical forms in terms of their structure and characteristics. Second, while there are many standard datasets on genre classification available for Western music like GTZAN dataset [5], 2005 MIREX dataset, Magnatune [21], USPOP [22] etc. There is no standard dataset for Indian genre classification. For our study, we built a dataset of 5 Indian genres- Dhrupad, Thumri, Carnatic, Punjabi and Ghazals. There are 100 samples in each genre where each sample is an audio clip of 30 seconds length. This dataset can be used for future work on Indian music genre classification. Third, we have incorporated short-time and long-time features to capture the static and temporal structure of the spectrogram and have analysed both cultural forms. Fourth, we propose a novel approach of using timbre and chromagram features with Gaussian Mixture Models(GMM) based Hidden Markov Models to identify the patterns in Indian music which gave an overall accuracy of 98.0%, which is better than the previous state-of-the-art [12] which also worked with 5-genres.

2.5 Outline of Paper

In section 3 we discuss the dataset used for the experiments and the structure and characteristics that make an Indian genre distinct from other genres. Sections 4 and 5 give a detailed explanation of the features chosen and classifiers used. Section 6 discusses the experimental results. Section 7 outlines possible future work.

3. DATASET

In this section, we look at the structural differences between Indian and Western music forms and discuss the characteristic features of the Indian genres used in the experiment.

3.1 Indian Music

For the experiments we have considered five popular genres - Hindustani (Dhrupad + Thumri), Carnatic, Folk (Punjabi) and Ghazal. We constructed our own dataset because no standard dataset is available. Each genre contains 100 audio recordings each 30 seconds long extracted randomly from recordings available on-line. All samples are vocals.

3.1.1 Indian classical Music

Indian classical music is based on melody without the presence of any major harmonic structure. It has seven basic notes with five interspersed half- notes, resulting in a 12-note scale. In Indian music any frequency can be set as the base frequency, known as *swara*(note) 'Sa' with other notes taking frequency values following their fixed ratios

with respect to Sa. Indian classical music has three important characteristics- *raga* (melodic aspect of music) and *taal* (cycle of fixed number of beats repeated over and over as shown in Figure 1) and a *drone*(a sustained note). Indian popular music have the the melodic (though not *raga*) and *taal/beat* components but do not usually have the *drone*.

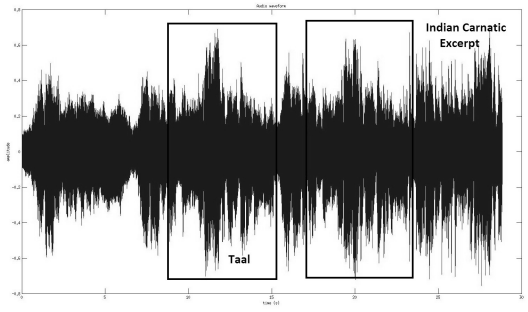


Figure 1. Repetition of beats *(taal)* in Indian Classical

Dhrupad: Dhrupad is a very old form of Hindustani music that is primarily devotional in content. It has two major parts- Alap (sung without words) and Dhrupad (fixed composition part) accompanied by Pakhawaj, a two headed barrel drum. A performance begins slowly with alap where usually the syllables of the following mantra is recited: *"Om Anant tam Taran Tarini Twam Hari Om Narayan Anant Hari Om Narayan"*. In alap an artist develops each note and uncovers the personality of the chosen raga. The tempo gradually increases and an alap ends after exploring three octaves. Then begins the dhrupad in which the artist is joined by the Pakhawaj. It is usually set to the following *talas/taal*: tala chautal (12 beats), tivra (7 beats), sulfak (10 beats), matt (9 beats), or farodast (14 beats). In the dataset samples were extracted from dhrupad performances where about 25% of them were by female artists.

Thumri: Thumri is a semi-classical form of Hindustani music which is romantic and devotional in nature. The text revolves around a girl's love for Lord Krishna. The lyrics are written in dialects of Hindi called Awadhi and Brij Bhasha. Thumri has greater flexibility and more liberties are taken with the raga as compared to Dhrupad or Khayal. The compositions are usually of 8 beats (*kaherava*), 14 beats (*dipchandi*) or of 16 beats (*addha tala*). Samples were extracted from 10 Thumri performances where 3 were by female artists.

Carnatic: Carnatic music is the classical music of Southern India. Like Hindustani music, it is based on ragas and *taals*. Performances in Carnatic music are divided into a number of sections. It starts with a *varanam*, literally meaning "a description", which unfolds the important features of the raga being performed. Then comes the fixed composition in the raga, known as *kritis*. The main composition is *alapana*, which explores the raga and uses the sounds like aa, ri, na, ta etc to elaborate the notes. This starts slowly and finally builds a complicated exposition of the raga without any rhythmic accompaniment. There are *niruval* and *kalpana* swara(notes) that provide opportunities to improvise. In the dataset, about 55% of the samples are sung by female singers.

Hindustani and Carnatic music differs mainly in the following characteristics-

- Hindustani music like Dhrupad gradually builds up tempo from very slow to very fast while in Carnatic music there is a constant and fairly fast tempo throughout.
- In Hindustani music notes are held longer and are mostly approached and released by ornaments or small ornamental phrases whereas in Carnatic music notes are not held for long and usually released by a characteristic oscillation using indeterminate pitch.
- Hindustani music has more improvisation around the *raga* unlike Carnatic which has a some what more rigid structure and is composition based.

3.1.2 Folk- Punjabi

Punjabi music has a smaller range which seldom extends beyond an octave. It is accompanied by a lively rhythm (*chaal*) and is usually based on *kaherava taal*, a cycle of eight beats. Greater emphasis of the music is on the lyrical nature of the songs. The characteristic which makes it different from other music forms is the use of *Dhol*- a barrel shaped drum that gives a bass sound. Samples are taken from *Folk N Duet 2010* Album contributed by 16 singers which includes 3 female singers.

3.1.3 Ghazal

Ghazal is defined as a poetic genre and consists of a series of rhymed verses, each symmetrically divided into two half-verses. The structure of a Ghazal is defined by its metre and rhyme which are maintained throughout the poem. Samples were extracted from ghazal performances where 19% of total pieces were sung by female singers.

3.2 Western Music

For western music we considered the GTZAN dataset [5]. It consists of 10 distinct genres- Blues, Classical, Country, Disco, Hip hop, Jazz, Metal, Pop, Reggae and Rock. Each genre is represented by 100 samples of 30 seconds length each. Since we have 5 genres for the Indian music, we considered only top 5 genres (Classical, Hiphop, Jazz, Pop and Reggae) from the GTZAN dataset based on the results of [5].

4. FEATURE EXTRACTION

For feature extraction, we used the MIRToolbox [27], an open source toolbox for musical feature extraction. For the experimental set-up we used three types of feature sets as described below:

4.1 Feature set 1- Frequency of chromagram notes

A chromagram is a redistribution of the spectrum energy along the different pitch levels (or chromas). We have con-

sidered 12 pitch classes- C, C#, D, D#, E, F, F#, G, G#, A, A# and B. We used a framesize of 100 milli-seconds with 50% overlap. The choice of a large framesize is to cover for the semitone energies in lower frequencies. For each frame, we assigned the note that has the maximum energy in the frame's 12-dimensional chromagram vector; we call this the "dominant note". The use of the dominant note in each frame captures the melodic information as the melody note is the one that typically dominates the feature. Concatenating the dominant note in each frame gave us a note sequence representing the sample. We then formed a 12-dimensional vector of the normalized frequencies (counts) of the notes in the note sequence. This feature set is motivated by the repetition of notes in Indian music and the use of chromagrams in [6] and [7].

4.2 Feature set 2- Global features

We extract a set of global features based on the main dimensions of music which includes melody, rhythm, timbre and spatial features as described in [8] [9] and [10]. We used the following features to form a vector of 86 global features-

- **Energy features**- mean and variance of the energy envelope, Root Mean Square (RMS) energy, low-energy rate(percentage of frames having less than average energy).
- **Rhythm features**- mean and variance of notes onset time (successive bursts of energy indicating estimated positions of the notes), event density (average frequency of events, i.e., the number of note onsets per second), tempo, pulse clarity [11](strength of beats).
- **Pitch features**- mean and variance of pitch.
- **Tonality features**- 12-chromagram pitch class energies, 24- key strength major and minor (cross correlation of the chromagram), 6- dimensional tonal centroid vector from the chromagram (corresponds to projection of the chords along circles of fifths, of minor thirds, and of major thirds).
- **Timbre features**- mean and variance of attack time of notes onset, rolloff frequency, brightness (percentage of energy above 1500Hz), 13 Mel-Frequency Cepstral Coefficients (MFCCs), roughness(sensory dissonance), irregularity (degree of variation of the successive peaks of the spectrum)
- **Spectral-shape features**- zero-crossing rate, spread, centroid, skewness, kurtosis, mean and variance of Inverse Fourier Transform of logarithm of spectrum, flatness, mean and variance of spectrum, mean and variance of spectral flux, mean and variance of spectrum peaks.

These features are taken to capture the static structure of the spectrogram.

4.3 Feature set 3- Frame-wise features

We analyse each song using a framesize of 100 milli seconds with 50% overlap and extract the following features: 12-chromagram features, 13 MFCC, 13 delta- MFCC, 13 delta-delta-MFCC, 11 spectral features (zero-crossing rate, rolloff, brightness, centroid, spread, skewness, kurtosis, flatness, entropy, roughness and irregularity). For the chromagram features, we first perform a logarithm transform and then a discrete cosine transform (DCT) which essentially decorrelates the features. We use these modified chromagram features with the above mentioned features to get a vector of 62 features for each frame. A sample is now represented by a 2-D matrix of size $62 \times frames$, where *frames* are the number of frames in the sample. These features carry the temporal information of the spectrogram.

5. EXPERIMENTAL RESULTS

We performed four experiments based on the above three feature sets. For each run, we randomly divided the dataset into 80% train and 20% test.

5.1 Experiment 1: Adaboost on Feature set 1

We train One-Vs-One Adaboost classifiers. A sample is assigned the class which gets the maximum votes. In case of a tie, the class with the lower index is the predicted label. Table 1 shows the confusion matrix for 10 runs of the experiment for Indian Music and Western Music. Overall the accuracies obtained for Indian Music and Western Music are 58.70% and 46.90% respectively.

Table 1. Confusion Matrix (in %) for Exp 1

(a) Indian Music

	Ca	Dh	Gh	Pu	Th
Ca	**66.50**	9.00	5.50	6.50	12.50
Dh	10.50	**64.00**	6.00	9.00	10.50
Gh	16.50	6.50	**64.50**	7.00	5.50
Pu	18.50	12.50	14.00	**47.50**	7.50
Th	19.50	6.50	13.00	10.00	**51.00**

Legend: Ca:Carnatic, Dh:Dhrupad, Gh:Ghazal, Pu:Punjabi, Th:Thumri

(b)Western Music

	Cl	Hi	Ja	Po	Re
Cl	**60.00**	8.00	13.50	12.50	6.00
Hi	3.50	**58.50**	12.00	14.00	12.00
Ja	8.50	21.00	**48.00**	13.00	9.50
Po	9.50	18.50	22.00	**32.00**	18.00
Re	14.00	17.50	16.50	16.00	**36.00**

Legend: Cl:Classical, Hi:Hip-hop, Ja:Jazz, Po:Pop, Re:Reggae

5.2 Experiment 2: Adaboost on Feature set 2

The Adaboost classification technique is used in a manner similar to Experiment 1. The confusion matrices for Indian music and Western music are shown in Table 2. The overall accuracies obtained are: Indian music- 87.30%, Western music- 74.50%.

5.3 Experiment 3: Adaboost on Feature set 1+2

Adaboost classification technique is used similar to Experiment 1. Here we take a combination of feature set 1 and feature set 2 for the training and testing. Table 3 shows the confusion matrices. The overall accuracies achieved are: Indian music- 87.80% , Western music- 77.50%.

Table 2. Confusion Matrix (in %) for Exp 2

(a) Indian Music

	Ca	Dh	Gh	Pu	Th
Ca	**89.50**	1.50	5.00	0.50	3.50
Dh	5.00	**90.00**	2.50	0	2.50
Gh	5.50	6.50	**81.00**	1.50	5.50
Pu	3.00	0	2.00	**95.00**	0
Th	10.50	5.00	3.50	0	**81.00**

Legend: Ca:Carnatic, Dh:Dhrupad, Gh:Ghazal, Pu:Punjabi, Th:Thumri

(b)Western Music

	Cl	Hi	Ja	Po	Re
Cl	**87.50**	0	10.50	0	2.00
Hi	0.50	**69.50**	0.50	14.00	15.50
Ja	9.00	2.00	**78.50**	0	10.50
Po	1.50	10.50	3.00	**74.50**	10.50
Re	4.50	18.50	4.00	10.50	**62.50**

Legend: Cl:Classical, Hi:Hip-hop, Ja:Jazz, Po:Pop, Re:Reggae

Table 3. Confusion Matrix (in %) for Exp 3

(a) Indian Music

	Ca	Dh	Gh	Pu	Th
Ca	**90.00**	1.50	4.00	0.50	4.00
Dh	4.50	**91.00**	2.00	0	2.50
Gh	6.50	5.50	**81.50**	2.00	4.50
Pu	3.00	0	2.00	**95.00**	0
Th	7.50	3.50	7.50	0	**81.50**

Legend: Ca:Carnatic, Dh:Dhrupad, Gh:Ghazal, Pu:Punjabi, Th:Thumri

(b)Western Music

	Cl	Hi	Ja	Po	Re
Cl	**90.50**	0	9.00	0.50	0
Hi	0.50	**79.00**	1.00	12.50	7.00
Ja	10.50	2.00	**78.00**	0.50	9.00
Po	0.50	15.00	1.50	**72.50**	10.50
Re	3.50	11.50	9.50	8.00	**67.50**

Legend: Cl:Classical, Hi:Hip-hop, Ja:Jazz, Po:Pop, Re:Reggae

5.4 Experiment 4: HMM with GMM based on Feature set 3

We train Hidden Markov Models on Feature set 3 with 62 hidden states and full covariance matrix. The emissions of all states were randomly initialized and the states need not represent exactly the underlying feature. The algorithms uses maximum likelihood parameter estimation using Expectation Maximization. The state emissions were 62 dimensional chroma and timbre data modelled by Gaussian mixtures with 8 Gaussians. We used the Hidden Markov Model Toolbox of Matlab by [23] for the same. The confusion matrix for Indian music and Western music are shown in Table 4. Overall accuracies of 98.00% and 67.00% were obtained for Indian and Western music respectively.

6. DISCUSSION

Our experiments performed better on Indian music than on Western music for given classification techniques and sets of features (Figure 2). The differences in performance of the two cultural forms of music can perhaps be traced to the more well-defined structural form and strong melodic content of Indian Music. In Indian music, melody and rhythm offer a variety of cues that do not seem to be present in Western music.

Table 4. Confusion Matrix (in %) for Exp 4

(a) Indian Music

	Ca	Dh	Gh	Pu	Th
Ca	**95.00**	0	3.00	2.00	0
Dh	0	**99.00**	1.00	0	0
Gh	1.00	0	**98.50**	0.50	0
Pu	0	0.500	0	**99.50**	0
Th	1.00	0	1.00	0	**98.00**

Legend: Ca:Carnatic, Dh:Dhrupad, Gh:Ghazal, Pu:Punjabi, Th:Thumri

(b)Western Music

	Cl	Hi	Ja	Po	Re
Cl	**85.00**	0	10.00	0	5.00
Hi	0	**75.00**	5.00	15.00	5.00
Ja	25.00	15.00	**55.00**	5.00	0
Po	0	20.00	5.00	**65.00**	10.00
Re	0	15.00	0	30.00	**55.00**

Legend: Cl:Classical, Hi:Hip-hop, Ja:Jazz, Po:Pop, Re:Reggae

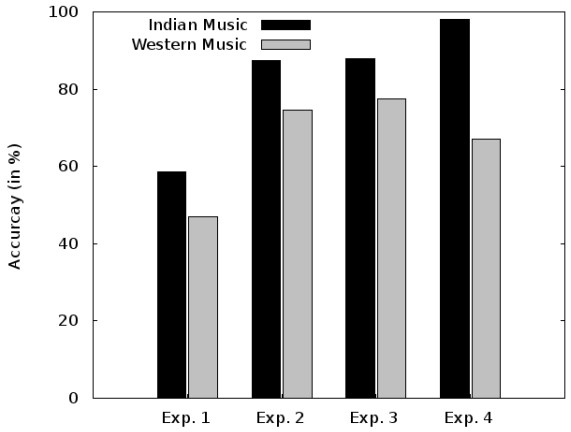


Figure 2. Comparison of performance of Indian and Western Genres on four experiments

The better performance of Indian music over Western music on similar experiments answers the second question of our research problem which suggests that the state-of-the-art features (spectral, timbre, harmony etc) used for Western music genre classification can be applied to Indian music where they show higher accuracies.

In Indian music, the classical forms (Carnatic and Dhrupad) performed better than Ghazal. This is may be because the Indian classical music is more structured as it has 3 components of *raga* (melody), *taal* (rhythm) and *drone* (sustained note) and has a strong melodic structure. In last three experiments Punjabi music has lowest confusion because its two characteristics features: lively rhythm and distinct timbre due to bass sound produced by *Dhol*; are accounted for in features used. In Western music, Classical and Jazz performed better than other genres, with Reggae performing the worst. These results are consistent with G.Tzanetakis et al. [5].

Performance on Feature set 1 (which is just based on the frequency of chromagram notes) for Indian music and

Western music is about 59% and 47% respectively. This is expected as the Indian music is based on melody, i.e. a sequence of notes played in a given order. Whereas, the Western music is more harmonic in nature, i.e. group of notes played simultaneously, which is not captured by a single dominant note in Feature set 1.

In Feature set 2 we have used the 12-chromagram pitch class energies which tries to capture the dominance order of notes. The dominance order of notes is also represented by the frequency of chromagram notes in Feature set 1. Thus, there is no significant difference in accuracies of Experiment 2 (using Feature set 2) and Experiment 3 (using Feature set 1+2).

The highest accuracy of 98.0% for Indian music genres in Experiment 4 is better than S.Jothilakshmi et al. [12] and the state-of-the-art to the best of our knowledge. The major difference in accuracies between local frame-wise features and global features in case of Indian music may be because the local frame-wise features are better in capturing the characteristics of Indian music like raga notes and *taal* (repetition of beats) which require small size windows to be analysed. An accuracy of 77.5% for Western music genres in Experiment 3 is better than G.Tzanetakis et al. [5] on the same dataset of five genres (Classical, Hiphop, Jazz, Pop and Reggae).

7. FUTURE WORK

This work can be extended in various ways: forming a 'golden-set' of features that are genre-specific like rhyme in Ghazal, beats in Folk Punjabi etc.; recognition of patterns like *taal* in Indian music and *chords* in Western music; expansion of classes in terms of genres and sub-genres so that we can work with more classes in both datasets (GTZAN has 10-genres); studying music forms of other cultures for example Chinese and Japanese and comparing them with Indian and Western genres.

8. REFERENCES

[1] J. J. Aucouturier, F. Pachet. "Representing musical genre: A state of the art" *Journal of New Music Research*, pp. 83-93, 2003.

[2] E. Benetos, C. Kotropoulos "A tensor-based approach for automatic music genre classification" *Proc. EUSIPCO,2008*

[3] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, B. Kegl "Aggregate features and Adaboost for music classification" *Machine Learning* vol. 65, no. 2-3, pp. 473-484, 2006

[4] S. Sukittanon, L. E. Atlas,J. W Pitton "Modulation-scale analysis for content identification" *IEEE Trans. Signal Processing* vol. 52, no. 10, pp 3023-3035, Oct. 2004

[5] George Tzanetakis and Perry Cook "Music Genre Classification of Audio Signals" *IEEE Transactions on Speech and Audio Processing* vol. 10, no. 5, pp. 293-302, 2002.

[6] D. P. W. Ellis and G. E. Poliner. "Identifying cover songs with chroma features and dynamic programming beat tracking" *ICASSP* Hawaii, USA 2007.

[7] S. Kim and S. Narayanan "Dynamic chroma feature vectors with applications to cover song identification" *In MMSP* Cairns, Australia, 2008.

[8] N. Scaringella, G. Zoia, and D. Mlynek, "Automatic Genre Classification of Music Content: A Survey" *IEEE Signal Processing Magazine* 2006, 23(2), pp. 133141.

[9] Kaichun K. Chang, Jyh-Shing Roger Jang, and Costas S. Iliopoulos. "Music Genre Classification via Compressive Sampling" *11th International Society for Music Information Retrieval Conference (ISMIR 2010)* 2010, pp. 387-392.

[10] Thibault Langlois, and G. Marques "A Music Classification Method Based on Timbral Features" *10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, 2009, pp. 81-86.

[11] Olivier Lartillot, Tuomas Eerola, Petri Toiviainen, Jose Fornari "Multi-feature modelling of pulse clarity: Design, validation, and optimization" *International Conference on Music Information Retrieval* Philadelphia, 2008

[12] S.Jothilakshmi,N.Kathiresan "Automatic Music Genre Classification for Indian Music" *ICSCA 2012*

[13] P. Chordia and A. Rae "Raag recognition using pitch-class and pitch-class dyad distributions" *Proc. of ISMIR, 2007* pp.431436.

[14] A.Vidwans, K.K. Ganguli and Preeti Rao "Classification of Indian Classical Vocal Styles from Melodic Contours" *Proc. of the 2nd CompMusic Workshop*, Istanbu, Turkey, July 12-13, 2012

[15] J. Martens Lippens, S. and T. De Mulder "A comparison of human and automatic musical genre classification" *In IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 233236, 2004

[16] D. Perrot and R. R. Gjerdigen. "Scanning the dial: an exploration of factors in the identification of musical style" *In Proceedings of the 1999 Society for Music Perception and Cognition*, 1999.

[17] M. Ogihara Li, T. and Q. Li. "A comparative study on content-based music genre classification" *In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 282289, 2003.

[18] West, K. and S. Cox, "Features and classifiers for the automatic classification of musical audio signals" *Proc. 5th International Conference on Music Information Retrieval (ISMIR)*, 2004

[19] Michael I Mandel and Daniel P.W.Ellis. "Song-level features and support vector machines for music classification" *In International Society for Music Information Retrieval*, 2005.

[20] A. Flexer Pampalk, E. and G. Widmer "Improvements of audio-based music similarity and genre classification" *In Crawford and Sandler*, 2005.

[21] www.magnatune.com (Access date: 5 May 2013)

[22] www.ee.columbia.edu/ dpwe/research/musicsim/ (Access date: 5 May 2013)

[23] http://www.cs.ubc.ca/ murphyk/Software/HMM/hmm.html (Access date: 5 May 2013)

[24] C. Marques, I. R. Guilherme, R. Y. M. Nakamura and J. P. Papa "New trends in musical genre classification using optimum path forest" *ISMIR*, 2011.

[25] Yannis Panagakis, Constantine Kotropoulos, Gonzalo R. Arce "Music genre classification via sparse representations of Auditory Temporal Modulations" *EUSIPCO*,2009.

[26] Justin Salamon, Bruno Rochay and Emilia Gomez "Music genre classification using melody features extracted from polyphonic music signals" *ICASSP*,2012.

[27] Olivier Lartillot, Petri Toiviainen "A Matlab Toolbox for Musical Feature Extraction From Audio" *International Conference on Digital Audio Effects*, Bordeaux, 2007

[28] "Raga" http://www.britannica.com/EBchecked/topic/489518 (Access date: 5 May 2013)

SWARA HISTOGRAM BASED STRUCTURAL ANALYSIS AND IDENTIFICATION OF INDIAN CLASSICAL RAGAS

Pranay Dighe[1], Harish Karnick[1], Bhiksha Raj[2]
1. Indian Institute of Technology, Kanpur, India. 2. Carnegie Mellon University, Pittsburgh PA, USA
pranay.dighe@idiap.ch, hk@cse.iitk.ac.in, bhiksha@cs.cmu.edu

ABSTRACT

This work is an attempt towards robust automated analysis of Indian classical ragas through machine learning and signal processing tools and techniques. Indian classical music has a definite heirarchical structure where macro level concepts like thaats and raga are defined in terms of micro entities like swaras and shrutis. *Swaras or notes* in Indian music are defined only in terms of their relation to one another (akin to the *movable do-re-mi-fa* system), and an inference must be made from patterns of sounds, rather than their absolute frequency structure. We have developed methods to perform scale-independent raga identification using a random forest classifier on swara histograms and achieved state-of-the-art results for the same. The approach is robust as it directly works on partly noisy raga recordings from *Youtube* videos without knowledge of the scale used, whereas previous work in this direction often use audios generated in a controlled environment with the desired scale. The current work demonstrates the approach for 8 ragas namely Darbari, Khamaj, Malhar, Sohini, Bahar, Basant, Bhairavi and Yaman and we have achieved an average identification accuracy of 94.28% through the framework.

1. INTRODUCTION

Music information retrieval(MIR) is an active and growing field of research as most music today whether available over the internet or otherwise is in digital form. The important feature of classical music, both western and Indian, that distinguishes it from other kinds of music is that it is supported by a proper well established theory and rules. There has been a lot of work on content analysis of western classical music in terms of information retrieval, genre detection and instrument/singer identification etc. for example (see [1], [3], [2]). Though Indian Classical music is also a major form of music, the current literature related to it is very limited, in comparison to its western counterpart. Indian classical music is known for its technical soundness and its well defined structure. A basic musical performance unit, akin to a song, is a *raga*. A *raga* has layers of additional finer structure that can help in content identification and classification, yet these aspects have been under-utilised by the music analysis community.

While an expert in Indian Classical music can identify a raga just by noticing the unique constituent patterns such as *swaras (notes), arohan, avarohan* and *pakad* in the performance (explained later), developing computational models for the same has been a challenging task for music researchers. The freedom that Indian classical music provides to an artist to give his/her own personal flavour to a raga makes it harder for a novice to identify two different performances of the same raga. Another key challenge is the fact that swaras are defined only in terms of their relation to one another, and inferences must be made from patterns of sounds, rather than their absolute frequency structure. This motivated us to move away from diretly trying to identify notes as features and conceptualize various features based on swaras and their structural form.

2. APPLICATIONS

Automatic Tagging/Annotation: Continued digitisation of old archives of classical music and the sole availability of newer classical performances in digital form has resulted in huge digital databases of music. Automatic content tagging of this unorganised media is important to generate metadata for available data and thus facilitate creation of easily accessible databases. Bertin-Mehiuex et al. have already created a tool [7] for music tagging, but the work does not include Indian ragas.

Music Recommendation System: Based on an user's earlier choices of music, a music recommendation system filters out and recommends music through tags or content analysis. In the Indian system ragas are associated with emotions, time of the day, and seasons of the year. For queries based on such criteria, a proper recommendation system can make appropriate choices using raga identification as a base.

Music Tutoring/Correctness Detection System: A probabilistic raga identification system can be modified to a tutoring [9] or correctness detection system as well. While achieving this for a complex professional musical performance might be difficult, yet using it for cleaner and simpler instrumental versions of a raga may give positive results. Mistakes like skipping some swaras or a particular rule not being followed, can be identified by such a tutoring system.

Raga Generation: Within some broad rules, Indian classical music allows a performer to modify the components of ragas according to his creativity to create his own personalised performance of that raga. Generative models induced from human raga performances can be used to synthesize new raga performances that can be made unique or personalized by injecting controlled randomness or variations.

3. RELATED WORK

For a novice human ear, the basic way of recognising a raga is to correlate two tunes on the basis of how similar they sound. A trained expert on the other hand looks for characteristic phrases like arohanam, avrohanam, pakad and constituent swaras, gamakas etc. to arrive at a conclusion about the raga. This well defined manner of raga identification using the above properties has motivated researchers to conceptualize computational models for them.

Sahasrabuddhe and Upadhye [10] (1992) modelled a raga as a finite state automaton based on the *swara* constituent sequential patterns. Pandey et al. (2003) [11] extended this idea of *swara* sequence in the "Tansen" raga recognition system where they worked with Hidden Markov Models on *swara* sequences. These swara sequences were extracted using two methods- hill peak heuristic and note duration heuristic. They also employed two separate *pakad* matching algorithms that improved the HMM based results. The first algorithm used substring matching for pakad identification and the second algorithm was based on counting occurences of n-grams of frequencies in the pakad. Tansen was able to perform with an accuracy of 87% on a dataset of two ragas.

In [13] Sreedhar and Geetha created a database of ragas and used the scale of the raga performance as the similarity metric. Within a scale, notes are matched with the existing sets of notes in the ragas in the database. The closest raga in the database is given as the output for the test raga. Chrodia and Rae [12](2007) derived Pitch-class Distributions (PCD) and Pitch-class Dyad Distributions (PCDD) from Harmonic Pitch Class Profiles (HPCP) and used these distributions for classification using SVMs. They achieved state-of-the-art results with accuracies of 78% for PCDs and 97.1% for PCDDs. The dataset they had used consisted of 17 ragas played by a single artist.

Inspired by the use of HMMs over swara sequences and PCDs and PCDDs in [11] and [12], we propose a new approach to raga identification using swara based features extracted from chromagrams. Similar to HPCP features, we extract feature vectors from chromagram patterns (details later). But instead of learning probability distributions and using their parameters for an SVM (support vector machine) based classification as in [12], we modify them to extract Swara based features that are then used for raga modelling and identification.

The novelty in our approach is in performing raga identification without the knowledge of the scale of the performance. We employ *swara* based features extracted from the chromagram using the concept of *vadi* (explained later) and perform random forest classifier based classification to achieve state-of-the-art results.

4. INDIAN CLASSICAL MUSIC: BASIC STRUCTURAL ELEMENTS

The theory of Indian classical music discussed in this chapter is based on the texts [8] of well known Indian musicologist and scholar Vishnu Narayan Bhatkhande. We first discuss some important concepts of Indian classical music that are needed to understand the methods used in this work.

4.1 Swaras

Swaras (or notes) correspond to the frequencies being performed vocally or by a musical instrument. Collectively the seven Swaras are symbols used for a set of frequency values. They are the constituent units of a raga and thus act as an alphabet. They are closely related to the solfege (do re mi fa so la ti) in western music. Basically, there are 7 *swaras* in Indian classical music: *Shadja (Sa), Rishab (Re), Gandhara (Ga), Madhyama (Ma), Panchama (Pa), Dhaivata (Dh), and Nishad (Ni)*. These *swaras* are related to each other by the fixed ratio of absolute frequency values they denote. Similar to notes in western music, we get the same *swara* one octave above or below a particular *swara*.

The notes Sa, Re, Ga, Ma, Pa, Dha and Ni are today known as "shuddha svaras" or "pure notes". This expression is used in contrast to "vikrta svaras" or "altered notes". Thus Re may be flattened to get the vikrta swara "Komal Re" and Ma can be "sharpened" to get the vikrata swara "tivra" Ma. *Sa* and *Pa* only have the *Shuddha*(pure) form, while the rest have variants in *Tivra*(sharp) or *Komal*(soft) forms. Table 1 describes relations between the various swaras and similarity to western notes if C is labelled as Sa.

Table 1. Scale of 12 Swaras used in Ragas (if the note C is labelled as the swara Sa)

Hindustani Name (Symbol)	Solfa	Scale of C	Ratio to Sa
Shadja (Sa)	Doh	C	1
Komal Rishabh (Re)		C#,Db	256/243
Shuddha Rishabh (Re)	Re	D	9/8
Komal Gandhr (Ga)		D#,Eb	32/27
Shuddha Gandhr (Ga)	Mi	E	5/4
Shuddha Madhyam (Ma)	Fa	F	4/3
Tvra Madhyam (M)		F#,Gb	45/32
Pancham (Pa)	Sol	G	3/2
Komal Dhaivat (Dha)		G#,Ab	128/81
Shuddha Dhaivat (Dha)	La	A	5/3
Komal Nishd (Ni)		A#,Bb	16/9
Shuddha Nishd (Ni)	Ti	B	15/8
Shadja (Sa)	Doh	C'	2

The swaras are related to each other through frequency value ratios. If the swara Sa is assigned some frequency value in hertz, the rest of the *swaras* are spread around *Sa* as per the ratios governing them. For example the *swara Pa* is always $\frac{3}{2}$ times *Sa* in terms of actual frequencies. The 12-note system which is mostly used with *Sa* as the 1^{st} note and *Ni* as 12^{th} is described in table 4.1 in terms of the frequency ratios.

4.2 Raga

A raga (Sanskrit meaning - "color/hue") is a complex melodic construction over swaras meant to induce a specific emotional response i.e. color the mind of the listener with a particular emotion.

A raga is not just a simple collection of *swaras*, but is typically identified by swara patterns, the presence or absence of certain *swaras*, the possible ornamentation (*alankars*) applied on these swaras and their relative importance while playing the raga.

Here we discuss some relevant raga related terminology.

4.2.1 Arohan, Avrohan and Pakad

Though two ragas may have the same constituent *swaras*, a unique differentiation between them is the ascending and descending sequences of *swaras* termed as *arohan* and *avarohan* respectively. *Pakad* is a small sequence of *swaras* in a raga that acts as a signature for the raga and an artist often visits and revisits the *pakad* over a performance.

4.2.2 Vadi and Samvadi

Vadi is the most prominent *swara* in a raga and is often described as the King of *swaras* for that raga. The *swara* second to *vadi* in importance is called the *samvadi*. An artist stays at the *vadi* and *samvadi* for significant durations and emphasises them in a performance. The *swaras* other than *vadi* and *samvadi* which constitute the raga are called *anuvadi swaras* whereas the *swaras* which are totally absent are called *vivadi*.

4.2.3 Scale of the Raga - Tonic Frequency

An artist always tunes his whole raga performance around a fixed tonic frequency which she chooses according to her own comfort. This frequency defines the scale in which the raga will be performed. Irrespective of the absolute value of this tonic frequency, it is termed as the *swara* Sa. The rest of the swaras get aligned in accordance with their ratios with the swara Sa.

4.2.4 Jati- Audhav/Shadav/Sampoorna

Ragas are classified based on the number of swaras in the arohan and avrohan. Sampoorna is all 7 swaras, shadhav is 6 swaras, audhav is 5 swaras and Surtar is 4 swaras. So, an audhav-audhav raga has 5 swaras in its arohan and 5 swaras in its avrohan.

5. FEATURE EXTRACTION

5.1 Chromagram

A chromagram [6] is a visual representation of energies in the 12 semitones(or chromas) of the western musical octave namely C, C#, D, D#, E, F, F#, G, G#, A, A# and B. So, it basically depicts the distribution of energy in the twelve pitch classes.

The western semitones are such that they are fixed with respect to absolute frequency values and the musical octave has the property that the semitone one octave below or above is equivalent to the current semitone. For example,

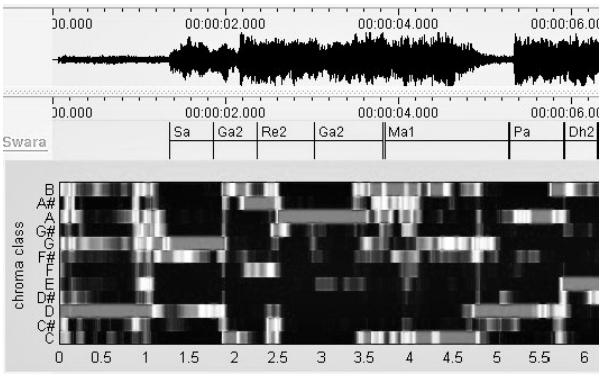


Figure 1. *Chromagram and Swara sequence for an arohan of Raga Bhairavi*

if one note has a frequency of 440 Hz, the note an octave above it is at 880 Hz, and the note an octave below is at 220 Hz. Since the semitones get repeated in each octave above and below, the energies in the chromagram for each semitone(chroma) is computed by wrapping and adding it up over different octaves.

Figure 1 depicts a chromagram generated from an *arohan* of raga Bhairavi. The arohan for raga bhairavi is:

Sa Re-Kom Ga-Kom Ma Pa Dh-Kom Ni-Kom Sa

The *Sa swara* of *Bhairavi* coincides with the semitone G and the rest of the *swaras* in the *arohan* get aligned with the semitone pattern in the chromagram. Note that, though this notes-to-swara alignment is not perfect and energies may spill over to the adjacent bins, still a signature pattern very close to the Bhairavi arohan is visible. This alignment is deduced using the swara and western seminotes equivalence discussed above and in Table 4.1.

These observations and previous use of chromagrams in audio analysis [14] and chord recognition [15] motivated us to use the chromagram to extract information about swaras. For processing chromagrams, the MIRToolbox [16], an open source toolbox for musical feature extraction was used to extract features.

5.2 Swara Histograms

From the chromagram, we extract the semitone with maximum energy in each frame and get a sequence of semitones for the raga. Though these sequences of semitones might have some identifying information about the raga, using them for raga identification is not appropriate since ragas are defined over patterns of *swaras* that in turn may get associated with different semitones depending on the tonic frequency selected by the artist for a particular performance.

In our approach, we assume that we do not have information about the tonic frequency of the raga performance. We must therefore find the mapping from the absolute frequency scale employed by the chromagram to the relative scale of the musical piece, so that the *swara* sequence can be identified. To do so, we use the concept of *vadi* discussed earlier to convert the semitone sequence to a *swara* sequence. We compute the most frequently occur-

ring semitone from the semitone sequence and associate it with the *vadi* of a raga which is known for each raga. For example *vadi* for raga Khamaj is *swara* "Ga". In an audio of Khamaj, if the semitone C# is most prominent, we label *swara* "Ga" at semitone C# and then convert the rest of the semitone sequence into a *swara* sequence.

The above procedure is raga specific, *i.e.* the conversion from semitone sequence to *swara* sequence utilizes the identity of the raga-specific *vadi* swara. Assume that we are building a system for n ragas and the actual raga for the test audio is not known, then for the given audio, we must compute separate *swara* transcriptions for each of the n ragas. They are combined into an unified representation in the algorithm below. The concatenation of 12-dimensional normalized frequency vectors $N_i's$ is done so as to capture the underlying raga's behavior even in the cases when it is being transcripted using a vadi of some other raga.

Algorithm 1 ***Computation of Swara based Features***

Step 1: From a raga audio, extract 3 minutes long snippets.

Step 2: Compute the snippet's western semitone transcription T from the chromagram by assigning each frame the highest energy semitone.

Step 3: Find the most frequently occuring semitone s in transcription T.

Step 4: Compute the snippet's *swara* transcription S_i for each raga R_i where $i = 1..n$ by matching semitone s to vadi v_i of raga R_i.

Step 5: From each transcription S_i, create a 12-dimensional normalized frequency vector N_i, that is a histogram of *swaras* over the 3 minute snippet.

Step 6: Concatenate all such N_i's to get k-dimensional feature vector where $k = 12 \times n$. We call this feature the **swara histogram** and use classification trees to identify the raga.

6. EXPERIMENTS

6.1 Raga Dataset

There does not exist any standard database for raga identification yet and the current dataset has been collected from YouTube videos. We include both vocal and instrumental performances by artists. In terms of quality, the relatively newer raga performances whose digital recordings are available have cleaner audio on YouTube. The older recordings do not have very clear vocal or instrument sounds. We believe that the methods used here should be unaffected by these. We demonstrate our system on 8 ragas whose details are in Table 2.

Raga	Number of Recordings	Time (in minutes)
Bahar	16	109
Basant	13	125
Bhairavi	18	162
Darbari	14	125
Khamaj	13	91
Malhar	13	124
Sohini	14	121
Yaman	16	151

Table 2. *Details of the Raga Dataset Used*

6.2 Random Forest Classifier over Swara Histograms

Data: Swara histograms were created for raga audios using Algortihm 1. For each snippet of 3 minutes, a hop factor of 0.025 seconds was used which gave 7200 frames. Each frame was tagged with a swara. We created a histogram of the 12 swaras over these 7200 frames and then normalised it to get swara frequencies that added up to 1. As explained earlier, for each snippet, we got 8 swara transcriptions corresponding to each possible raga. After concatenation of histograms from each such transcription, we finally got a 96 dimensional histogram feature for the snippet.

Random Forest Configuration: An ensemble of 100 trees was grown using the random forest algorithm [18]. For each tree, if there are N datapoints in the data, we choose N datapoints from them with repetition. This is called a bootstrap sample. Within each tree, when splitting criteria for a decision at a node are being calculated, a fixed number of dimensions are randomly sampled from the 96 dimensions. This number has to be less than the total number of dimensions and the best split on these dimensions is used to split the node. We take the square root of the number of dimensions i.e. $\lceil\sqrt{96}\rceil = 10$ for this.

7. RESULTS AND ANALYSIS

7.1 Results for Random Forest Classifier Experimrent

A 10-fold cross validated experiment has been done and the average accuracy attained for the 8 ragas is $\mathbf{94.28\%}$. Apart from Khamaj, all ragas have been identified with an accuracy of more than 90%.

Raga	Da	Kh	Ma	So	Ba	Bh	Bs	Ya
Da	**95.96**	0.83	0.77	0.00	0.00	0.00	0.83	1.60
Kh	1.11	**88.00**	2.22	0.00	0.00	2.11	0.00	6.56
Ma	0.77	0.00	**91.86**	0.00	8.00	4.17	0.00	3.21
So	0.00	0.00	0.00	**96.67**	0.00	1.11	0.00	2.22
Ba	0.00	0.91	0.91	0.00	**96.36**	0.00	0.00	1.82
Bh	4.23	0.00	0.00	0.00	0.59	**94.56**	0.00	1.82
Ba	0.00	0.00	0.00	1.11	1.11	1.11	**95.56**	1.11
Ya	0.67	2.67	0.67	0.00	0.00	1.33	0.00	**94.67**

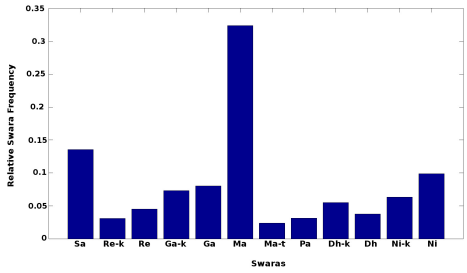
Table 3. *Confusion Matrix for the 8 ragas*

We make some important observations here:

1. Raga Malhar is being confused as Bahar and Bhairavi with 8% and 4% misclassifications respectively. The fact that all three of them have the same vadi (Ma) and samvadi (Sa) is the likely reason this happens.
2. Similarly, Khamaj and Yaman where 6.5% of Khamaj test cases are misclassified. Again, both have the same vadi (Ga) and samvadi (Ni) pair.
3. Raga Sohini is the only raga among the three which employs 6 *swaras*) (a class of ragas called *shadav*) whereas the rest are *sampoorna* and have 7 notes. This is a possible reason for its superior performance.

7.2 Plots for Swara Histograms

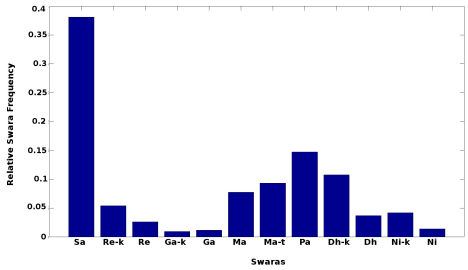
The swara histogram features have been plotted for all the 8 ragas and they reveal exact correspondence of our swara



(a) Swara Histogram for Bahar

Swara	Actual	Plot
Vadi-Sam	Ma-Sa	Ma-Sa
Anuvadi	Re,Ga-k,Pa,Dh,Ni-k	Ni,Ga,Ga-k,Ni-k,Dh-k,Re
Vivadi	Re-k,Ga,Ma-t,Dh-k	Ma-t,Re-k

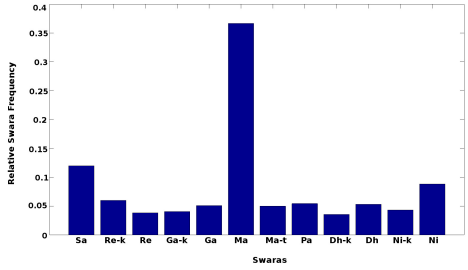
(b) Plot Observations vs. Actual Swaras in Bahar



(c) Swara Histogram for Basant

Swara	Actual	Plot
Vadi-Sam	Sa-Pa	Sa-Pa
Anuvadi	Re-k,Ga,Ma-t,Pa,Dh-k,Ni	Dh-k,Ma-t,Ma,Re-k
Vivadi	Re,Ga-k,Ma,Dh,Ni-k	Re,Ga-k,Ga,Dh,Ni-k,Ni

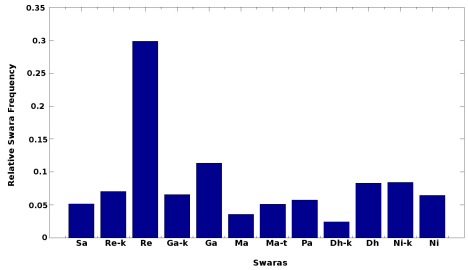
(d) Plot Observations vs. Actual Swaras in Basant



(e) Swara Histogram for Bhairavi

Swara	Actual	Plot
Vadi-Sam	Ma-Sa	Ma-Sa
Anuvadi	Re-k,Ga-k,Pa,Dh-k,Ni-k	Ni,Re-k,Pa,Ga,Ga-k
Vivadi	Re,Ga,Ma-t,Dh,Ni	ReDh-k

(f) Plot Observations vs. Actual Swaras in Bhairavi

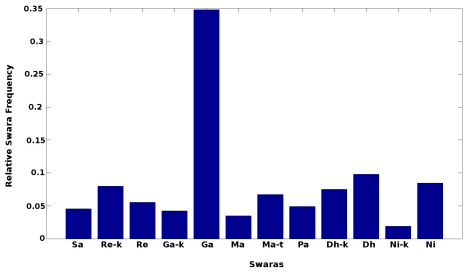


(g) Swara Histogram for Darbari

Swara	Actual	Plot
Vadi-Sam	Re-Pa	Re-Ga
Anuvadi	Sa,Ga-k,Ma,Dh-k,Ni	Re-k,Dh,Ni-k,Ni,Ga-k,Pa,Ma-t,Sa
Vivadi	Re-k,Ga,Ma-t,Dh,Ni	Dh-k,Ma

(h) Plot Observations vs. Actual Swaras in Darbari

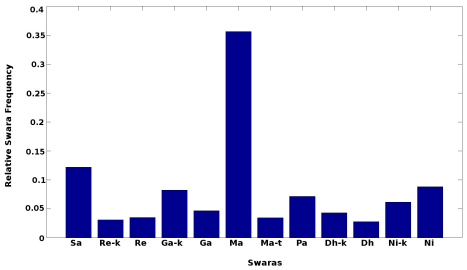
Figure 2. *Swara Histogram Plot for Bahar, Basant, Bhairavi and Darbari*



(a) Swara Histogram for Khamaj

Swara	Actual	Plot
Vadi-Sam	Ga-Ni	Ga-Dh
Anuvadi	Sa,Re,Ma,Pa,Dh,Ni-k	Ni,Dh,Re-k,Dh-k,Ma-t,Re,Pa,Sa
Vivadi	Re-k,Ga-k,Ma-t,Dh-k	Ni-k,Ma

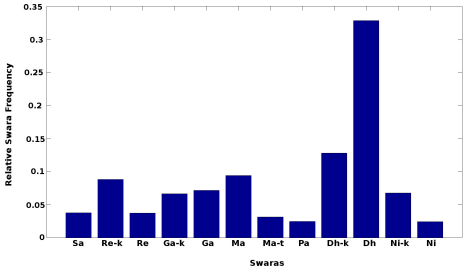
(b) Plot Observations vs. Actual Swaras in Khamaj



(c) Swara Histogram for Malhar

Swara	Actual	Plot
Vadi-Sam	Ma-Sa	Ma-Sa
Anuvadi	Re,Ga,Pa,Dh,Ni-k,Ni	Ga-k,Ni,Ni-k,Pa,Ga
Vivadi	Re-k,Ga-k,Ma-t,Dh-k	Re-k,Re,Ma-t,Dh

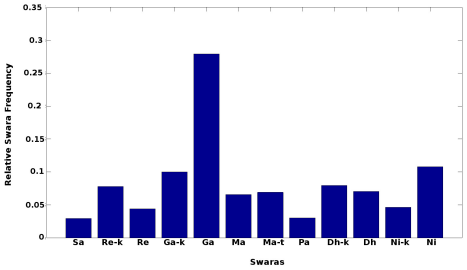
(d) Plot Observations vs. Actual Swaras in Malhar



(e) Swara Histogram for Sohini

Swara	Actual	Plot
Vadi-Sam	Dh-Ga	Dh-Ma
Anuvadi	Sa,Re-k,Ma-t,Ni	Ma,Dh-k,Re-k,Ga,Ga-k,Ni-k
Vivadi	Re,Ga-k,Ma,Dh-k,Pa,Ni-k	Sa,Re,Ma-t,Pa,Ni

(f) Plot Observations vs. Actual Swaras in Sohini



(g) Swara Histogram for Yaman

Swara	Actual	Plot
Vadi-Sam	Ga-Ni	Ga-Ni
Anuvadi	Sa,Re,Ma-t,Pa,Dh	Ga-k,Ma-t,Dh-k,Dh,Re-k
Vivadi	Re-k,Ga-k,Ma,Dh-k,Ni-k	Sa,Re,Pa,Ni-k

(h) Plot Observations vs. Actual Swaras in Yaman

Figure 3. *Swara Histogram Plot for Darbari, Malhar, Sohini and Yaman*

extraction approach to the musical theory of vadi, samvadi, anuvadi and vivadi. The plots are bar graphs of the swaras' relative frequency with respect to each other for each graph. The plots also strongly suggest the use of classification methods based on swara histograms. We have compared the observations we made from the plots with what the actual theory says about that raga in the tables below the histogram plots.

We note that while creating swara histograms, we had forcefully aligned the most frequent note with the vaadi. So, in all the plots, the vadi suggested by the plot matches with the actual vadi of the raga. What is important is that in most ragas, there is significant resemblence between the plot observations and the actual samvadi, anuvadi and vivadi. The discrepancies can be again attributed to the fact that actual raga performances are much more complex and contain variations that are not present in the written form of the raga. Since these discrepancies are present in the training as well as test data, it should have only a small impact on the results.

8. CONCLUSION AND FUTURE WORK

Our approach to raga identification relies heavily on the theoretical base for Indian classical music. The whole work confirms the premise that Indian classical music has a very well defined structure. The whole concept of a complex raga being built out of small substructures of arohan, avrohan and pakad which in turn are made up of swaras can be efficiently modelled efficiently if treated and analysed using these hierarchies. Breaking the ragas into swaras and successful identification can help to develop highly accurate applications for automatic tagging, raga tutoring, music recommendation, and possibly, raga generation.

We achieved an average accuracy of 94.28% which is the best current result for scale independent raga identification beating the previous best by Chordia [12]. We have achieved good accuracies in an experiment which excludes common audio features like MFCC, supports our initial hypothesis that the finer swara based substructure contains the defining information about ragas.

Most earlier approaches use raga audios which are performed with a specific tonic frequency by artists in controlled environments usually using only instrumental music. In comparison, we have attempted raga recognition without the knowledge of the tonic frequency, on performances downloaded from YouTube videos that have a good mixture of vocal (male and female) as well as instrumental music.

Competing with expert human identification is the final goal in raga identification and it can always be assumed that professional artists can perform this task with 100% accuracy. So, there is still scope for improvement in the results. Future work also lies in fusing the above approaches to achieve more efficiency and expanding our dataset to many more ragas. We also wish to investigate structure discovery through minimum entropy or maximum-structure learning methods. Chroma and swara features can also be tested for classification of genres and *taals* (rhythms).

9. REFERENCES

[1] Martin, Keith D., Eric D. Scheirer, and Barry L. Vercoe. "Music content analysis through models of audition." Proc. ACM Multimedia Workshop on Content Processing of Music for Multimedia Applications. 1998.

[2] Tzanetakis, G.; Cook, P., "Musical genre classification of audio signals," Speech and Audio Processing, IEEE Transactions on , vol.10, no.5, pp.293,302, Jul 2002.

[3] G. Poliner, D. Ellis, A. Ehmann, E. Gmez, S. Streich, B. Ong, "Melody Transcription from Music Audio: Approaches and Evaluation," IEEE Transactions on Audio, Speech, Language Processing 2006.

[4] Phon-Amnuaisuk, S and Keh-Siong Chee, "Interactivities in music intelligent tutoring system", In proc. of Fifth IEEE International Conference on Advanced Learning Technologies, 2005.

[5] Schulze, W. "Music Generation with Markov Models", In IEEE MultiMedia, March 2011, Vol 18 ,Issue 3, Page 78 - 85.

[6] Dan Ellis, "Chroma Feature Analysis and Synthesis", Resources of Laboratory for the Recognition and Organization of Speech and Audio-LabROSA,

[7] T. Bertin-Mahieux, D. Eck and M. Mandel, Automatic tagging of audio: The state-of-the-art. In Wenwu Wang, editor, Machine Audition: Principles, Algorithms and Systems. IGI Publishing, 2010.

[8] Vishnu Bhatkhande, "A comparative Study of the Music Systems of the 15th, 16th, 17th and 18th Centuries".

[9] S. Belle, R. Joshi, and P. Rao, Raga Identification by using Swara Intonation, Journal of ITC Sangeet Research Academy, 2009, vol. 23.

[10] H. Sahasrabuddhe and R. Upadhye, "On the computational model of raag music of india." in Workshop on AI and Music: European Conference on AI, 1992.

[11] G. Pandey, C. Mishra, and P. Ipe, "Tansen: A system for automatic raga identification" in Proc. of Indian International Conference on Artificial Intelligence , 2003, pp. 1350-1363.

[12] P. Chordia and A. Rae, "Raag recognition using pitch-class and pitch-class dyad distributions," in Proc. of ISMIR, 2007, pp. 431-436.

[13] R. Sridhar and T. Geetha, "Raga identification of carnatic music for music information retrieval," International Journal of Recent trends in Engineering, vol. 1, no. 1, 2009, pp. 571-574.

[14] Xiaoqing Yu; Jing Zhang; Junwei Liu; Wanggen Wan; Wei Yang, "An audio retrieval method based on chromagram and distance metrics," International Conference on Audio Language and Image Processing (ICALIP), Nov. 2010.

[15] K.Lee, "Automatic Chord Recognition from Audio Using Enhanced Pitch Class profile," in Proc. of the International Computer Music Conference, 2006.

[16] Olivier Lartillot, Petri Toiviainen, "A Matlab Toolbox for Musical Feature Extraction From Audio", International Conference on Digital Audio Effects, Bordeaux, 2007.

[17] G. Koduri, S. Gulati and P. Rao,"A Survey Of Raaga Recognition Techniques And Improvements To The State-Of-The-Art," Sound and Music Computing, 2011.

[18] Leo Breiman and E. Schapire, "Random forests", Machine Learning,2001,pages 5–32.

COMBINING MODELING OF SINGING VOICE AND BACKGROUND MUSIC FOR AUTOMATIC SEPARATION OF MUSICAL MIXTURES

Zafar Rafii[1], François G. Germain[2], Dennis L. Sun[2,3], and Gautham J. Mysore[4]
[1]Northwestern University, Department of Electrical Engineering & Computer Science
[2]Stanford University, Center for Computer Research in Music and Acoustics
[3]Stanford University, Department of Statistics
[4]Adobe Research
zafarrafii@u.northwestern.edu, {fgermain,dlsun}@stanford.edu, gmysore@adobe.com

ABSTRACT

Musical mixtures can be modeled as being composed of two characteristic sources: singing voice and background music. Many music/voice separation techniques tend to focus on modeling one source; the residual is then used to explain the other source. In such cases, separation performance is often unsatisfactory for the source that has not been explicitly modeled. In this work, we propose to combine a method that explicitly models singing voice with a method that explicitly models background music, to address separation performance from the point of view of both sources. One method learns a singer-independent model of voice from singing examples using a Non-negative Matrix Factorization (NMF) based technique, while the other method derives a model of music by identifying and extracting repeating patterns using a similarity matrix and a median filter. Since the model of voice is singer-independent and the model of music does not require training data, the proposed method does not require training data from a user, once deployed. Evaluation on a data set of 1,000 song clips showed that combining modeling of both sources can improve separation performance, when compared with modeling only one of the sources, and also compared with two other state-of-the-art methods.

1. INTRODUCTION

The ability to separate a musical mixture into singing voice and background music can be useful for many applications, e.g., query-by-humming, karaoke, audio remixing, etc. Existing methods for music/voice separation typically focus on estimating either the background music, e.g., by training a model for the accompaniment from the non-vocal segments, or the singing voice, e.g., by identifying the predominant pitch contour from the vocal segments.

Some methods estimate the background music by training a model on the non-vocal segments in the mixture, identified manually or using trained vocal/non-vocal classifiers. Ozerov et al. used Bayesian models to train a model for the background music from the non-vocal segments, which they then used to train a model for the singing voice [7]. Han et al. used Probabilistic Latent Component Analysis (PLCA) to also train a model for the background music, which they then used to estimate the singing voice [2].

Other methods estimate the background music directly, without prior vocal/non-vocal segmentation, by assuming the background to be repeating and the foreground (i.e., the singing voice) non-repeating. Rafii et al. used a beat spectrum to identify the periodically repeating patterns in the mixture, followed by median filtering the spectrogram of the mixture at period rate to estimate the background music [9]. Liutkus et al. used a beat spectrogram to further identify the varying periodically repeating patterns [6].

Other methods instead estimate the singing voice by identifying the predominant pitch contour in the mixture. Li et al. used a pitch detection algorithm on the vocal segments in the mixture to estimate the predominant pitch contour, which they then used to derive a time-frequency mask to extract the singing voice [5]. Hsu et al. also used a pitch-based method to model the singing voice, while additionally estimating the unvoiced components [3].

Other methods are based on matrix decomposition techniques. Vembu et al. used Independent Component Analysis (ICA) and Non-negative Matrix Factorization (NMF) to decompose a mixture into basic components, which they then clustered into background music and singing voice using trained classifiers such as neural networks and Support Vector Machines (SVM) [12]. Virtanen et al. used a pitch-based method to estimate the vocal segments of the singing voice, and then NMF to train a model for the background music from the remaining non-vocal segments [14].

Other methods estimate both sources concurrently. Durrieu et al. used a source-filter model to parametrize the singing voice and a NMF model to parametrize the background music, and then estimated the parameters of their models jointly using an iterative algorithm [1]. Huang et al. used Robust Principal Component Analysis (RPCA) to jointly estimate background music and singing voice, assuming the background music as a low-rank component and the singing voice as a sparse component [4].

In this work, we propose a method for modeling the

singing voice, which learns a singer-independent model of voice from singing examples using a NMF based technique. We then propose to combine this method with a method for modeling the background music, which derives a model of music by identifying and extracting repeating patterns using a similarity matrix and a median filter. Combining a method that specifically models the singing voice with a method that specifically models the background music addresses separation performance from the point of view of both sources.

The rest of the article is organized as follows. In Section 2, we present a method for modeling singing voice. In Section 3, we review an existing method for modeling background music, and we propose combining the two methods to improve music/voice separation. In Section 4, we evaluate the method for modeling the singing voice and the combined approach on a data set of 1,000 song clips, and we compare them with the method for modeling the background music alone, and two other state-of-the-art methods. Section 5 concludes this article.

2. MODELING SINGING VOICE

In this section, we present a method for modeling the singing voice. Because singer-specific training examples are generally not available for the music/voice separation methods, models for the singing voice are typically based on *a priori* assumptions, e.g., it has a sparse time-frequency representation [4], it is accurately modeled by a source-filter model [1], or it is reasonably described by pitch [5].

Recently, universal models were proposed as a method for incorporating general training examples of a sound class for source separation when specific training examples are not available [11]. We use these ideas to model the singing voice using a *universal voice model*, learned from a corpus of singing voice examples. Since the formulation of universal voice models is based on matrix factorization methods for source separation, we begin by reviewing Nonnegative Matrix Factorization (NMF).

2.1 NMF for Source Separation

The magnitude spectrogram $\boldsymbol{X}$ is a matrix of non-negative numbers. We assume that the spectrum at time t, $\boldsymbol{X}_t$, can be approximated by a linear combination of basis vectors $\boldsymbol{w}_i$, each capturing a different aspect of the sound, e.g., different pitches, transients, etc.:

$$\boldsymbol{X}_t \approx \sum_{i=1}^{K} h_{it} \boldsymbol{w}_i$$

The collection of basis vectors $\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w}_1 & ... & \boldsymbol{w}_K \end{bmatrix}$ can be regarded as a model for that sound class, since all possible sounds are assumed to arise as linear combinations of these basis vectors. Likewise, $\boldsymbol{H} = (h_{it})$ can be regarded as the activations of the basis vectors over time. In matrix notation, this can be expressed as:

$$\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H}.$$

NMF attempts to learn $\boldsymbol{W}$ and $\boldsymbol{H}$ for a given spectrogram $\boldsymbol{X}$, i.e., it solves the optimization problem:

$$\underset{\boldsymbol{W},\,\boldsymbol{H}}{\text{minimize}} \;\; D\left(\boldsymbol{X} || \boldsymbol{W}\boldsymbol{H}\right)$$

subject to the constraints that $\boldsymbol{W}$ and $\boldsymbol{H}$ are non-negative. D is a measure of divergence between $\boldsymbol{X}$ and $\boldsymbol{W}\boldsymbol{H}$.

To use NMF to separate two sources, say, singing voice and background music:

1. Learn $\boldsymbol{W}_V$ using NMF from isolated examples of the singing voice.
2. Learn $\boldsymbol{W}_B$ using NMF from isolated examples of the background music.
3. Use NMF on the mixture spectrogram $\boldsymbol{X}$, fixing $\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_V & \boldsymbol{W}_B \end{bmatrix}$, and learning $\boldsymbol{H}_V$ and $\boldsymbol{H}_B$.
4. Estimates of the singing voice-only and background music-only spectrograms can be obtained from $\boldsymbol{W}_V\boldsymbol{H}_V$ and $\boldsymbol{W}_B\boldsymbol{H}_B$.

A more detailed description of this approach can be found in [10], although the authors use an equivalent probabilistic formulation (PLCA) instead of NMF.

Of these tasks, steps 1 and 2 pose the greatest challenge. While it may be possible to use the non-vocal segments of the background music as isolated training data of the background music, it is rare to find segments in music where the voice is isolated. Source separation is still possible in this setting where training data of only one source is available—one simply learns $\boldsymbol{W}_V$ together with $\boldsymbol{H}_V$ and $\boldsymbol{H}_B$ in step 3. This is the approach taken in [2, 7], but it requires a sufficiently accurate prior vocal/non-vocal segmentation and a sufficient amount of non-vocal segments to effectively learn a model of the background music.

2.2 Universal Voice Model

One alternative when training data of a specific singer is not available is to learn a model from a corpus of singing voice examples. The universal model is a prescription for learning a model from general training examples and incorporating the model in NMF-based source separation [11].

The idea is to independently learn a matrix of basis vectors for each of M singers from training data of the individual singers. This yields M matrices of basis vectors $\boldsymbol{W}_1, ..., \boldsymbol{W}_M$. The universal voice model is then simply the concatenation of the matrices of basis vectors:

$$\boldsymbol{W}_V = \begin{bmatrix} \boldsymbol{W}_1 & ... & \boldsymbol{W}_M \end{bmatrix}$$

The hope is that an unseen singer is sufficiently similar to one or a blend of a few of these singers, so that the universal voice model can act as a singer-independent surrogate for singer dependent models.

In applying the universal voice model for source separation, we make the assumption that the activation matrix for the singing voice

$$\boldsymbol{H}_V = \begin{bmatrix} \boldsymbol{H}_1 \\ \vdots \\ \boldsymbol{H}_M \end{bmatrix}$$

is block sparse, i.e., several of the $H_i \equiv 0$. This is necessary because the number of singers is typically large, and the matrix factorization problem can be underdetermined. The block sparsity is a regularization strategy that incorporates the structure of the problem; it captures the intuition that only a few voice models should be sufficient to explain any given singer. We achieve block sparsity by adding a penalty function Ω to the objective function to encourages this structure. λ controls the strength of the penalty term.

$$\underset{\boldsymbol{W},\boldsymbol{H}}{\text{minimize}} \; D\left(\boldsymbol{X}||\boldsymbol{W}\boldsymbol{H}\right) + \lambda\Omega(\boldsymbol{H}_V) \tag{1}$$

As in [11], we choose the Kullback-Leibler divergence for D:

$$D(\boldsymbol{Y}||\boldsymbol{Z}) = \sum_{i,j} Y_{ij} \log \frac{Y_{ij}}{Z_{ij}} - Y_{ij} + Z_{ij}$$

and a concave penalty on the ℓ_1 norm of the block:

$$\Omega(\boldsymbol{H}_V) = \sum_{i=1}^{M} \log(\epsilon + ||H_i||_1)$$

The algorithm for optimizing (1) is known as Block KL-NMF. Further details can be found in [11].

3. COMBINED APPROACH

In this section, we review an existing method for modeling the background music, and we propose to use it to refine the residual from the singing voice modeling.

3.1 Modeling Background Music

A number of methods have been proposed to estimate the background music, without prior vocal/non-vocal segmentation, by assuming the background to be repeating and the foreground (i.e., the singing voice) to be non-repeating. REPET-SIM is thus a generalization of the REpeating Pattern Extraction Technique (REPET) [1] , a simple approach for separating the repeating background from the non-repeating foreground in a mixture, by identification of the repeating elements and the smoothing of the non-repeating elements.

In particular, REPET-SIM uses a similarity matrix to identify the repeating elements in the mixture - which ideally correspond to the background music, followed by median filtering to smooth out the non-repeating elements - which ideally correspond to the singing voice [8]. Unlike the earlier variants of REPET that use a beat spectrum or beat spectrogram to identify the periodically repeating patterns [6, 9], REPET-SIM uses a similarity matrix and is thus able to handle backgrounds where repeating patterns can also happen non-periodically.

3.2 Combined Approach

In order to improve the music/voice separation that we obtain from using the universal voice model alone, we propose cascading the model with REPET-SIM. The idea is that the universal voice model specifically models the singing voice and, through the residual, provides a preliminary estimate of the background music, which can then be refined by feeding it to REPET-SIM. The pipeline is shown in Figure 1, and detailed below.

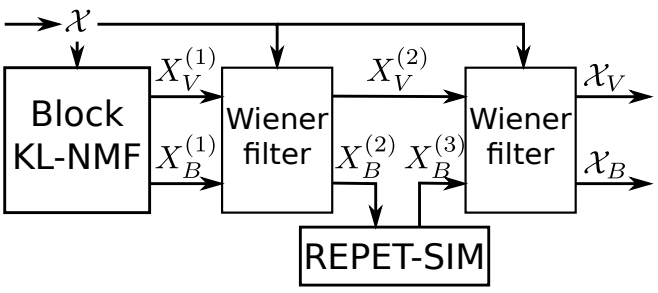


Figure 1. Combined approach which takes in the spectrogram of a mixture $\mathcal{X}$ and returns refined estimates of the spectrogram of the singing voice $\mathcal{X}_V$ and the background music $\mathcal{X}_B$

The universal voice model first outputs an estimate for the magnitude spectrogram of the singing voice $X_V^{(1)}$, and a residual $X_B^{(1)}$ corresponding to the background estimate, which are initially filtered into $X_V^{(2)}$ and $X_B^{(2)}$ by using Wiener filtering, as follows:

$$X_V^{(2)} = \left| \frac{X_V^{(1)}}{X_V^{(1)} + X_B^{(1)}} \odot \mathcal{X} \right|$$

$$X_B^{(2)} = \left| \frac{X_B^{(1)}}{X_V^{(1)} + X_B^{(1)}} \odot \mathcal{X} \right|$$

where $\mathcal{X}$ denotes the complex spectrogram of the mixture and $\odot$ the Hadamard (component-wise) product. Wiener filtering is used here to reduce separation artifacts. Note that we only use the magnitudes of the estimates here.

The background estimate from the universal voice model $X_B^{(2)}$ is then fed to REPET-SIM which refines it into $X_B^{(3)}$.

The estimates for the complex spectrogram of the singing voice $\mathcal{X}_V$ and the background music $\mathcal{X}_B$ are finally obtained by filtering $X_V^{(2)}$ and $X_B^{(3)}$ using Wiener filtering, as follows:

$$\mathcal{X}_V = \frac{X_V^{(2)}}{X_V^{(2)} + X_B^{(3)}} \odot \mathcal{X}$$

$$\mathcal{X}_B = \frac{X_B^{(3)}}{X_V^{(2)} + X_B^{(3)}} \odot \mathcal{X}$$

4. EVALUATION

In this section, we evaluate the method for modeling the singing voice and the combined approach on a data set of 1,000 song clips. We also compare them with the method for modeling the background music alone, as well as two other state-of-the-art methods.

[1] http://music.eecs.northwestern.edu/research.php?project=repet

4.1 Data Set

The MIR-1K data set [2] consists of 1,000 song clips in the form of split stereo WAV files sampled at 16 kHz, with the background music and the singing voice recorded on the left and right channels, respectively. The song clips were extracted from 110 karaoke Chinese pop songs performed by 8 female and 11 male singers. The durations of the clips range from 4 to 13 seconds [3].

We created a set of 1,000 mixtures by summing, for each song clip, the left (background music) and right (singing voice) channels into a monaural mixture

4.2 Performance Measures

The BSS Eval toolbox [3] consists of a set of measures that intend to quantify the quality of the separation between a source and its estimate. The principle is to decompose an estimate into a number of contributions corresponding to the target source, the interference from unwanted sources, and the artifacts such as "musical noise."

Based on this principle, the following measures were then defined (in dB): Sources to Interferences Ratio (SIR), Sources to Artifacts Ratio (SAR), and Sources to Distortion Ratio (SDR) which measures the overall error [13].

4.3 Competitive Methods

Durrieu et al. proposed a method [4] based on the modeling of a mixture as an instantaneous sum of a signal of interest (i.e., the singing voice) and a residual (i.e., the background music), where the singing voice is parametrized as a source-filter model, and the background music as an unconstrained NMF model [1]. The parameters of the models are then estimated using an iterative algorithm in a formalism similar to NMF. A white noise spectrum is added to the singing voice model to better capture the unvoiced components. We used an analysis window of 64 milliseconds, a window size of 1024 samples, a step size of 32 milliseconds, and 30 iterations.

Huang et al. proposed a method [5] based on Robust Principal Component Analysis (RPCA) [4]. RPCA is a method for decomposing a data matrix into a low-rank component and a sparse component, by solving a convex optimization problem that aims to minimize a weighted combination of the nuclear norm and the L_1 norm. The method assumes that the background music typically corresponds to the low-rank component and the singing voice typically corresponds to the sparse component.

4.4 Training Universal Models

Our experiments used a leave-one-out cross validation approach. For each of the 19 singers, we learned a universal model using NMF on the other 18 singers, with different choices for the number of basis vectors per singer: $K = 5, 10, 20, 30, 40, 50, 60$.

[2] http://sites.google.com/site/unvoicedsoundseparation/mir-1k

[3] http://bass-db.gforge.inria.fr/bss_eval/

[4] http://www.durrieu.ch/research/jstsp2010.html

[5] https://sites.google.com/site/singingvoiceseparationrpca/

4.5 Parameters

We used a Hamming window of 1024 samples, corresponding to a duration of 64 milliseconds at a sampling frequency of 16 kHz, with 50% overlap.

For REPET-SIM [6], pilot experiments showed that a minimal threshold of 0, a maximal order of 50, and a minimal distance of 0.1 second gave good separation results.

For the universal voice model, pilot experiments showed that different settings of K, K_B (number of background music basis vectors), and λ yielded optimal results for different measures (see Section 4.2) of the separation quality of singing voice and background music. We considered $K = 5, 10, 20, ..., 60$, $K_B = 5, 10, 20, 30, 50, 80$, and a logarithmic grid of λ values.

4.6 Comparative Results

Figures 2, 3, and 4 show the boxplots of the distributions for the SDR, SIR, and SAR (in dB) for the background music (left plot) and the singing voice (right plot) estimates, for the method of Durrieu et al. (*Durrieu*), the method of Huang et al. (*Huang*), REPET-SIM alone (*REPET*), universal voice model alone (*UVM*), and the combination of universal voice model and REPET-SIM (*combo*). The horizontal line in each box represent the median of the distribution, whose value is displayed above the box. Outliers are not shown. Higher values are better.

We used two parameter settings for the universal voice model: one that gave the best SDR for the background music estimates ($K = 20$, $K_B = 5$, and $\lambda = 1448$), and one that gave the best SDR for the singing voice estimates ($K = 10$, $K_B = 5$, and $\lambda = 2896$). The boxplots then show the results for the background music estimates (left plots) and the singing voice estimates (right plots) for the parameter settings that gave the best SDR, for the universal voice model (*UVM*) and the combination (*combo*).

The plots show that the universal voice model alone, for the right parameter settings, achieves higher SDR than REPET-SIM and the other state-of-the-art methods, for both the background music and the singing voice estimates. Combining the universal voice model with REPET-SIM typically yields further improvement.

If we focus on SIR, for the background music estimates, the universal voice model alone achieves higher SIR than REPET-SIM and the other competitive methods; the combination further increases the SIR. For the singing voice estimates, the universal voice model alone achieves higher SIR than REPET-SIM and the method of Huang et al., but the combination does no better than the universal voice model alone.

On the other hand, if we focus on SAR, for the background music estimates, the universal voice model alone has slightly lower SAR than REPET-SIM and the other competitive methods; the combination further decreases the SAR. For the singing voice estimates, the universal voice model alone has higher SAR than the method of Durrieu et al.; the combination further improves the results.

[6] http://music.eecs.northwestern.edu/includes/projects/repet/codes/repet_sim.m

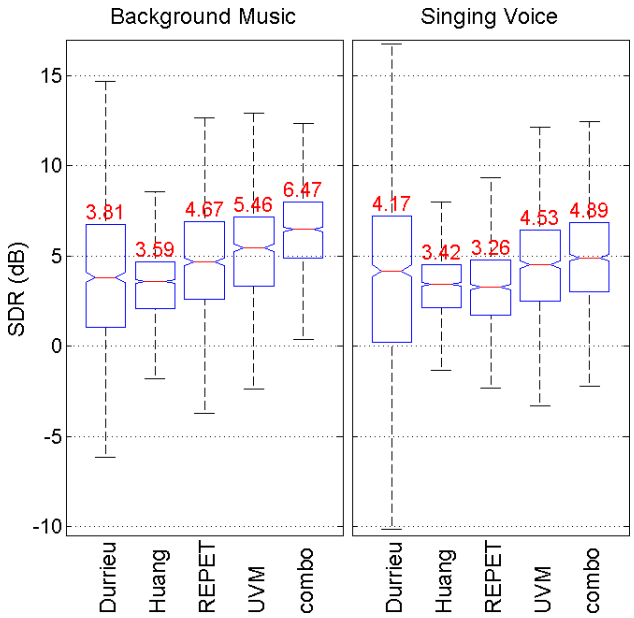


Figure 2. Box plots of the distributions for the SDR (dB).

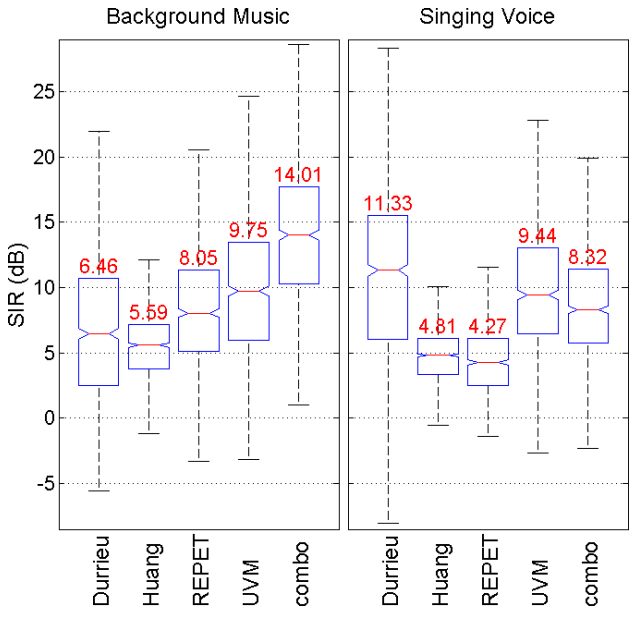


Figure 3. Box plots of the distributions for the SIR (dB).

These results show that, given the right parameter settings, the universal voice model is particularly good at reducing in one source the interference of the other source, however at the expense of adding some artifacts in the estimates. This is related to the SIR/SAR performance trade-off commonly seen in source separation.

The results also show that combining the universal voice model with REPET-SIM helps to increase the SIR for the background music estimates and the SAR for the singing voice estimates, but at the expense of decreasing the SAR for the background music estimates and the SIR for the singing voice estimates. This is related to the music/voice performance trade-off commonly seen in music/voice separation. In other words, the combination helps to reduce in the background music estimates the interference from the singing voice but at the expense of introducing some artifacts in the estimates. On the other hand, it helps to reduce artifacts in the singing voice estimates, at the expense of introducing interference from the background music.

4.7 Statistical Analysis

We compared the SDR of the background music and singing voice estimates across the different methods using a two-sided paired t-test. The universal voice model alone achieved a significantly higher SDR on the background music than the three state-of-the-art methods: the closest competitor was REPET-SIM ($t = 3.92$, $p < .0001$). The combination represented a significant improvement over the universal model alone ($t = 19.4$, $p \approx 0$). A similar story is true for the SDR of the singing voice estimates: the universal voice model alone is significantly better than any of the existing methods, with the method of Durrieu et al. the closest competitor ($t = 6.13$, $p \approx 0$), and the combination represents a significant improvement over it ($t = 13.8$, $p \approx 0$).

In terms of the SIR of the background music estimates, the combination is significantly better than the universal voice model alone ($t = 37.7$, $p \approx 0$), which is significantly better than any of the existing methods, with the closest competitor being REPET-SIM ($t = 7.75$, $p \approx 0$). For the SIR of the singing voice estimates, the universal voice model is not significantly different from the method of Durrieu et al. ($t = -0.29$, $p = .77$), but significantly better than the other existing methods, and also the combination ($t = 20.1$, $p \approx 0$).

Finally, for the SAR of the background music estimates, the universal voice model is competitive with REPET-SIM ($t = -1.26$, $p = 0.21$), but significantly worse than the other competitive methods ($t = 9.61$ and $t = 13.2$). On the other hand, in terms of the SAR of the singing voice estimates, the combination performs significantly better than the universal voice model ($t = 50.1$), which in turn is significantly better than the method of Durrieu et al. ($t = 9.69$). However, both are significantly worse than the other two competitors, the closest being the method of Huang et al. ($t = -15.6$).

Note that there are 7 tests for each configuration of the three measures (SDR, SIR, SAR) and the two sources (background music and singing voice): comparing the universal voice model and the combination to each of the three competitors, and then comparing the universal voice model to the combination. Therefore, we are implicitly conducting a total of $3 \times 2 \times 7 = 42$ tests. All of the findings above remain significant at the $\alpha = .05$ level if we use a Bonferroni correction to adjust for the 42 tests, corresponding to a rejection region of $|t| > 3.25$. These results confirm the findings in Figures 2, 3, and 4.

5. CONCLUSION

In this work, we proposed a method for modeling the singing voice. The method can learn a singer-independent model from singing examples using a NMF based technique. We

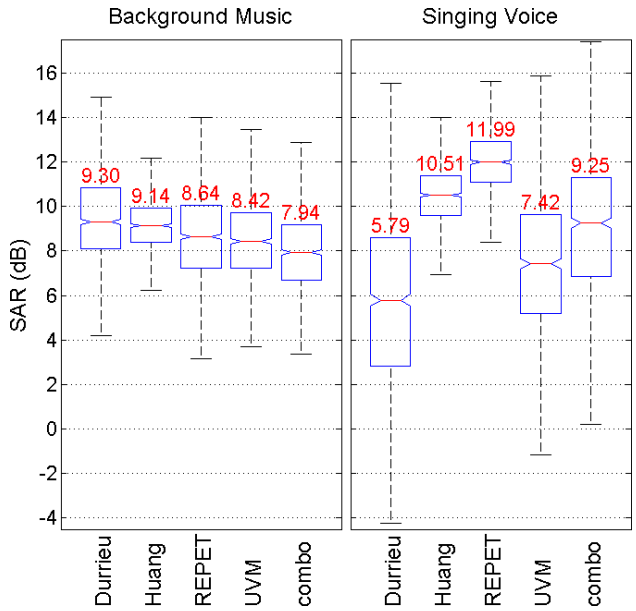


Figure 4. Box plots of the distributions for the SAR (dB).

then proposed to combine this method with a method that models the background music. Combining a method that specifically models the singing voice with a method that specifically models the background music addresses separation performance from the point of view of both sources.

Evaluation on a data set of 1,000 song clips showed that, when using the right parameter settings, the universal voice model can outperform different state-of-the-art methods. Combining modeling of both sources can further improve separation performance, when compared with modeling only one of the sources.

This work was supported in part by NSF grant number IIS-0812314.

6. REFERENCES

[1] Jean-Louis Durrieu, Bertrand David, and Gaël Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *IEEE Journal on Selected Topics on Signal Processing*, 5(6):1180–1191, October 2011.

[2] Jinyu Han and Ching-Wei Chen. Improving melody extraction using probabilistic latent component analysis. In *36th International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, May 22-27 2011.

[3] Chao-Ling Hsu and Jyh-Shing Roger Jang. On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2):310–319, February 2010.

[4] Po-Sen Huang, Scott Deeann Chen, Paris Smaragdis, and Mark Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *37th International Conference on Acoustics, Speech and Signal Processing*, Kyoto, Japan, March 25-30 2012.

[5] Yipeng Li and DeLiang Wang. Separation of singing voice from music accompaniment for monaural recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1475–1487, May 2007.

[6] Antoine Liutkus, Zafar Rafii, Roland Badeau, Bryan Pardo, and Gaël Richard. Adaptive filtering for music/voice separation exploiting the repeating musical structure. In *37th International Conference on Acoustics, Speech and Signal Processing*, Kyoto, Japan, March 25-30 2012.

[7] Alexey Ozerov, Pierrick Philippe, Frédéric Bimbot, and Rémi Gribonval. Adaptation of Bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(5):1564–1578, July 2007.

[8] Zafar Rafii and Bryan Pardo. Music/voice separation using the similarity matrix. In *13th International Society for Music Information Retrieval*, Porto, Portugal, October 8-12 2012.

[9] Zafar Rafii and Bryan Pardo. REpeating Pattern Extraction Technique (REPET): A simple method for music/voice separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(1):71–82, January 2013.

[10] Paris Smaragdis, Bhiksha Raj, and Madhusudana Shashanka. Supervised and semi-supervised separation of sounds from single-channel mixtures. In *Independent Component Analysis and Signal Separation*, pages 414–421. Springer, 2007.

[11] Dennis L. Sun and Gautham J. Mysore. Universal speech models for speaker independent single channel source separation. In *38th International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, May 26-31 2013.

[12] Shankar Vembu and Stephan Baumann. Separation of vocals from polyphonic audio recordings. In *6th International Conference on Music Information Retrieval*, pages 337–344, London, UK, September 11-15 2005.

[13] Emmanuel Vincent, Rémi Gribonval, and Cedric Févotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 14(4):1462–1469, July 2006.

[14] Tuomas Virtanen, Annamaria Mesaros, and Matti Ryynänen. Combining pitch-based inference and non-negative spectrogram factorization in separating vocals from polyphonic music. In *ISCA Tutorial and Research Workshop on Statistical and Perceptual Audition*, pages 17–20, Brisbane, Australia, 21 September 2008.

ON FINDING SYMBOLIC THEMES DIRECTLY FROM AUDIO FILES USING DYNAMIC PROGRAMMING

Antti Laaksonen[1], **Kjell Lemström**[2,1]
[1] Department of Computer Science, University of Helsinki
[2] Laurea University of Applied Sciences
ahslaaks@cs.helsinki.fi, kjell.lemstrom@laurea.fi

ABSTRACT

In this paper our goal is to find occurrences of a theme within a musical work. The theme is given in a symbolic form that is searched for directly in an audio file. We present a dynamic programming algorithm that is related to an existing time-warp invariant algorithm. However, the new algorithm is computationally more efficient than its predecessor, and it can also be used for approximate time-scale invariant search. In the latter case the note durations in the query are taken into account, but some time jittering is allowed for. When dealing with audio, these are important properties because the number of possible note events is large and the note positions are not exact. We evaluate the algorithm using a collection of themes from Tchaikovsky's symphonies. The new approximate time-scaled algorithm seems to be a good choice for this setting.

1. INTRODUCTION

In this paper we present new content-based music retrieval (CBMR) algorithms for discovering occurrences of a given musical theme, called the *pattern*, in a musical work under consideration, called the *database*. The algorithms have both important scientific applications and theoretical interest: they can be used in music analysis to locate occurrences of themes in a single musical work or to search for a given theme within a music database. Moreover, to the best of our best knowledge, our algorithm for time-warped search is the most efficient algorithm available for the task.

We allow both the pattern and the database to be polyphonic. Traditionally CBMR problems like this have been tackled by using a linear string representation combined with a string matching algorithm. However, the representation does not effectively capture important intrinsic features of music. The geometric modeling of music [8] is more appropriate for this case and has recently been successfully used by several authors [4, 7, 10, 13]. Reasonable geometric modelling allows the matching process to ignore extra intervening notes in the database that do not appear in the query. Such extra notes always occur because of polyphony and unexpected noise.

Until recently, the geometric framework suffered from the fact that the timing of the notes had to be exact. Recent studies have challenged this problem. First, in [5, 10] the occurrences were allowed to be transposed and time-scaled copies of the query. Under this *transposition and time-scale invariance* (the *TTSI* setting), however, the queries still need to be given exactly in tempo. Under a more realistic, *transposition and time-warp invariance* (the *TTWI* setting), local time jittering is allowed for in every note-onset. The first solutions for this setting were introduced by Lemström and Laitinen [4, 6].

In this paper we introduce a new dynamic programming algorithm that can be used for both time-scale and time-warp invariant search. As an application of the algorithm, we search for musical themes from audio files. Only a few other algorithms for symbolic queries from audio have been proposed in the literature [2, 11, 12]. Our experiment shows that our algorithms are efficient in practice, and the music search results from audio files are also promising.

Let us next define the problems under consideration. The problems are elaborations of Ukkonen, Lemström and Mäkinens P1 problem [13]. In each case, the music is represented in a pitch-against-time representation of note-on information. The database T contains n note events, and the query pattern P contains m note events. In a typical retrieval case, the pattern P is monophonic and much shorter than the polyphonic database T. The problems are:

W1: Find *time-warped translations* of P such that each note in P matches with a note in T [4, 6].

S1: Find *time-scaled translations* of P such that each note in P matches with a note in T [5, 10].

AS1: Find *approximate time-scaled translations* of P such that each note in P matches with a note in T.

The new problem (AS1) can be seen as an intermediate form of W1 and S1. Instead of a constant time-scaling factor α, we allow the scaling to vary in a range $[\alpha/\epsilon, \alpha\epsilon]$ where $\epsilon \geq 1$. If $\epsilon = 1$, the problem is the same as S1.

2. ALGORITHMS

The dynamic programming algorithm of Laitinen and Lemström [4] solves the problem W1 in $O(nmw)$ time where w is the window size. In this section we present a modification for the algorithm that reduces the time complexity to

$O(nm)$. Furthermore, the new algorithm also solves problems S1 and AS1 in $O(nms)$ time where s is the number of possible scaling factors.

The input of the algorithm consists of two lists: the database notes $T[1], T[2], \ldots, T[n]$ and the pattern notes $P[1], P[2], \ldots, P[m]$. Each database note $T[k]$ is a triplet: $T[k].time$ is the onset time, $T[k].pitch$ is the pitch value and $T[k].score$ is a real value in the range $[0, 1]$. Each pattern note $P[k]$ is a pair with the fields $P[k].time$ and $P[k].pitch$. We assume that both of the lists are sorted in increasing order according to the onset times and that the pitches are represented in semitones.

The score field of the database notes is only used with audio material. If the input is symbolic music, each score is simply set 1. However, when audio material is used, the note events are approximations. The idea is that the score reflects the amplitudes of the frequencies that correspond to the note in the audio.

The output of the algorithm is a list $S[1], S[2], \ldots, S[n]$ consisting of real values. Each value $S[k]$ is the maximum score for a pattern occurrence starting from the database note k. In an occurrence, each pattern note is matched with a database note, and the score is calculated as a sum of individual scores of the database notes. In the case of a symbolic database, a score of m denotes that the pattern is found, and in the case of an audio database a high score suggests that the pattern is found.

Next, in Section 2.1 we present our modifications to the time-warped search algorithm. After this, in Section 2.2, we show how the algorithm can also be used for time-scaled search.

2.1 Time-warped search

Let us next introduce a reformulation of the dynamic programming algorithm for W1 that was presented in [4]. The original algorithm uses a three-dimensional array $M[t, p, c]$ where t and p refer to a database and pattern note and c is a counter that limits the search window. Instead of this, we use a two-dimensional array $M[t, p]$ and go through all possible values of c.

for $p = m \to 1$ **do**
  for $t = n \to 1$ **do**
    if $p = m$ **then**
      $M[t, p] = T[t].score$
    else
      $b = -\infty$
      for $c = 1 \to \min(w, n - t)$ **do**
        $d_1 = T[t + c].pitch - T[t].pitch$
        $d_2 = P[p + 1].pitch - P[p].pitch$
        if $d_1 = d_2$ **then**
          $b = \max(b, M[t + c, p + 1] + T[t].score)$
        end if
      end for
      $M[t, p] = b$
    end if
  end for
end for

This algorithm can be used to calculate the list S because $S[k] = M[k, 1]$ for each k. The time complexity is still $O(nmw)$, but next we will show how to reduce it to $O(nm)$. The idea is to remove the innermost loop and find the best value more efficiently. To achieve this, we maintain a deque of maximum values for each possible pitch and use a standard sliding window maximum algorithm. Note that we assume that the set of possible pitches is constant. This is a very natural assumption when we are working with pitch values in semitones.

Consider the situation where p and t have been fixed and $p < m$. We are looking for maximum value $M[t', p + 1]$ where $t < t' \leq t + w$ with the constraint that $T[t'].pitch = T[t].pitch + P[p + 1].pitch - P[p].pitch$. Now, for each pitch v, we maintain a deque $D[v]$ containing pairs whose first element is the value $M[t', p + 1]$ and second element is the position t'. The last pair in the deque always corresponds to the maximum value in the window.

We update the deques as follows. When the value t has changed, we always add the pair $(M[t+1, p+1], t+1)$ to the front of deque $D[T[t + 1].pitch]$ and remove the pair $(M[t + w + 1, p + 1], t + w + 1)$ from the tail of deque $D[T[t + w + 1].pitch]$. Before adding the new pair, we remove all pairs from the front of the deque whose value is smaller than the new value. When removing the pair, we require that the second element equals the current position (i.e. the pair has not been removed earlier).

The amortized time complexity for finding all the maximum values for fixed p is $O(n)$ because every value is added and removed to a deque only once. Thus the time complexity of the whole algorithm is $O(nm)$.

The following pseudocode clarifies the structure of the $O(nm)$ algorithm. The array D contains a deque for each possible note, and the deque operations work as described above. Note that the operation $remove$ only takes the second element of the pair to be removed.

for $p = m \to 1$ **do**
  $D.clear()$
  for $t = n \to 1$ **do**
    if $p = m$ **then**
      $M[t, p] = T[t].score$
    else
      if $t + 1 \leq n$ **then**
        $D[T[t+1].pitch].add((M[t+1, p+1], t+1))$
      end if
      if $t + w + 1 \leq n$ **then**
        $D[T[t + w + 1].pitch].remove(t + w + 1)$
      end if
      $v = T[t].pitch + P[p + 1].pitch - P[p].pitch$
      if $D[v].empty()$ **then**
        $M[t, p] = -\infty$
      else
        $M[t, p] = D[v].maximum() + T[t].score$
      end if
    end if
  end for
end for

Note that if we use a symbolic database and every note score is 1, the deques are not needed. In this case it is sufficient to store, for each pitch v, the most recent position $R[v]$, for which $M[t', p+1] = n - p$. If $R[v] \leq t + w$, then $M[t, p]$ equals $n - p + 1$ and otherwise $-\infty$.

2.2 Time-scaled search

The algorithm described in Section 2.1 can also be used for time-scaled search. The only difference is that the size of the window is not constant. Instead of this, the size of the window depends on the position in the pattern and the time values of the notes. However, the sliding window maximum algorithm can still be used and the time complexity of the algorithm remains $O(nm)$ when the range of the scaling factor is fixed.

Let α be the scaling factor and ϵ be the jittering tolerance. In this case, pattern notes $P[p]$ and $P[p+1]$ can be matched to database notes $T[t]$ and $T[t']$ only if $(P[p+1].time - P[p].time)\alpha/\epsilon \leq T[t'].time - T[t].time$ and $(P[p+1].time - P[p].time)\alpha\epsilon \geq T[t'].time - T[t].time$. In other words, we are looking for t' values for which $t + w_1 \leq t' \leq t + w_2$, $T[t + w_1]$ is the first note that fulfils the first condition and $T[t + w_2]$ is the last note that fulfils the second condition.

We can combine the parameters w_1 and w_2 with the algorithm presented in Section 2.1 as follows. Consider the situation for fixed p. First, we set $w_1 = w_2 = n$. When t changes, we decrease w_1 as long as $(P[p+1].time - P[p].time)\alpha/\epsilon \leq T[w_1 - 1].time - T[t].time$. Then, we decrease w_2 as long as $(P[p+1].time - P[p].time)\alpha\epsilon \leq T[w_2 - 1].time - T[t].time$. When decreasing w_1, we add new values to the deque, and when decreasing w_2, we remove old values. Even if the deque operations are more irregular than in the time-warped search, each value is added and removed only once and the time complexity remains $O(nm)$.

In practice, the correct α value is unknown and several searches have to be performed. Thus, the time complexity becomes $O(nms)$ where s is the number of searches. In practice, however, the number of scaling factors that needs to be tested is small because the duration of a single note is typically at most some seconds.

3. EXPERIMENT

In this section, we present the results of our experiment where we searched for themes within audio material using the algorithms described in Section 2. We consider two scenarios: (1) finding the location of the given theme in a single audio file, and (2) finding the audio file which contains the given theme. Moreover, we analyse the performance of the $O(nmw)$ and $O(nm)$ algorithms in practice.

3.1 Material

The material used in the experiment, shown in Table 1, consists of Tchaikovsky's six symphonies.

We created the audio database from Deutsche Grammophon recordings found under the catalogue id 423 504-2. We converted the material from CD tracks into mono WAV files with a sample rate 44,100 Hz. Each audio file contains a single movement of a symphony. The total duration of the audio is 4 hours and 22 minutes.

Work	Movements	Total length	Themes
Symphony 1	4	44 min	10
Symphony 2	4	35 min	11
Symphony 3	5	46 min	15
Symphony 4	4	42 min	13
Symphony 5	4	49 min	15
Symphony 6	4	45 min	11
Total	25	262 min	75

Table 1. The material used in the experiment.

The themes to be searched for were taken from the book *A Dictionary of Musical Themes* [1]. The book contains 75 themes from Tchaikovsky's symphonies and we used all of them in the experiment. Each theme has an identifier in the form of TXXX where X is a digit. The themes are also available online [1] as MIDI files which we automatically converted to simple symbolic notation. In addition, for each theme, we marked the locations where it appears in the audio material to allow automatic evaluation.

The duration of a theme ranges from about 5 seconds to about 25 seconds. We kept all of the themes unchanged with two exceptions. First, theme T231 actually consists of two themes from which the second one is very short. They are indexed separately online, and we have removed the second theme from our material. Second, there is an error in the notation of theme T243 both in the book and online: only the first two triplets are marked and the other notes are too slow. We corrected this in our material.

3.2 Audio processing

Our algorithms require that the database is represented as a list of note events. However, there is no direct way to convert an audio file into symbolic notation. Therefore we estimated the note events using the discrete Fourier transform. In general, strong notes in the music can be observed as strong frequencies in the audio signal.

We used standard techniques for processing the audio signal: we divided each audio file into frames of 4,096 samples. After this, we calculated a frequency spectrum for each frame using the discrete Fourier transform. We assumed that the frequency of A4 is 440 Hz and calculated the amplitude of its frequency in the spectrum for each note from G3 to B5. The selected note range encompasses a large part of the melodies in orchestral music and avoids very low and very high notes. All the amplitudes were normalized and estimated using linear interpolation.

We built three databases of note events, which are summarized in Table 2. The value k refers to the amount of note events produced from a single audio file. The values $\min k$ and $\max k$ are the minimum and maximum amounts

[1] `http://www.multimedialibrary.com/barlow/`

Database	$\min k$	$\max k$	n
D1	107,793	353,249	4,948,386
D2	33,830	107,914	1,541,098
D3	37,510	122,703	1,732,997

Table 2. The number of note events in the databases.

of note events in a file, and the value n denotes the total number of note events in the database.

Database D1 contains a note event for each note in each frame, and the score of the note event is the amplitude of the note in the spectrum. In this case the database includes a lot of note events with low scores that are unlikely to be a part of a theme occurrence.

Databases D2 and D3 are subsets of D1, and the aim is to locate potential note candidates from the full spectrum. Database D2 contains the notes whose amplitudes are peak amplitudes (i.e. the note with pitch p is selected only if notes with pitches $p-1$ and $p+1$ have a weaker amplitude). Database D3 contains the 10 strongest notes in terms of the amplitude for each frame. A value of 10 is selected so that the sizes of D2 and D3 are nearly equal to each other.

This estimation of note events resembles salience value calculation in automatic melody extraction (for a review, see [9]). However, salience values are usually calculated as combinations of the harmonics of the note. We experimented with various ways to use higher harmonics, but to our surprise, using only the fundamental frequency produced the best results with this material.

3.3 Algorithms

We implemented the audio processing method described in Section 3.2 and the time-warped and time-scaled algorithms described in Sections 2.1 and 2.2 by using C++ and FFTW library. For the rest of the paper, we call the time-warped algorithm W and the time-scaled algorithm S.

When using algorithm W, we had to choose the window length w. In this experiment, w has to be relatively high because there can be a large number of note events between two consecutive theme notes in the database. More precisely, the amount of note events per second is about 300 in D1 and about 100 in D2 and D3, and a note in a theme occurrence can have a duration of several seconds.

In algorithm S there are several parameters. First, the scaling factors α have to be specified. In this experiment we assumed that the duration of theme is between 5 and 25 seconds and tested α values for which the time between the first and last note in the theme is $5, 6, 7, \ldots, 25$ seconds. The jittering tolerance ϵ was more difficult to choose; in orchestral music the rhythm is usually quite exact, thus indicating a small value to be the most suitable option.

Moreover, we had to choose how the occurrences of the themes were reported. For this purpose we used the parameter δ, a real number in the range $[0, 1]$. All occurrences with a score of at least δb were reported, where b is the best score. The lower the δ is, the more results are produced. However, a small δ would result in a large number of false positives. Finally, we required that the interval between two reported occurrences is at least 5 seconds. This was done to prevent one occurrence from being reported multiple times.

Database	Metric	W	S
D1	Themes found	29	34
	Total results	76	77
	Precision	0.38	0.44
D2	Themes found	30	29
	Total results	76	77
	Precision	0.39	0.38
D3	Themes found	31	33
	Total results	76	77
	Precision	0.41	0.43

Table 3. The best results of the algorithms. For W, parameter w is 500 in D1 and 150 in D2 and D3, and for S, parameter ϵ is 1.50

$w \setminus \delta$	1.00	0.99	0.95	0.90	0.75
100	12	15	28	46	69
	0.16	0.14	0.09	0.05	0.01
200	22	25	35	52	70
	0.29	0.24	0.12	0.06	0.01
500	29	34	51	64	71
	0.38	0.28	0.09	0.03	0.01
1000	23	32	61	71	74
	0.24	0.15	0.04	0.02	0.01
2000	22	41	72	73	74
	0.07	0.05	0.02	0.01	0.01

Table 4. The results of W on database D1 when w and δ change. Themes found and precision levels are reported.

3.4 File search

In the first part of the experiment, we searched for each theme in the file where it appears. We calculated three measures: the number of themes located correctly, the total number of occurrences reported and the search precision as the ratio of previous two values. We considered that the theme was found if the difference between one of the occurrences reported by the algorithm and one of the occurrences in the ground truth was less than 5 seconds.

Table 3 shows the best results of the algorithms. The parameter δ is 1.00 in each case. For algorithm W, parameter w is 500 in D1 and 150 in D2 and D3. For algorithm S, parameter ϵ is 1.50. Interestingly, the best results were achieved in D1 which contains a note event for each possible note in each audio frame. The results of S were in general somewhat better than the results of W.

Note that in some cases there can be several occurrences even if δ is 1.00. For example, when using W and D1, the number of occurrences is 76 instead of 75. In those cases all normalized note scores in the occurrences are 1, thus the total score is m where m is the number of notes.

$\epsilon \setminus \delta$	1.00	0.99	0.95	0.90	0.75
1.10	25	29	37	51	70
	0.33	0.27	0.13	0.06	0.02
1.20	30	34	43	53	71
	0.40	0.33	0.14	0.06	0.02
1.33	33	38	46	58	71
	0.44	0.36	0.14	0.06	0.02
1.50	34	37	49	60	72
	0.44	0.36	0.13	0.05	0.01
2.00	30	37	49	63	73
	0.36	0.25	0.07	0.03	0.01

Table 5. The results of S on database D1 when ϵ and δ change. Themes found and precision levels are reported.

Table 4 shows the results of W in D1 with different settings of parameters w and δ. The optimal value for w is approximately 500. Finally, Table 5 shows the results of S in D1 with different settings of parameters ϵ and δ change. The optimal value for ϵ is approximately 1.50. Using larger δ values, S would still find themes accurately in some situations. For example, when $\delta = 0.99$ and $\epsilon = 1.33$, S found 38 themes with a precision level of 0.36.

The database used in the experiment is challenging in general; finding some of the themes requires a moderate amount of work even for an experienced human listener. An interesting phenomenon is that the results of the algorithms varied very strongly depending on the symphony. For example, S was able to find 9 out of 10 themes from Symphony 1, but only 1 out of 11 themes from Symphony 6. A possible explanation for this is that the themes in Tchaikovsky's later works are more subtle.

3.5 Database search

In the second part of the experiment, we searched for each theme in the database D1. This resembles the query-by-humming setting [3]; however, our queries are exact (for example, created by a musician) and the database is atypical because it contains a small number of files, each having more than 100,000 note events.

The search was conducted as follows. For each file in the database, we calculated the maximum score for an occurrence of the theme. Having done this, we constructed a list of files that were sorted in decreasing order according to their scores. Following the usual convention, we concentrated on the rank of the correct file (the file where the theme actually appears) in the sorted list. For example, a rank of 5 means that the correct file has the 5th highest score in the search results.

Figures 1 and 2 show the distribution of ranks using W and S in D1 with the same parameters as in Table 3. There were 18 and 22 themes with a rank of 1, and 26 and 31 themes, respectively, with a rank of 1–3. Somewhat surprisingly, in this material locating the theme in the whole database was not much more difficult than locating the theme in the correct file.

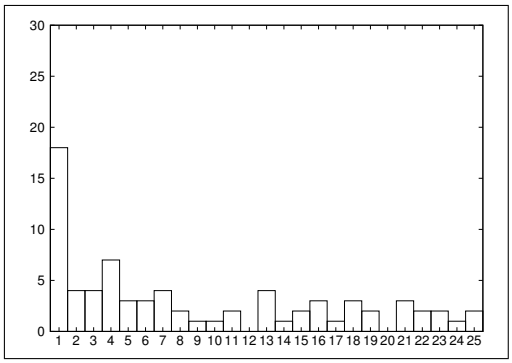


Figure 1. Distribution of correct file ranks using algorithm W and database D1.

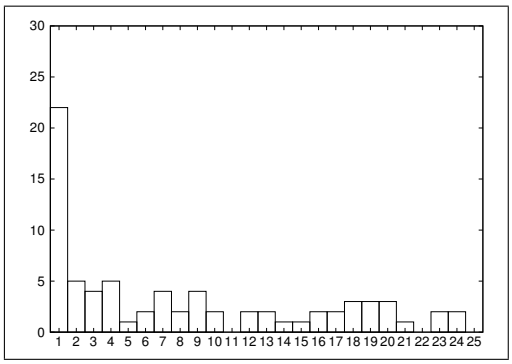


Figure 2. Distribution of correct file ranks using algorithm S and database D1.

3.6 Performance

Finally, we analyzed the practical performance of the algorithms. We ran the tests on an Intel Core 2 Duo Processor with a clock rate of 3.16 GHz.

Figure 3 shows the running times of three algorithms: S is the $O(nms)$ implementation of the time-scaled algorithm, and W and W' are $O(nm)$ and $O(nmw)$ implementations of the time-warped algorithm. We searched for theme T252 with 27 notes from each audio file in database D1 using parameters $\delta = 1.00$, $\epsilon = 1.50$ and $w = 500$, as shown in Table 3. As discussed previously, the variable k denotes the number of note events in the audio file.

As expected, W clearly outperformed the other two al-

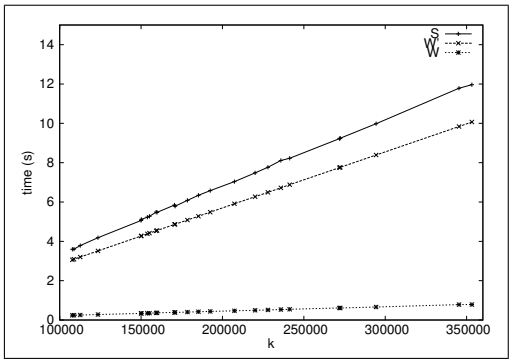


Figure 3. Running times of S, W' and W using the parameters shown in Table 3.

gorithms: it was more than 10 times faster than both W' and S. Furthermore, S was almost as fast as W' although it performed twenty separate searches for each query.

4. CONCLUSIONS

In this paper we presented a new time-warp invariant polyphonic music search algorithm that works in $O(nm)$ time. Unlike the earlier algorithms [4, 6] with time complexities $O(n \log nmw)$ and $O(nmw)$, the running time of our algorithm does not depend on the window size w, and it can be used with arbitrarily large windows.

In addition, our algorithm can also be used for time-scale invariant search. We introduced a new approximate time-scale invariant schema which allows limited jittering within the scaled note durations. In this case, the time complexity of our algorithm is $O(nms)$ where s is the number of possible scalings. In typical queries s is small.

We used our algorithms to search for musical themes in Tchaikovsky's symphonies. We estimated the note events using amplitudes of frequencies in the audio files. Considering the challenging nature of orchestral music input, the results of the algorithms were promising, and the experiment also showed that the modified dynamic programming algorithm is efficient in practice.

Our future plan is to further develop the database construction. Limiting the number of notes in the database could make the search both more accurate and more efficient. However, within the context of a frame, it is difficult to decide which note pitches should be included; in our experiment, the best results were achieved when all pitches were included. Therefore, another approach would be to remove entire frames which are probably not needed in the search.

5. ACKNOWLEDGEMENTS

This work has been supported by the Helsinki Doctoral Programme in Computer Science and the Academy of Finland (grant number 118653).

6. REFERENCES

[1] H. Barlow and S. Morgenstern: *A Dictionary of Musical Themes*, Crown Publishers, New York, 1948.

[2] A. Duda, A. Nürnberger and S. Stober: "Towards query by singing/humming on audio databases," *Proceedings of the 8th International Conference on Music Information Retrieval*, 331–334, 2007.

[3] A. Ghias et al: "Query by humming: musical information retrieval in an audio database," *Proceedings of ACM Multimedia 95*, 231–236, 1995.

[4] M. Laitinen and K. Lemström: "Dynamic programming in transposition and time-warp invariant polyphonic content-based music retrieval," *Proceedings of the 12th International Society for Music Information Retrieval Conference*, 369–374, 2011.

[5] K. Lemström: "Towards more robust geometric content-based music retrieval," *Proceedings of the 11th International Society for Music Information Retrieval Conference*, 577–582, 2011.

[6] K. Lemström and M. Laitinen: "Transposition and time-warp invariant geometric music retrieval algorithms," *Proceedings of the 3rd International Workshop on Advances in Music Information Research*, 1–6, 2011.

[7] A. Lubiw and L. Tanur: "Pattern matching in polyphonic music as a weighted geometric translation problem," *Proceedings of the 5th International Society for Music Information Retrieval Conference*, 289–296, 2004.

[8] D. Meredith, G. Wiggins and K. Lemström: "Pattern induction and matching in polyphonic music and other multi-dimensional datasets," *Proceedings of the 5th World Multi-Conference on Systemics, Cybernetics and Informatics*, 61–66, 2001.

[9] G. Poliner et al: "Melody transcription from music audio: approaches and evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4), 1247–1256, 2007.

[10] C.A. Romming and E. Selfridge-Field: "Algorithms for polyphonic music retrieval: The hausdorff metric and geometric hashing," *Proceedings of the 8th International Society for Music Information Retrieval Conference*, 457–462, 2007.

[11] M. Ryynänen and A. Klapuri: "Query by humming of midi and audio using locality sensitive hashing," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2249–2252, 2008.

[12] J. Salamon, J. Serrà and E. Gómez: "Tonal representations for music retrieval: from version identification to query-by-humming," *International Journal of Multimedia Information Retrieval*, 2(1), 45–58, 2013

[13] E. Ukkonen, K. Lemström and V. Mäkinen: "Geometric algorithms for transposition invariant content-based music retrieval," *Proceedings of the 4th International Society for Music Information Retrieval Conference*, 193–199, 2003.

TOWARDS LIGHT-WEIGHT, REAL-TIME-CAPABLE SINGING VOICE DETECTION

Bernhard Lehner, Reinhard Sonnleitner, Gerhard Widmer
Department of Computational Perception
Johannes Kepler University of Linz
k0357205@students.jku.at, {reinhard.sonnleitner,gerhard.widmer}@jku.at

ABSTRACT

We present a study that indicates that singing voice detection – the problem of identifying those parts of a polyphonic audio recording where one or several persons sing(s) – can be realised with substantially fewer (and less expensive) features than used in current state-of-the-art methods.

Essentially, we show that MFCCs alone, if appropriately optimised and used with a suitable classifier, are sufficient to achieve detection results that seem on par with the state of the art – at least as far as this can be ascertained by direct, fair comparisons to existing systems. To make this comparison, we select three relevant publications from the literature where publicly accessible training/test data were used, and where the experimental setup is described in enough detail for us to perform fair comparison experiments.

The result of the experiments is that with our simple, optimised MFCC-based classifier we achieve at least comparable identification results, but with (in some cases much) less computational effort, and without any need for extensive lookahead, thus paving the way to on-line, real-time voice detection applications.

1. INTRODUCTION

Identifying the regions in a song where a singing voice is present does not seem to be a difficult task for humans, regardless of the singer's specific voice characteristics, dynamics of articulation, instrumental background, or even the language. However, the automatic classification of vocals remains difficult, to a considerable degree due to the extreme extent of vocal tone diversity. At the same time, automatic singing voice detection would be extremely useful for many applications such as audio segmentation and indexing, language detection, singer recognition, vocal extraction, query-by-lyrics, real-time tracking and synchronisation, etc.

Consequently, there has been a lot of research recently into this problem. A multitude of diverse audio features have been proposed, along with many (and sometimes complex) detection methods. In extensive experimental work, where we tried to reproduce some of this work and determine what the most useful features are, we were surprised to find that, when we invested a lot of effort into searching for optimal parametrisations of the features, in the end simple MFCCs always turned out to be at least as good as larger and more complex sets of features.

That is the starting point for this paper, in which we will demonstrate that singing voice detection of state-of-the-art quality can be done in a very light-weight way, using only appropriately parametrised MFCCs. We will ascertain this by comparing our very simple method to three selected methods from the literature where publicly accessible training/test data were used, and where the experimental setup is described in enough detail for us to perform fair comparison experiments.

The result of the experiments is that with our simple, optimised MFCC-based classifier we achieve at least comparable identification results, but with (in some cases much) less computational effort, and without any need for extensive lookahead, thus paving the way to on-line, real-time voice detection applications.

The paper is structured as follows: Section 2 gives an overview of previous work on singing voice detection and on publicly available corpora for this task, and motivates the choice of state-of-the-art methods that we choose to compare our method to. Section 3 presents our own simple method and describes how we experimentally optimised the MFCC and classifier parameters. Section 4 compares our method to the other methods selected above, and Section 5 outlines what we consider to be the most important direction for further improvement.

2. PREVIOUS WORK AND AVAILABLE DATA

We start with a brief overview of selected existing methods, and of publicly available data corpora. Finally, we identify the methods we chose to compare our results to, along with the reasons why we chose those methods.

2.1 Problem statement

The signal consists only of music and the problem is to detect the presence of singing voice therein. Hence, no discrimination of normal speech and singing voice is done,

in contrast to the speech/music discrimination task done by Chou and Gu in [4].

2.2 Previous Work on Singing Voice Detection

In [21], Rocamora and Herrera compared MFCCs, Perceptually derived LPC (PLPC), Log Frequency Power Coefficients (LFPC) and Harmonic Coefficient (HC). Spectral features often utilised for instruments classification were also combined into one vector, namely Centroid, Roll-off, Flux, Skewness, Kurtosis, and Flatness. Additionally, pitch as the only non-spectral feature, extracted with the monophonic f0-estimator YIN [5] was used. Several empirically motivated post-processing strategies were also implemented. In their setting, MFCCs are the most suitable features, and an SVM the best performing classifier for singing voice detection. Although they tried to combine different descriptors, this did not improve the classification performance of 78.5% accuracy on a test set of 46 manually annotated songs. Since this material is not publicly available, it is not possible to conduct a fair comparison.

In [16], HA-LFPCs (harmonic attenuated LFPC) are proposed for singing voice detection. For the construction of the attenuation filter, the key determination technique from [23] is used. Compared to MFCCs, the LFPCs performed better, with a Multi-Model HMM (taking song structure into account) as classifier (86.7% vs. 81.3% accuracy). The experiments were carried out on 20 unknown popular songs from different artists and time spans, with six songs used for training and the remaining 14 songs for testing. Since this material is not publicly available, it is not possible to conduct a fair comparison.

Li and Wang [14] used a singing voice detection stage for the separation of singing voice from instrumental accompaniment. They used MFCCs, LPCs, PLPs, and the 4-Hz harmonic coefficient as features. A HMM and the Viterbi algorithm [18] were used to classify clips from five rock and five country songs via a 10-fold CV. Again, this musical material is not publicly available.

Regnier and Peeters' method [20] involves thresholds for vibrato and tremolo to detect singing voice. They reached an f-value of 76.8% for singing voice compared to a (more sophisticated) machine learning approach using MFCCs, SFM, and their first and second derivatives for both features and a GMM as classifier, which yielded 77.4% f-measure. The experiments were conducted on the Jamendo corpus [19], with 63 songs as training set, and 32 songs as the test set. Since the results reported are not as good as those reported in [19], which were also obtained on the Jamendo corpus, we will compare our method to the latter.

Hsu and Jang [11] used GMMs as states in a fully connected HMM and the Viterbi algorithm [18] to decode music signals into the three classes *accompaniment, unvoiced* and *voiced*. They achieved an accuracy of 77.95% with a 39-dimensional feature vector containing 12 MFCCs, the log energy, and their first and second derivatives. They used all 1000 clips of the MIR-1K data set [11] for training and evaluating, divided into two subsets of similar sizes (487 versus 513, recorded by disjoint subjects) for a 2-fold CV. Since the precise split of the data set is not publicly available, it is not possible to conduct a fair comparison.

In [12], Hsu et al. used basically the same setting as in [11], except that they used Harmonic/percussive source separation (HPSS) as a preprocessing step. Additionally, their HMM decodes into just vocal and nonvocal sections. The usage of the preprocessed signal showed significant improvement compared to the raw signal, especially in lower SNR levels. For an -5, 0, and 5 dB SNR they reached accuracies of $\sim 80\%$, $\sim 85\%$, and $\sim 89\%$ respectively.

The three methods of Vembu and Baumann [25], Mauch et al. [15] and Ramona et al. [19], to which we will compare our own method, will be described in more detail in 3.4, 4.2, and 4.1 respectively. The reasons why we chose them are explained in Section 2.4.

2.3 Publicly Available Corpora

As outlined above, numerous very different approaches have been taken to the problem of singing voice detection. Unfortunately, due to a lack of commonly used corpora, the results are often not directly comparable. To our knowledge, there are three corpora along with vocal activity annotations publicly available:

1. **RWC Music Database: Popular Music (RWC-MDB-P-2001):** 100 songs released by Goto et al. [8], with singing voice annotations provided by Mauch et al. [15]. Along with the annotations a novel method was introduced which will be described in more detail in Section 4.2.

2. **Jamendo Corpus:** 93 copyright-free songs from the Jamendo music sharing website [9], collected and annotated by Ramona et al. [19]. Also used for singing voice detection by Regnier and Peeters in [20].

3. **MIR-1K Corpus:** 1000 song clips taken from 110 karaoke songs, and released by Hsu and Jang in [11]. The songs were recorded in their lab, and sung by 8 females and 11 males. Also used for singing voice detection by Hsu et al. in [12].

2.4 Algorithms Selected for Comparison

Eventually, we selected three different methods as benchmarks for our proposed method.

2.4.1 Vembu and Baumann [25]

This method is relatively simple and achieved remarkable results on an unknown test set. The method is described in such detail that it can be re-implemented. It will be described in more detail in subsection 3.4. Thus, our re-implementation of it will be used as a "baseline" to compare our simple optimised MFCC-only classifier to (see Section 3.4). We will show that MFCCs alone can achieve better performance, if appropriately parametrised.

2.4.2 Ramona et al. [19]

For this method, a very large and diverse set of features is used. They precisely report (also song-wise) results on a publicly available corpus (Jamendo) with exact information regarding which pieces were used for training and evaluation. This allows for the most fair and precise comparison. The method will be described in more detail in subsection 4.1.

2.4.3 Mauch et al. [15]

This is one of the most recent publications on this topic. The authors report excellent results on (parts of) the publicly available RWC corpus, with a rather complex procedure. This method will be described in more detail in subsection 4.2.

3. A SIMPLE MFCC-BASED RECOGNISER

In this section, we describe how we designed and optimised our simple singing voice detector using solely MFCC features. The optimisation was done in three phases, and we will show, for each phase, how the improvements compare to our "baseline" algorithm from Vembu and Baumann [25]. This will give a first impression of results achievable with MFCCs alone.

3.1 Classification Setting and Basic Features

Our classification setting is as follows. The units of audio to be classified are frames of 200 ms duration. Thus, we have 5 classifications per second of audio. The actual window over which the features for classifying a frame F are computed, may be larger than that. We will call this the *observation window*. It will always be placed symmetrically around a classification frame F. We use the VOICEBOX toolbox [3] to extract a D-dimensional MFCC feature vector from the observation window. As the standard parameters we used 30 triangle shaped filters and extracted 13 coefficients, without the 0th coefficient.

For the parameter optimisation phase, we used a set of 75 annotated songs by 75 different artists, which come from a different source than the corpora we will use for the comparison experiments below. All songs were unified, i.e. downsampled to 22kHz and converted to mono. Approximately 52% of the frames are annotated as vocal, and the amount of pure singing, i.e. without instrumental accompaniment, is negligible. All optimisation decisions in the following are based on 15-fold cross validation (CV) experiments, where the data set was randomly split into 15 subsets of 5 songs each.

3.2 Classifier and Post-Processing

Among the most popular classifiers for singing voice detection are Gaussian Mixture Models (GMM), Support Vector Machines (SVM) and Multi-layer perceptron neural networks (MLP). Random forests [2] have proven to deliver good results in other contexts, for instance in speech detection in [24] or music detection in [22]. Compared to SVMs and MLPs they perform much faster in both the training and testing phase. Also, there is no need to determine an appropriate kernel function in order to obtain the best performance. Thus, we chose to use a random forest as classifier, and the implementation from WEKA [10].

For post-processing (smoothing of the prediction sequence), a simple median filter with a window length of seven frames (1.4s) was found to give the best trade-off between complexity and accuracy.

3.3 Optimising the MFCC Features

In this subsection we describe the task of the parameter optimisation, which was done in three phases:

3.3.1 Phase I:

The number of coefficients as well as the size of the filterbank was optimised. The best results were achieved with 30 coefficients (including the 0th) and a filterbank with 30 triangular shaped filters.

3.3.2 Phase II:

The length of the observation window was optimised. In all cases, the observation window, when it was larger than the current classification frame, was placed symmetrically around the frame. Best results were achieved with a total window of 800ms around the 200ms center frame. The relatively long observation window has the effect that the contribution of percussive components (which are only active for a short period of time) to the spectrum is diminished. Before the last phase was conducted, the first derivatives (deltas) of the MFCCs were also added to the feature vector.

3.3.3 Phase III:

The parameters of the classifier were optimised. A good trade-off between computational complexity and the results was found to be a random forest with 128 trees, each with five attributes. Additionally, the threshold for the vocal class was raised to 55% to reduce some of the false positives.

3.4 Comparison to Vembu and Baumann

Vembu and Baumann [25] extracted 13 MFCCs, 39 PLPs, and 12 LFPCs, resulting in a feature vector with 64 elements. They achieved the best results with all features combined and using a SVM as classifier (93.47% accuracy on an unknown data set). Additionally, they provide the parameters of the optimised RBF kernel they used ($C = 2^8$, $\sigma = 2^2$). They do not mention any post-processing. Since there is enough information available to implement this method, it will be used as a baseline for further comparison. Although theirs is an online algorithm, it turns out that training and testing is extremely time consuming (about 75 times slower than with our optimised random forest).

Table 1 shows the results of the 15-fold CV on our data set (see Section 3.1), at different stages of the optimisation

	MFCC	I	II	PROP	VB
acc [%]	69.14	74.51	78.74	82.36	77.16
recall	0.727	0.783	0.834	0.883	0.819
precision	0.712	0.757	0.788	0.810	0.774
f-measure	0.719	0.770	0.810	0.845	0.796

Table 1. Results of the parameter optimisation compared to the baseline method VB. The columns are as follows: MFCC: "standard" (unoptimised) MFCCs. I: after optimisation stage I (number of MFCCs; filterbank). II: after optimisation stage II (observation window; delta MFCCs). PROP: proposed method after optimisation III (optimised random forest; median filter). VB: Vembu & Baumann trained and tested in the same way. Recall, precision, and f-measure relate to our class of interest, *vocals*.

procedure, along with the results of the VB method (computed on exactly the same data splits). To illustrate the effectiveness of every optimisation stage, we begin with the results achieved with standard MFCCs (see Section 3.1) and a standard random forest classifier (column MFCC). After the first optimisation stage (I), where we use 30 coefficients instead of just 13, there is an improvement in accuracy of more than 5 percentage points. The second optimisation regarding the observation time (II) yields an improvement of almost 10 points compared to the standard MFCCs, and already better results than the baseline algorithm of Vembu and Baumann [25] (VB). The final proposed method (PROP.) with the optimised random forest classifier and median-filter post-processing, reaches an accuracy 13 percentage points higher than standard MFCCs, and more than 5 points better than the VB method.

4. COMPARISON TO STATE-OF-THE-ART METHODS

In this section our proposed simple method is compared to two other methods. In Section 2.4 we already explained the motivation behind the selection of the algorithms; now we explain them in more detail.

To recapitulate, our proposed method uses just the optimised long-term MFCCs (800ms, 30 coefficients incl. the 0th, and a filterbank with 30 triangular shaped filters) along with their first derivatives. There is no pre-processing involved, and a simple median filter over seven frames (1.4sec) is used to smooth out the predictions of a random forest classifier (128 trees with five features each).

4.1 Ramona et al.

Ramona et al. [19] use an SVM classifier with a combination of the most diverse set of features compared to the other methods discussed in this paper. These include MFCCs, LPCs, ZCR, sharpness, spread, f0 and aperiodicity measure extracted with the monophonic YIN library [5]. Furthermore, short-scale frames contain spectral descriptors like centroid, width, asymmetry, slope, decreasing, flux, and similar temporal statistical moments. Additionally, long-scale frames contain features that do not represent an instantaneous characteristic. Those include (again) the ZCR, tremolo and granularity for the frequencies 4-8Hz and 10-40Hz, and some temporal statistical moments. Those features add up to a vector with 116 components.

	Ramona et al.		PROP		VB	
	Acc%	F%	Acc%	F%	Acc%	F%
03 - Say me Good Bye	80.1	85.8	91.4	83.6	90.6	82.7
03 - School	84.3	87.3	84.8	86.6	71.5	77.8
03 - Si Dieu	76.4	80.7	87.2	89.4	76.2	66.7
03 - Une charogne	85.3	91.7	89.8	93.5	78.8	85.9
03 - castaway	79.0	87.3	71.3	80.5	73.0	80.0
04 - Believe	80.0	88.5	94.1	95.6	83.0	87.2
04 - Healing Luna	85.5	81.6	87.8	84.4	72.7	70.8
04 - Inside	83.3	68.2	79.4	66.0	75.3	58.0
04 - You are	87.0	91.9	87.9	90.6	74.4	77.7
05 - 05 LIrlandaise	57.7	64.2	65.0	68.6	61.7	60.5
05 - 16 ans	91.5	84.8	87.3	79.8	70.8	60.3
05 - 2003-Circons[...]	87.6	88.2	75.5	77.7	79.8	79.6
05 - A Poings Fermes	93.7	92.2	89.7	83.0	86.9	81.2
05 - Crepuscule	85.2	88.8	80.1	83.6	76.8	80.0
05 - Dance	77.0	83.2	84.1	88.7	75.7	82.2
05 - Elles disent	71.8	78.7	84.4	87.4	69.6	77.0
ALL	82.2	84.3	84.8	84.6	77.4	76.9

Table 2. Results of the proposed method on the Jamendo corpus, compared to those of Ramona et al. in [19] and Vembu and Baumann's method trained and tested in the same way.

Afterwards, the dimensionality is reduced to d=40 with the IRMFSP algorithm [17], leaving only the most discriminating features. A silence detection is applied as a pre-processing step. Finally, a HMM and the Viterbi algorithm are used for post-processing the SVM output, and instrumental segments shorter than 0.5s are discarded.

The authors report 82.2% accuracy on a precisely described split of the Jamendo corpus, with a training set consisting of 63 given songs, and validation and test sets of 16 songs each. Thus, a fair comparison of the results is possible.

In Table 2, the results of Ramona et al. are compared to those of the proposed method. All in all, better results regarding both accuracy and f-measure are achieved with the proposed method (82.2% vs. 84.8% accuracy). Nevertheless, there are some songs that get better classified with the method of Ramona et al.; the biggest difference is with the song *05 - 2003-Circons[...]* (87.6% vs. 75.5% accuracy). It would be interesting to have a feature with which we could determine the better suited method for a specific song, or even a shorter segment.

4.2 Mauch et al.

Mauch et al. [15] utilise four features in total, among them MFCCs. Additionally, they use Goto's polyphonic fundamental frequency(f0)-estimator PreFEst [7] to isolate the predominant melody. They propose three novel features which are based on it:

Pitch fluctuation, which is basically the frame-wise standard deviation of intra-semitone f0 differences. First, the estimated f0 is mapped to pitch space. Afterwards, these

estimations are shifted based on a song-wide inferred tuning. As a last step, the frequency differences are calculated, and the frame-wise Hamming-weighted standard deviation of those differences yields the pitch fluctuation. Since a song-wide lookahead is necessary to infer its tuning, this method is not an online algorithm. Pitch fluctuation is found to be the most salient feature for singing voice detection.

In addition to the MFCCs of the signal as it is, the authors also introduce *MFCCs of the re-synthesised predominant voice* to capture its timbre. The re-synthesis employs sinusoidal modelling based on the predominant melody as well as the estimated amplitudes of its harmonics as described in [6].

The *normalised amplitude of harmonic partials* is also extracted from the predominant voice, and is considered to add information on another dimension of timbre which is not provided by MFCCs. It is a vector-shaped feature (d=12), and calculated by normalising the estimated harmonic amplitudes according to the Euclidean norm.

A SVM-HMM [1,13] is used as classifier. Additionally, segments shorter than 0.5s are merged with the preceding regions.

The best result (87.2% accuracy) was achieved with all four features combined, employing a 5-fold CV on a 102 song data set that is composed of 90 songs from the RWC music database [8] (exactly which 90 of the 100 is unknown to us and could, unfortunately, not be found out), and 12 additional (also unknown) songs. Since we had only access to the 100 song RWC music database, our results are only comparable to a certain extent. Nevertheless, to allow for the best comparison possible, we matched their decision frequency of one feature instance every 100ms and utilised a 5-fold CV.

In Table 3, the results of Mauch et al. are compared to those of the proposed method. To illustrate the difference of the data set used by Mauch et al. to the original RWC data set, we also give the results achieved with standard MFCCs (see Section 3.1). Additionally, to reveal the amount of vocals, we give the results of the mode, i.e. the proportion of the majority class (which is *vocals*). As can be seen, there is a difference of 5 percentage points regarding the vocal content, which indicates a limited comparability.

All in all, our proposed method performs not much worse than the method of Mauch et al. (85.9% vs. 87.2% accuracy). This difference virtually disappears when we apply a more complex post-processing strategy involving a HMM and the Viterbi algorithm (row PROP+).

Mauch	**accuracy**	**precision**	**recall**	**f-measure**
MODE	0.654	0.654	1.000	0.791
MFCC	0.738	0.739	0.926	0.822
FMRH	0.872	0.887	0.921	0.904
Proposed	**accuracy**	**precision**	**recall**	**f-measure**
MODE	0.604	0.604	1.000	0.753
MFCC	0.718	0.764	0.771	0.767
VB	0.813	0.827	0.808	0.818
PROP	0.859	0.858	0.918	0.887
PROP+	0.868	0.879	0.906	0.892

Table 3. The results of our proposed method compared to the methods of Mauch et al. and Vembu and Baumann. Along with the methods the class distribution in the respective test set is given (row MODE – the overall proportion of vocals), as well as the results achieved with the standard MFCCs, and Vembu and Baumann's Method (row VB). Clearly, even though the majority of the data we used is the same as used by Mauch et al., there are differences regarding the content of vocals, which makes a fair comparison unfeasible. The results PROP+ are achieved with a post-processing involving the Viterbi algorithm.

5. CONCLUSION AND FUTURE WORK

This paper has proposed an extremely simple method to detect the presence of singing voice in mixed audio signals. By comparing the results to those of three well selected algorithms, we could show that regarding the features, appropriately parametrised MFCCs along with their first derivatives are sufficient to achieve results as good as those of sometimes much more complicated state-of-the-art systems. Our method is simple, fast, and requires no look-ahead, making it a good candidate for on-line, real-time singing voice detection applications.

Our main goal for further improvement is *precision*, that is, a reduction of the number of *false positives*. A detailed inspection of the results of our classifier has shown that instruments mistaken for vocals have the biggest negative impact on the results. This is especially true for string instruments like electric guitars, which can mimic the temporal as well as the timbral characteristics of vocals. Certain effects commonly used to enhance the sound or extend the expressiveness of guitars like chorus, flanger, and wah-wah are responsible for this. Unfortunately, experiments reported in [21] indicate that features often utilised for speech/music discrimination like harmonic coefficient [4] are not able to distinguish between highly harmonic instruments and vocals. Therefore, it would be beneficial to develop a method that is less sensitive to differences between singers' specific voice characteristics, while maintaining good discriminative properties for instruments that resemble vocals.

6. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Fund FWF under grants TRP307-N23 and Z159 (Wittgenstein Award).

7. REFERENCES

[1] Y. Altun, I. Tsochantaridis, T. Hofmann, et al. "Hidden Markov support vector machines". In *Proceedings of the 20th International Conference on Machine Learning (ICML 2003)*, volume 20, 2003.

[2] L. Breiman. "Random forests". *Machine learning*, 45(1):5–32, 2001.

[3] M. Brookes. "Voicebox: Speech Processing Toolbox for Matlab". Website, 1999. Available online at `http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html`; visited on March 4th 2013.

[4] W. Chou and L. Gu. "Robust singing detection in speech/music discriminator design". In *Proceedings of the 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2001*, volume 2, pages 865–868. IEEE, 2001.

[5] A. De Cheveigné and H. Kawahara. "YIN, a fundamental frequency estimator for speech and music". *The Journal of the Acoustical Society of America*, 111:1917–1930, 2002.

[6] H. Fujihara, M. Goto, J. Ogata, K. Komatani, T. Ogata, and H. G. Okuno. "Automatic synchronization between lyrics and music CD recordings based on Viterbi alignment of segregated vocal signals". In *Proceedings of the Eighth IEEE International Symposium on Multimedia, ISM 2006*, pages 257–264. IEEE, 2006.

[7] 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, 2004.

[8] 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 2002)*, volume 2, pages 287–288, 2002.

[9] P. Grard, L. Kratz, and S. Zimmer. "Jamendo, open your ears". Website, 2005. Available online at `http://www.jamendo.com`; visited on March 18th 2013.

[10] M. Hall, F. Eibe, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. "The WEKA data mining software: an update". *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[11] C-L. Hsu and J-S. R. Jang. "On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset". *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2):310–319, 2010.

[12] C-L. Hsu, DL. Wang, J-S. 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*, 20(5):1482–1491, 2012.

[13] T. Joachims, T. Finley, and C. J. Yu. "Cutting-plane training of structural SVMs". *Machine Learning*, 77(1):27–59, 2009.

[14] Y. Li and DL. Wang. "Separation of singing voice from music accompaniment for monaural recordings". *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1475–1487, 2007.

[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 *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, pages 233–238, 2011.

[16] T. L. Nwe, A. Shenoy, and Y. Wang. "Singing voice detection in popular music". In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 324–327. ACM, 2004.

[17] G. Peeters. "Automatic Classification of Large Musical Instrument Databases Using Hierarchical Classifiers with Inertia Ratio Maximization". In *115th AES Convention*, 2003.

[18] L. R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". *Proceedings of the IEEE*, 77(2):257–286, 1989.

[19] M. Ramona, G. Richard, and B. David. "Vocal detection in music with support vector machines". In *Proceedings of the 2008 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2008*, pages 1885–1888. IEEE, 2008.

[20] L. Regnier and G. Peeters. "Singing voice detection in music tracks using direct voice vibrato detection". In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2009*, pages 1685–1688. IEEE, 2009.

[21] M. Rocamora and P. Herrera. "Comparing audio descriptors for singing voice detection in music audio files". In *Brazilian Symposium on Computer Music, 11th. San Pablo, Brazil*, volume 26, page 27, 2007.

[22] K. Seyerlehner, T. Pohle, M. Schedl, and G. Widmer. "Automatic music detection in television productions". In *Proceedings of the 10th International Conference on Digital Audio Effects (DAFx'07)*, 2007.

[23] A. Shenoy, R. Mohapatra, and Y. Wang. "Key determination of acoustic musical signals". In *Proceedings of the 2004 IEEE Conference on Multimedia and Expo, ICME 2004*, volume 3, pages 1771–1774. IEEE, 2004.

[24] R. Sonnleitner, B. Niedermayer, G. Widmer, and J. Schlüter. "A Simple And Effective Spectral Feature For Speech Detection In Mixed Audio Signals". In *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx'12)*, 2012.

[25] S. Vembu and S. Baumann. "Separation of vocals from polyphonic audio recordings". In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR 2005)*, volume 5, pages 337–344, 2005.

THE USE OF MELODIC SCALES IN BOLLYWOOD MUSIC: AN EMPIRICAL STUDY

Monojit Choudhury **Ranjita Bhagwan** **Kalika Bali**
Microsoft Research Lab India
{monojitc, bhagwan, kalikab}@microsoft.com

ABSTRACT

Hindi film music, which is commonly referred to as "Bollywood" music, is one of the most popular forms of music in the world today. One of the reasons for its popularity has been the willingness of Bollywood composers to adopt and be influenced by various musical forms including Western pop, jazz, rock, and classical music. However, till date, we are unaware of any systematic quantitative analysis of how this genre has changed and evolved over the years since its inception in the early 20^{th} century. In this paper, we study the evolution of Bollywood music with respect to the use of *melodic scales*. We analyse songs composed over seven decades using a database of top-lists, which reveals many interesting patterns. We also analyze the scale usage patterns in the music of some of the most popular composers, which clearly brings out certain idiosyncrasies and preferences of each of them.

1. INTRODUCTION

The Mumbai based Hindi language film industry, which is popularly referred to as *Bollywood* [1], produces more than 800 movies every year [2]. Almost all Bollywood movies feature several songs that are very popular in India, and have often been termed as the heartline of Indian popular culture [1, 11]. In fact, Bollywood songs are one of the most searched items on the Web from India [3]. Over decades, this music has influenced lives and cultures not just in India, but across all of Asia, Africa, Eastern Europe, and more recently, North America [7]. Several scholars in the past [2] have attributed this universal popularity of Bollywood music to its great ability in assimilating various styles of music from around the world and churning out compositions of global appeal. The cultural history of Bollywood, the emergence of musical styles, genres and sub-genres, and corresponding ethnomusicological aspects have been well studied by social scientists [5, 9, 10]. However, as far as we know, till date there are no quantitative and statistical analysis of musical styles of Bollywood, except for a recent study that analyzes Bollywood song lyrics through computational linguistic techniques [3].

In this paper, we present a quantitative study of the relative usage and popularity of various *melodic scales* in Bollywood over a period of seven decades. We analyzed the scales used in 310 songs from 1953 to the present day, all of which have been in the top 50 hit list for the year they appeared in. Our results show that some scales have waned in popularity over the years, while some other scales have always dominated the mix. We also identified five influential music composers of Bollywood from different time periods and analyzed the usage of musical scales in their compositions. Again we noticed several interesting composer specific trends. For instance, some composers like Allahrakka Rahman and Rahul Dev Burman, show great diversity in the usage of musical scales in their compositions, while others such as Shankar-Jaikishan or Pritam Chakraborty preferred to exploit the melodic patterns of a few popular scales.

The study of melodic scales in Bollywood music is particularly interesting because since its inception in the early 20^{th} century and until the 1960s, Hindi film songs were almost exclusively based on or inspired by Indian Classical Music (ICM). A characteristic feature of ICM is its use of a large number of scales. Hence, we expect Hindi film music to exhibit a similar diversity of scale usage than that found in the popular music of other cultures. However, the decreasing influence of ICM, and the gradually increasing influence of folk, rock, pop and blues on this genre can significantly alter the scale usage pattern over time. Thus, by studying the evolution of scale usage patterns in Bollywood, we can, in essence, objectively characterize the influence of various musical cultures as well as individual composers on Bollywood.

2. MELODIC SCALE OF A SONG

2.1 Definition of a Scale

Bollywood music, along the lines of ICM, uses the standard twelve note (in an octave) Western *chromatic scale*. Any given composition uses only a subset of these twelve notes, which can be loosely referred to as the *melodic scale* of the composition. This is a *relative scale* or *mode*-based system, where the intervals between the adjacent notes are fixed, but not the notes themselves. Once the *tonic*, the

[1] http://en.wikipedia.org/wiki/Bollywood
[2] http://geography.about.com/od/culturalgeography/a/bollywood.htm
[3] http://www.google.com/intl/en/press/zeitgeist2010/regions/in.html

NICM	*Sa*	*kom. Re*	*Re*	*kom. Ga*	*Ga*	*Ma*	*teevra Ma*	*Pa*	*kom. Dha*	*Dha*	*kom. Ni*	*Ni*
Notation	**S**	**r**	**R**	**g**	**G**	**m**	**M**	**P**	**d**	**D**	**n**	**N**
Western	C	$C\#$ or D_b	D	$D\#$ or E_b	E	F	$F\#$ or G_b	G	$G\#$ or A_b	A	$A\#$ or B_b	B

Table 1. Names and notations of the notes in NICM along with their Western counterparts. *kom.* stands for *komal* meaning a flat note. *teevra* is a sharp note.

first note of the octave referred to as *Sa* in Indian musical traditions (equivalent to *Do* in Solfaggio), is set to a specific note (say C or A_b) the rest of the notes in the scale gets automatically defined. Table 1 introduces the names of the notes used in Indian music and the notation that we will use in this paper along with the corresponding note in Western music when the tonic is on C.

Like Western music, the most common scale types in ICM are *heptatonic*, i.e., consist of seven notes. Therefore, the two dominant classical musical traditions prevalent in India today, the *Hindustani* or the North Indian Classical Music (NICM) and the *Carnatic* or the South Indian Classical Music (SICM), deem the heptatonic scales to be fundamental; all other scales – *pentatonic, hexatonic* and (rarely) *octatonic* are considered to be derived from a parent heptatonic scale. Musicologists in both the traditions [4,6] have extensively studied and proposed the defining features of the heptatonic scales used in the respective systems. A detailed discussion on these theories is beyond the scope of this paper, though it suffices to mention here that in NICM tradition, from which Bollywood music has derived its main genetic material, a basic heptatonic scale can have only one of the two Re's, Ga's, Ma's, Dha's and Ni's. This leads to 32 possible fundamental scales, of which only ten are traditionally identified as important or fundamental for deriving other scales and *ragas*[4] [4]. These scales are referred to as *thaats*. We shall refer to these scales by their corresponding *thaat* names. However, there are some more scales that are presently used in NICM (often borrowed from popular ragas, e.g., *Charukeshi* and *Kirwani*, in SICM system). Table 2 lists all these popular heptatonic scales used in NICM. We shall refer to these scales by the name of the *thaat* if it is one of the 10 thaats described by Bhatkhande, else we will refer to them by the name of a popular *raga* based on this scale. *One should note that a raga and a scale are not equivalent, it is just a naming convention that we will follow in this paper, and henceforth all the raga names should be interpreted as the corresponding scale (set of notes) used by the raga.*

2.2 Identifying the Scale of a Song

Almost all the Bollywood songs are composed in the scales mentioned in Table 2, though some of the compositions might use a pentatonic or hexatonic scale derived from one of these heptatonic scales, and sometimes *accidentals* (notes that do not belong to the parent scale) might be employed at a certain point in the composition. Since the (staff) notations are not readily available for Bollywood songs[5], we resort to music experts to identify the notes and hence the scale of a song. When the set of notes used can be easily identified as a heptatonic scale (which almost always maps to one of the scales listed in Table 2), we unambiguously mark the scale of the song with its corresponding name. Whenever a note and its flatter or sharper version are used in the same song (less than 10% in our dataset), one of the notes is marked as accidental based on the expert's intuition. This often involves looking at the frequency of usage, as the accidentals are used much less frequently than the other note which is a part of the scale. Once the accidentals are so identified, it is easy to map the rest of the notes into one of the heptatonic scales.

There are a substantial number of Bollywood compositions that use scales with fewer than seven notes. For instance, the Major Pentatonic scale – (**S R G P D**), that is used in ragas *Bhupali*, *Deshkar* etc., has been used in many Bollywood songs. This scale can be thought to be a derivative of *Kalyan*, *Bilawal* or *Khamaj*. Due to this ambiguity, we do not map pentatonic and hexatonic scales to a parent heptatonic scale; rather, we refer to those scales by the name of a popular raga that uses the scale. Some of the common pentatonic and hexatonic scales that we have encountered in Bollywood music (along with the names we

Scale	NICM name	Western name
S R G m P D N	*Bilawal**	Major Diatonic
S R g m P D N	*Patdeep*	Melodic Minor
S r G m P d N	*Bhairav**	Major Gypsy
S R g m P d N	*Kirwani*	Harmonic Minor
S R G m P D n	*Khamaj**	Mixolydian
S R g m P D n	*Kafi**	Dorian
S R G m P d n	*Charukeshi*	Major Minor
S R g m P d n	*Asavari**	Natural Minor
S r g m P d n	*Bhairavi**	Phrygian
S R G M P D N	*Kalyan**	Lydian
S r G M P D N	*Marwa**	Not known
S r G M P d N	*Purvi**	Chromatic Hypolydian
S r g M P d N	*Todi**	Chromatic Lydian Inverse

Table 2. List of heptatonic scales used in NICM. The *Thaat* names are marked with *.

[4] A *raga* is based on an ascending and descending scale, but is characterized using many other features and evades a formal definition. See [4] for a detailed exposition.

[5] This is again a common feature of Indian musical traditions, where improvisations abound in all genres of classical and semi-classical music, and consequently a fixed notation is rarely used to describe the composition.

will refer to them with): **S R G P D** (*Bhupali* or the Major Pentatonic scale), **S R m P n** (*Megh* or the Suspended Pentatonic scale), **S R G P N** (*Hamsadhvani*), **S R g P D** (*Shivaranjani*) and **S R g m P n** (*Nayaki* or the Minor Hexatonic scale).

It is important to note that Bollywood songs are usually structured with a beginning *mukhda* (analagous to a chorus) followed by anywhere between one and four *antaras* (analogous to verses). Sometimes, a song may use different melodic scales in the chorus and the verses. For the purpose of this study, we study the scales used only in the chorus, since this is the most important and remembered part of a song, on which the success or popularity of a song primarily depends.

3. DATA COLLECTION METHOD

In order to study the usage pattern of scales in Bollywood songs and its evolution over the years, we want to build a dataset of songs along with their scales. Information about the year of release of the song and the composer are also necessary because our objective is to do a trend analysis and identify influences of composers on scale usage, if any. Thus, we first identify the songs that we would like to analyze, identify the scales of those songs, and then identify a set of composers who are well represented in our dataset and hence can be meaningfully studied. The dataset is publicly available at: `http://bit.ly/18Edp7Y`

3.1 Selection of Songs

There are no published authentic statistics on the number of Bollywood songs produced to date. Some crude estimates tell us that this number should be between 100,000 and 200,000. Therefore, it is impossible to manually identify the scales for even a small subset, say a randomly selected 10%, of Bollywood songs. We decided to select a subset of popular songs from every year that appeared in the year-specific top-lists. There are two reasons for this choice. First, the top songs represent the choice of the people or what was popular at a particular point of time. Second, top songs probably also have an influence on the composers and their future compositions. We selected songs from *Binaca Geetmala* [6] charts, which published yearly top-lists from 1953 to 1994 and were the authoritative resource at the time. The lists have somewhere between 12 to 40 songs per year with more than 1000 songs in all. We chose two to three years per decade and included the top songs from those years in our dataset. We collected top-lists for 2000s from other online sources [7] .

3.2 Selection of Composers

From the list of selected songs, we identified the composers for whom we already had a substantial number of songs. These composers are (active year range for our dataset is given in parentheses): Shankar-Jaikishan [8] (1953-71), Kalyanji-Anandji [9] (1960-81), R. D. Burman [10] (1971-81), A. R. Rahman [11] (1992-2013, composer and Academy award winner for Slumdog Millionaire) and Pritam Chakraborty [12] (2004-2013). All of these have composed many hit songs over more than a decade and are considered as very influential composers who have introduced new musical styles and genres in Bollywood music [1]. Moreover, they composed during different periods of time. Hence, by studying their scale usage preferences it might be possible to gain interesting insights into the evolution of Bollywood melodies. In order to make the analysis more reliable, we then selectively included several other songs in our dataset that were composed by these five composers and that featured in the yearly top-lists.

Decade	50	60	70	80	90	2000	2011-
#Songs	45	31	76	32	40	51	35

Table 3. Number of songs selected for analysis per decade. Total number of songs is 310.

The final dataset of songs, that was thus created, has 310 songs composed between 1953 and 2013. Table 3 reports the number of songs selected per decade in our dataset.

3.3 Scale Identification

We asked two experts, who have 10+ years of training in NICM and Bollywood music, to independently identify the scales for a randomly selected 50 songs from our dataset according to the rules specified in Sec. 2.2. The experts agreed on the scales for all the 50 songs, which indicates that scales for the songs are unambiguously identifiable, and both the experts were good at it. Hence, finally we asked only one of the two experts to identify the scales for rest of the songs in the dataset. The process takes, on an average, approximately 2 to 5 minutes per song. We intend to make this dataset publicly available for future research.

4. FINDINGS

A dataset of 310 songs spread over seven decades does not provide sufficient information to conduct a sophisticated per-year statistical analysis. Therefore, here we will report overall statistics on the temporal distribution of melodic scales, which reveal several interesting trends. Then we will move beyond gross statistical analysis, and focus on a case-by-case study of several issues that we could observe even from this limited dataset.

4.1 Overall Distribution of Scales

Fig. 1 shows the percentage of songs in the entire dataset that uses a particular melodic scale, represented as a pie-

[6] http://en.wikipedia.org/wiki/Binaca_Geetmala

[7] http://shaileshkapoor.com/2012/01/14/top-50-bollywood-songs-of-2011/, http://www.bollywoodmusicradio.com/announcements/2617-top-40-songs-2005-a.html

[8] http://en.wikipedia.org/wiki/Shankar_Jaikishan

[9] http://en.wikipedia.org/wiki/Kalyanji_Anandji

[10] http://en.wikipedia.org/wiki/R.D._Burman

[11] http://en.wikipedia.org/wiki/A._R._Rahman

[12] http://en.wikipedia.org/wiki/Pritam_Chakraborty

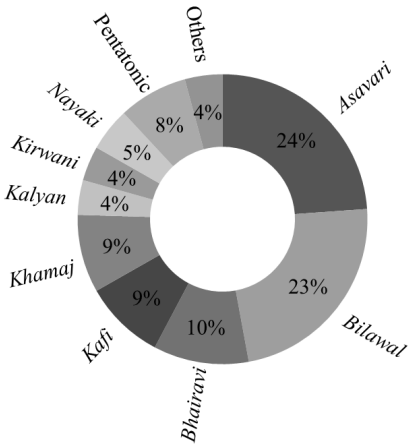


Figure 1. Distribution of Scales in our dataset.

chart. We observe that almost a quarter of the songs are composed in *Asavari* and another quarter in *Bilawal*. This is not entirely surprising because these two heptatonic scales are known to be the most popular major and minor scales across the globe. However, these statistics are indeed surprising if one considers the fact that in NICM as well as Indian folk traditions, though *Bilawal* is used quite often, use of *Asavari* and its related *ragas* is not so common. In fact, *Kalyan* and *Kafi* are by far the more popular scales presently used in NICM rather than *Asavari*.

Bhairavi, *Kafi* and *Khamaj* together cover another quarter of the songs, each being used almost equally frequently. It is surprising to see that only 4% of the songs are composed in *Kalyan*, which is otherwise a very popular scale in NICM. On the other hand, *Nayaki* and *Kirwani*, which are rarely used in NICM, are as well represented as *Kalyan*. Pentatonic scales are quite popular (8%) in Bollywood, of which the more common ones are *Bhupali* (4%), *Shivaranjani* (1%) and *Megh* (1%). We did notice a few cases where the scale used had only 3 or 4 notes, and sometimes a pentatonic or hexatonic scale that is not used in NICM or Western music. Together, these cases account for 10 songs in our dataset.

It is interesting to note that in our dataset we observe no compositions in *Bhairav*, *Marwa* and *Todi*, and only one in *Purvi*. These four scales enjoy the status of a *thaat* in NICM and are very popular in both NICM and SICM. This perhaps can be explained by the fact that these heptatonic scales have less symmetric tetrachords (see [8] for an in-depth discussion) than the six other *thaat* scales that are much more frequently used in Bollywood.

4.2 Temporal Dynamics of the Scales

Is there any significant change in the scale usage pattern over the years? To study the temporal dynamics of the scales, we computed the distribution of the scales for each decade as the fraction of the songs in a given decade that were composed in a particular scale. We observe that the scales can be broadly divided into three categories based on their temporal trends: (a) scales which did not show any drastic change in their usage pattern and have always been popular in Bollywood songs (i.e., always popular); (b) scales which did not show any drastic change in the trends, but were always less commonly used; and (c) scales which show drastic changes in their usage. Fig. 2 shows the trends for these three classes of scales in three different bar-charts. From the top chart one can easily see that the scales *Asavari* and *Bilawal* were always commonly used and together they accounted for half of the songs composed in any decade. The middle chart shows the trends for *Kafi*, *Khamaj* and the pentatonic scales. There are no specific observable trends (except for a drop in the use of *Kafi* as compared to the 1950s) and the minor variations are statistically insignificant. However, it is reasonable to conclude that these scales have always been in use in Bollywood songs, but only sparingly. The third set of scales represented in the lowermost chart shows several interesting trends.

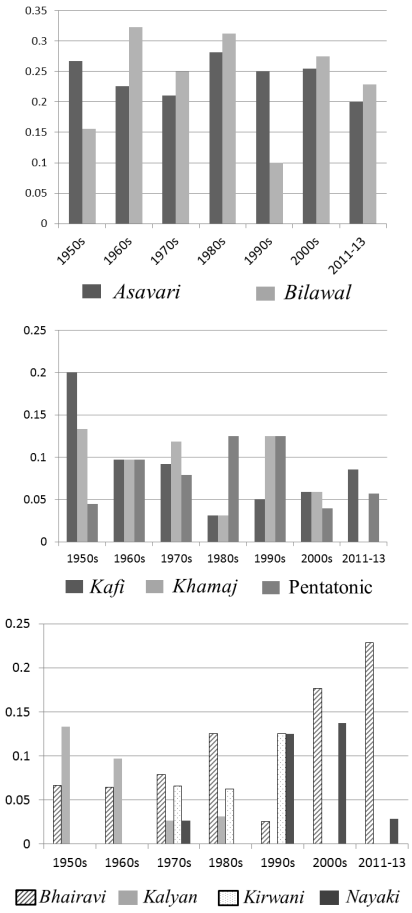


Figure 2. Change in the use of the scales over time.

We observe that the use of *Bhairavi* is steadily gaining popularity over the years, whereas the use of *Kalyan*, which was quite popular in 1950s and 60s, has steadily gone down and there is not a single composition in *Kalyan* in the dataset beyond 1980s. *Kirwani* made its appearance in 1970s and enjoyed some popularity over the next two decades, and then disappeared. Finally, *Nayaki*, which had made a brief appearance in 1970s, gained popularity in the 90s, which sustained over the next decade and beyond. Clearly, these drastic changes in the usage patterns of some

scales cannot be explained away as mere random fluctuations. These could be either due to the influence of specific music composers, cross-cultural influences, or both.

4.3 Composer-specific Trends

Table 4 reports the number of songs by a composer that uses a particular melodic scale. As one would expect, all the five composers have composed a significant number of songs in *Bilawal* and *Asavari*, the two most popular scales used in Bollywood. However, there are a couple of interesting observations that we would like to highlight here.

First, the number of distinct scales used by Kalyanji-Anandji (K-A), R. D. Burman and A. R. Rahman is much more than what we observe for the compositions of Pritam and Shankar-Jaikishan (S-J). In fact, we also noticed that even when using well-known scales like *Asavari* and *Bilawal*, R. D. Burman and A. R. Rahman have extensively used various accidentals. Hence, from the perspective of scale-usage, one can say that some of the famous Bollywood composers were inclined to experiment with scales and introduce new melodic patterns, whereas other equally famous composers focussed their attention on other components of the composition (e.g., the rhythm).

Second, the drastically changing trends for the usage of certain scales is evident from the composer-specific analysis as well. For instance, we notice an increase in the use of *Bhairavi* by Pritam, which partly explains the rise of its popularity in the 2000s and beyond. Similarly, *Nayaki* seems to have been introduced by R. D. Burman and then extensively used by more recent composers like A. R. Rahman and Pritam. On the other hand, *Kafi*, which was used to some extent by S-J and K-A has not been used by R. D. Burman or Pritam. This explains the slight drop in its use, though it is due to Rahman that it still shows up in some compositions.

There are also interesting composer-specific observations such as the strong preference towards *Bilawal* by Pritam or *Khamaj* by A. R. Rahman, which deviates from the Bollywood norms. Likewise, K-A's nearly equal preference given to a large number of scales highlights a unique characteristic of this composer duo. We will discuss the possible causes and repercussions of these findings in the next section.

Scales	S-J	K-A	RDB	ARR	PC
Asavari	9	4	9	8	6
Bilawal	7	3	9	2	12
Khamaj	2	3	6	5	2
Bhairavi		3	4	2	6
Kafi	1	3		4	
Kalyan		1	1		
Kirwani	2	1	3	2	
Nayaki			3	3	5
Other	2	7	3	4	1
Total	23	25	38	30	32
Distinct	6	11	10	10	6

Table 4. Scale usage of five influential composers: S-J – Shankar-Jaikishan, K-A – Kalyanji-Anandji, RDB – R. D. Burman, ARR – A. R. Rahman, PC – Pritam Chakraborty. Total and Distinct respectively refers to the total number of scales analyzed for the composer and number of distinct scales found in those compositions.

5. INTERPRETATIONS AND CONJECTURES

Bollywood is a melting pot of cultures and so its music incorporates numerous influences from musical traditions all over the world. The present study on the use of melodic scales can actually provide various interesting insights about these cultural, and at times individual, influences on the music of Bollywood. Here we discuss some of *our interpretations* of the observed trends in scale usage, and make some conjectures.

Decreasing Influence of ICM. The scale usage pattern of Bollywood differs significantly from that of ICM. The gradually decreasing popularity of *Kalyan* and to some extent *Kafi*, both of which are extremely popular in ICM, clearly indicates that Bollywood is moving away from ICM. On the other hand, we note that the scale usage pattern of the 1950s and 60s (*Khamaj*, *Kafi* and *Kalyan* were used equally and only slightly less than *Bilawal*) follow a trend that one would expect under a strong influence of ICM. Indeed, music by earlier composers such as Naushad, were heavily influenced by ICM [13] . Nevertheless, it is important to note that ICM-based songs are still occasionally used in the Bollywood movies, especially in period films, and we can expect this trend to continue.

Increasing Influence of Western Music. Since 1970s, Western popular music has had a steady influence on Bollywood. R. D. Burman is known for extensive use of ideas from Jazz and Blues in his compositions. His introduction of *Nayaki* – the Minor Hexatonic scale that is popularly used in Blues and other Western music – and subsequent popularity of this scale till the present day is a clear evidence of the increasing influence of the Western music. While *Bilawal* and *Asavari* are popular in ICM, they are prevalent in Western pop music as well. A deeper analysis of the recent songs composed in these scales shows that it is indeed the influence of the latter which has led to the recurrent usage of these scales.

Influence of other Musical Cultures. *Bhairavi* or the Phrygian scale, though popular in ICM, is also prevalent in the folk songs of the Middle East, some East European regions, and the Punjab and Kashmir regions of India. Most Bollywood compositions in *Bhairavi* hardly bear any resemblance to the *raga Bhairavi* or its derivatives. R. D Burman's famous songs of the 1970s and 80s in this scale rather reflect the influence of Middle Eastern folk songs. The music of the Middle East has been and still continues to be a strong source of inspiration for Bollywood composers. R. D. Burman's compositions in *Kirwani* or the Harmonic Minor scale, another popular scale of the region,

[13] http://chandrakantha.com/raga_raag/film_song_raga.html lists a large number of Bollywood songs based on ragas. Most of these songs were composed between 1940s and 1970s.

also show a distinct Middle Eastern influence. However, *Kirwani* is also used in Russian folk songs and early usage of this scale in Bollywood, promoted by S-J, can be traced back to its Russian inspiration. We also notice that the recent upsurge in the use of *Bhairavi*, especially in compositions by Pritam, is not related to Middle Eastern folk music, but is inspired by Punjabi folk songs. Interestingly, we observe several other *Bhairavi*-based songs composed by Pritam which do not have any apparent resemblance to either Punjabi or Middle Eastern folk songs. Nevertheless, the melodic structure of the songs clearly reveals an underlying Punjabi inspiration.

Inspiration Patterns for Composers. We observe that some composers, such as R. D. Burman, use a scale extensively within a short period of time for a large number of apparently (melodically) unrelated compositions. Such composers seem to be consciously or subconsciously guided by the same melodic inspiration, which results in compositions based on the same scale, even though the situational context of the song in the film may not demand a specific genre or regional character in the music. For instance, out of 16 songs in our dataset composed by R. D. Burman between 1971 and 73, 13 are in *Asavari* (7), *Bilawal* (4) and *Nayaki* (1) and all are clearly influenced by Blues, Rock and Western Pop. The rest are in *Khamaj* (2) and *Bhairavi* (1), and are based on South Indian and Punjabi folk songs. On the other hand, between 1980 and 81, out of the 16 compositions by him, 7 are in *Kirwani* and *Bhairavi* all reflecting distinctive Middle Eastern influence. On the other hand, for A. R. Rahman we do not see any such temporal clustering of scales between 1992 and 2005, but between 2006 and 2009, we observe a preference towards *Asavari* (5 out of 9). Surprisingly, these compositions have been inspired by Middle Eastern music and Sufi music which also use this scale, and not so much by Western pop, which is what *Asavari* is most commonly used for in Bollywood.

Diversity of Scales. Finally, we note that the number of scales used during each decade in Bollywood has been more or less constant throughout its history. To confirm this, we computed the entropy of the distribution of the scale usage for each decade, which turns out to be more or less constant. Thus, the commonly held opinion that "all Bollywood songs sound pretty much the same these days", i.e., the melodic diversity was greater previously than it is today, is not supported by our data. This diversity is an effect of multiple influences drawn from various cultures. Of course, the sources of inspiration change over time, and so do the scale usage and melodic patterns; but the scale diversity remains high at any given point of time.

6. CONCLUSION

In this work, we created a small dataset of 310 Bollywood songs sampled from the yearly top lists and identified the melodic scales of the songs. Through an analysis of this dataset, we identified and quantified various cultural influences on Bollywood music and its composers over the years. Even though 310 songs are not sufficient for making any strong claims or observing complex cultural influences, we believe that our analysis revealed some striking facts about the scale usage patterns and has helped us to formulate interesting conjectures which can be statistically verified if one had more data.

There has been a similar study for rhythmic patterns and other features of compositions in Western popular music [12], but we do not know of any study that analyzes melodic scales for a given genre. Our next step is to study scale usage in various other musical cultures around the world as well as regional film industries and parallel musical traditions extant in India and extend the current dataset to include more Bollywood songs.

7. ACKNOWLEDGMENTS

We would like to thank Ms. Nandini Kamath for helping us with identification of the scales.

8. REFERENCES

[1] G. Anantharaman. *Bollywood Melodies-A history of the Hindi film song*. Penguin Books India, 2008.

[2] Aison E. Arnold. Hindi film git: On the history of indian popular music. PhD. Thesis completed at the University of Illinois at Urbana-Champagne, 1991.

[3] A. Behl and M. Choudhury. A corpus linguistic study of Bollywood song lyrics. In *Proceedings of ICON*. 2011.

[4] V. N Bhatkhande. *Hindustani Sangit Paddhati*. Sakhi Prakashan, 1990.

[5] P. Chatterjee. When melody ruled the day. In Vasudevan, editor, *Frames of mind: Reflections on Indian cinema*. UBPSD.

[6] T. Christensen, editor. *The Cambridge History of Western Music Theory*. Cambridge University Press, 2001.

[7] S. Gopal and S. (ed) Moorti. *Global Bollywood: The Transnational Travels of Hindi Song and Dance*. University of Minnesota, 2008.

[8] N. A. Jairazbhoy. *The Rags of North Indian Music: Their Structure and Evolution*. Popular Prakashan, 1995.

[9] P. Manuel. *Cassette Culture: popular music and technology in north India*. University of Chicago Press, 1993.

[10] S. Marcus. Recycling Indian Film-Songs: Popular Music as a Source of Melodies for North Indian Folk Musicians. *Asian Music*, 24(1), 1993.

[11] A. Morcom. Hindi Film Songs and the Cinema. *SOAS Musicology Series*, 2007.

[12] J. Serrá, A. Corral, M. Boguñá, M. Haro, and J. L. Arcos. Measuring the evolution of contemporary Western popular music. *Nature Scientific Reports*, 2(521), 2012.

BILEVEL SPARSE MODELS FOR POLYPHONIC MUSIC TRANSCRIPTION

Tal Ben Yakar,[1] Roee Litman,[1] Pablo Sprechmann,[2] Alex Bronstein,[1] and Guillermo Sapiro[2]
[1]Tel Aviv University, [2]Duke University
talby10@gmail.com, roeelitm@post.tau.ac.il, bron@eng.tau.ac.il
{pablo.sprechmann,guillermo.sapiro}@duke.edu

ABSTRACT

In this work, we propose a trainable sparse model for automatic polyphonic music transcription, which incorporates several successful approaches into a unified optimization framework. Our model combines unsupervised synthesis models similar to latent component analysis and nonnegative factorization with metric learning techniques that allow supervised discriminative learning. We develop efficient stochastic gradient training schemes allowing unsupervised, semi-, and fully supervised training of the model as well its adaptation to test data. We show efficient fixed complexity and latency approximation that can replace iterative minimization algorithms in time-critical applications. Experimental evaluation on synthetic and real data shows promising initial results.

1. INTRODUCTION

The goal of automatic music transcription (AMT) is to obtain a musical score from an input audio signal. AMT is particularly difficult when the audio signal is polyphonic [12], as the harmonic relations and interactions in music signals challenges the detection of multiple concurrent pitches. Polyphonic AMT is still considered an open problem and the state-of-the-art solutions are far from the level of precision required in many applications. We refer the reader to [4] for a detailed description of the open questions and challenges in polyphonic AMT.

Work partially supported by BSF, ONR, NGA, NSF, ARO, and AFOSR.

1.1 Prior work

In what follows, we briefly review two main families of approaches recently used for AMT, which are particularly relevant for the present work. Reviewing all existing AMT methods is beyond the scope of this paper; for recent surveys, we refer to the reader to [4, 12] and references therein.

Being essentially a classification task, music transcription has been addressed by classification-based approaches. These techniques define a set of meaningful features for pitch and onset detection, and feed them to generic classification schemes such as neural networks [6, 15], deep believe networks [16], and support vector machines [17]. In [16], the features themselves were learned form the data. In [17], the authors argue that prior knowledge (such as harmonicity) is not strictly necessary for achieving levels of transcription accuracy comparable to the one obtained with competing approaches, and that such assumptions can be substituted with discriminative learning. While being feasible, the lack of insight makes such pure learning-based systems hard to train, since they need to infer from the training data all possible variations and combinations of pitches. In general, this translates into long off-line training times and requires huge training sets.

Another family of recent approaches is based on spectrogram factorization techniques, such as non-negative matrix factorization (NMF) [13], and its probabilistic counterpart – probabilistic latent component analysis (PLCA) [20]. The basic idea, first introduced in [19], aims at factorizing a spectral representation of the signal $\mathbf{X} \in \mathbb{R}^{n\times k}$, into a product of non-negative factors, $\mathbf{X} \approx \mathbf{DZ}$, where the $n \times p$ non-negative dictionary $\mathbf{D}$ contains templates of the individual pitches, and the $p \times k$ non-negative factor $\mathbf{Z}$ contains the corresponding activations for each frame. Ideally one would expect $\mathbf{Z}$ to resemble a piano-roll and reveal the active notes in each spectral frame. Unfortunately, this is not enough in practice. In order to overcome this problem, many approaches have proposed to regularize the factorization by including

sparsity [1], harmonicity, and smoothness [5, 22]. In [2], the authors propose a shift-invariant version of PLCA, where the dictionary contains note templates from multiple orchestral instruments. Including such a regularizations usually significantly improves the results but also translates into slower coding schemes. This contrasts with the discriminative approaches that, after training, have very light computational costs. On one hand, the generative nature of factorization approaches allows them to handle the spectral superposition of harmonically related pitches in a natural way. On the other hand, however, such generative approaches seem less flexible and more difficult to adapt to specific settings compared to their discriminative counterparts.

1.2 Contributions

In this paper, we present an attempt to inject the generative properties of factorization approaches into a discriminative setting. We aim at establishing a bridge between pure learning and factorization-based methods. Specifically, we propose the coupled training of a set of classifiers that detect the presence of a given pitch in a frame of audio, taking as the input the activations produced by a generative matrix factorization scheme. Instead of constraining the factorization algorithm, we design a very simple factorization method trained to produce the optimal input to a binary classifier in the sense of the classification performance. Once trained, the simplicity of the proposed factorization scheme allows to use fast approximation of sparse encoders, resulting in computation complexity comparable to that of pure discriminative models. With this implementation, the proposed method bridges between factorization and classification methods, designing a neural network that solves a meaningful factorization problem.

We formulate our model as a bilevel optimization program, generalizing supervised dictionary learning devised for sparse coding schemes [14]. We also incorporate elements of metric learning into this supervised sparse NMF setting in order to increase its discriminative power. The proposed approach is naturally amenable to semi-supervised training regimes, in which unlabeled data can be used to adapt the system. Finally, the output of these classifiers is temporally smoothed as is normally done in AMT [2, 17].

In Section 2 we present the proposed model coupling the codes with the pitch classifiers. Then, in Section 3, we formulate its supervised variant as a bilevel optimization problem. In Section 4, we describe how the proposed method can be significantly accelerated. Section 5 shows how to incorporate the proposed scheme into higher-level temporal models. Experimental evaluation is reported in Section 6. Finally, Section 7 concludes the paper.

2. NON-NEGATIVE SPARSE MODEL

Like the majority of music and speech analysis techniques, music transcription typically operates on the magnitude of the audio time-frequency representation such as the short-time Fourier transform or constant-Q transform (CQT) [8], as adopted in this work. Given a spectral frame $\mathbf{x} \in \mathbb{R}^n_+$ at some time, the transcription problem consists of producing a binary label vector $\mathbf{y} \in \{-1, +1\}^p$, whose i-th element indicates the presence $(+1)$ or absence (-1) of the i-th pitch at that time. We use $p = 88$ corresponding to the span of the standard piano keyboard (MIDI pitches $21 - 108$).

In the proposed model, the output vector is produced by applying a simple linear classifier $\mathbf{y} = \text{sign}(\mathbf{W}\mathbf{z} + \mathbf{a})$, parametrized by the $p \times m$ matrix $\mathbf{W}$ and $p \times 1$ vector $\mathbf{a}$, to the m-dimensional feature vector $\mathbf{z}$ obtained by solving the following non-negative sparse representation pursuit problem

$$\mathbf{z}(\mathbf{x}) = \arg\min_{\mathbf{z} \geq \mathbf{0}} \frac{1}{2}\|\mathbf{M}(\mathbf{x} - \mathbf{D}\mathbf{z})\|_2^2 + \lambda_1\|\mathbf{z}\|_1 + \lambda_2\|\mathbf{z}\|_2^2. \quad (1)$$

Here, $\mathbf{D}$ is an $n \times m$ over-complete ($m > n$) non-negative dictionary, whose columns represent different templates for each of the individual pitches, and $\mathbf{M}$ is a $r \times n$ metric matrix ($r \leq n$).

The first data fitting term requires the data to be well-approximated by a sparse non-negative combination of the atoms of $\mathbf{D}$, expressing the assumption that at each time, only a few pitches are simultaneously present. Replacing the standard Euclidean fitting term by a more general Mahalanobis metric parametrized by the matrix $\mathbf{M}$ allows to give different weights to different frequencies, as frequently practiced in music processing. The second term, whose relative importance is governed by the parameter λ_1, actually promotes the sparsity of the solution vector, while the third term is added for regularization.

Pursuit problem (1) is a strictly convex optimization problem, which can be solved efficiently using (among other alternatives) a family of optimization techniques called proximal methods. We adopt a non-negative variant of the iterative shrinkage-thresholding algorithm (ISTA) [9], summarized in Algorithm 1. While faster versions of this fixed-step proximal method can be used to reach linear convergence rates, the discussion of these extensions is beyond of the scope of this paper.

We observe that given a collection of spectral frames $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_k)$, the solution of the pursuit problem aims at finding a non-negative factoriza-

input : Data $\mathbf{x}$, dictionary $\mathbf{D}$, metric matrix $\mathbf{M}$, parameters λ_1, λ_2, step size α.
output: Non-negative sparse code $\mathbf{z}$.
Define $\mathbf{H} = (1 - \frac{\lambda_2}{\alpha})\mathbf{I} - \frac{1}{\alpha}\mathbf{M}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}\mathbf{D}\mathbf{M}$,
$\mathbf{G} = \frac{1}{\alpha}\mathbf{M}^{\mathrm{T}}\mathbf{D}^{\mathrm{T}}$, $\mathbf{t} = \frac{1}{\alpha}\lambda_1$.
Initialize $\mathbf{z}^1 = \mathbf{0}$ and $\mathbf{b}^1 = \mathbf{G}\mathbf{x}$.
for $k = 1, 2, \ldots$ *until convergence* **do**
  $\mathbf{z}^{k+1} = \sigma_{\mathbf{t}}(\mathbf{b}^k)$
  $\mathbf{b}^{k+1} = \mathbf{b}^k + \mathbf{H}(\mathbf{z}^{k+1} - \mathbf{z}^k)$
end

Algorithm 1: Non-negative iterative shrinkage-thresholding algorithm (ISTA). $\sigma_{\mathbf{t}}(\mathbf{b}) = \max\{\mathbf{0}, \mathbf{b} - \mathbf{t}\}$ denotes element-wise single-sided soft thresholding.

tion of $\mathbf{X}$ into $\mathbf{DZ}$, thus being essentially an instance of a non-negative matrix factorization (NMF) problem with a fixed left factor $\mathbf{D}$. The approach proposed in the paper can be essentially viewed as a supervised version of NMF.

Denoting the parameters of the sparse model as $\mathbf{\Theta} = \{\mathbf{D}, \mathbf{M}\}$, and those of the linear classifier as $\mathbf{\Phi} = \{\mathbf{W}, \mathbf{a}\}$, the proposed pitch transcription system can be expressed as $\mathbf{y} = \mathbf{y}_{\mathbf{\Phi}}(\mathbf{z}_{\mathbf{\Theta}}(\mathbf{x}))$, where $\mathbf{z}_{\mathbf{\Theta}}$ denotes the non-linear map produced by solving (1), and $\mathbf{y}_{\mathbf{\Phi}}$ refers to the application of the classifier. In what follows, we will address how to train and adapt the parameters $\mathbf{\Theta}$ and $\mathbf{\Phi}$ for the AMT task.

Dictionary initialization. The initial dictionary is constructed to contain spectral templates for each possible pitch. The training of the dictionary is done by learning a set of small sub-dictionaries, one per pitch, minimizing

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{Z}\geq\mathbf{0}} \frac{1}{2}\|\mathbf{M}(\mathbf{X} - \mathbf{DZ})\|_{\mathrm{F}}^2 + \lambda_1\|\mathbf{Z}\|_1 + \lambda_2\|\mathbf{Z}\|_{\mathrm{F}}^2, \quad (2)$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm, $\mathcal{D}$ is the space of appropriately sized non-negative matrices with unit columns, and $\mathbf{M} = \mathbf{I}$. Additional constrains such as harmonicity can be included by changing $\mathcal{D}$ to be more restrictive. The initial dictionary can be constructed for a specific instrument or for multiple instruments as in [2], via simple concatenation.

Classifier initialization. Once the initial dictionary has been trained, we can learn the classifier parameters $\mathbf{\Phi}$. To that end, we construct a training set $\mathcal{X}$ containing pairs of the form $(\mathbf{x}, \mathbf{y})$, of spectral frames with the corresponding groundtruth pitch labels. Here, unlike in the unsupervised dictionary training, the best performance of the classifier is obtained when the training set contains representative examples of chords and pitch combinations.

The classifier is trained by minimizing

$$\min_{\mathbf{\Phi}} \frac{1}{|\mathcal{X}|}\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{X}} \ell(\mathbf{y}_{\mathbf{\Phi}}(\mathbf{z}), \mathbf{y}) \quad (3)$$

on the outputs $\mathbf{z} = \mathbf{z}_{\mathbf{\Theta}}(\mathbf{x})$ of the pursuit algorithm. Here, ℓ denotes a loss function penalizing for the mismatch between the ground truth labels and the actual output of the classifier. We use the logistic regression loss function $\ell(\mathbf{y}', \mathbf{y}) = \log(1 + e^{-\mathbf{y}^{\mathrm{T}}\mathbf{y}'})$. The minimization of (3) can scale to very large training sets by using (projected) stochastic gradient descent (SGD) techniques [7], which we adopt in all our experiments.

3. BILEVEL SPARSE MODEL

A striking disadvantage of the two-stage training described so far is the fact that the training of the dictionary $\mathbf{D}$ aims at reducing the data reconstruction error $\|\mathbf{X} - \mathbf{DZ}\|_{\mathrm{F}}$ rather than reducing the classification error (3). Consequently, the dictionary trained in the initial unsupervised regime is suboptimal in the sense of (3); furthermore, there is no natural way to train the metric matrix $\mathbf{M}$. The ultimate way to perform supervised training of the entire system would be therefore by minimizing

$$\min_{\mathbf{\Theta},\mathbf{\Phi}} \frac{1}{|\mathcal{X}|}\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{X}} \ell(\mathbf{y}_{\mathbf{\Phi}}(\mathbf{z}_{\mathbf{\Theta}}(\mathbf{x})), \mathbf{y}) + \mu||\mathbf{\Phi}||_{\mathrm{F}}^2 \quad (4)$$

not only with respect to the parameters $\mathbf{\Phi}$ of the classifier, but also with respect to the parameters $\mathbf{\Theta}$ of the pursuit. This leads to a *bilevel* optimization problem, as we need to optimize the loss function ℓ, which in turn depends on the minimizer of (1). Note that, as is standard practice in machine learning, a regularization term on the classifier parameters is added to prevent over-fitting.

In particular, one would need to compute the gradients of the loss with respect to the parameters $\mathbf{\Theta} = \{\mathbf{D}, \mathbf{M}\}$ of the pursuit. Fortunately, $\mathbf{z}$ is almost everywhere differentiable with respect to $\mathbf{D}$ and $\mathbf{M}$ [14]. Denoting by Λ the active set of $\mathbf{z}$ (i.e., the set of indices at which it attains non-zero values), we define

$$\boldsymbol{\beta}_{\Lambda} = (\mathbf{D}_{\Lambda}^{\mathrm{T}}\mathbf{M}^{\mathrm{T}}\mathbf{M}\mathbf{D}_{\Lambda} + \lambda_2\mathbf{I}_{\Lambda})^{-1}(\nabla_{\mathbf{z}}\ell(\mathbf{y}_{\mathbf{\Phi}}(\mathbf{z}), \mathbf{y}))_{\Lambda},$$

where $\nabla_{\mathbf{z}}\ell$ is the gradient of the loss function with respect to $\mathbf{z}$. The elements of $\boldsymbol{\beta}$ outside Λ are set to zero. The gradients of $\ell(\mathbf{y}_{\mathbf{\Phi}}(\mathbf{z}), \mathbf{y})$ with respect to $\mathbf{D}$ and $\mathbf{M}$ can be expressed as

$$\begin{aligned} \nabla_{\mathbf{D}}\ell &= \mathbf{M}^{\mathrm{T}}\mathbf{M}((\mathbf{x} - \mathbf{D}\mathbf{z}^*)\boldsymbol{\beta}^{\mathrm{T}} - \mathbf{D}\boldsymbol{\beta}\mathbf{z}^*) \quad (5)\\ \nabla_{\mathbf{M}}\ell &= \mathbf{M}\mathbf{D}_{\Lambda}\boldsymbol{\beta}_{\Lambda}(\mathbf{x} - \mathbf{D}\mathbf{z}^*)^{\mathrm{T}} - \mathbf{M}(\mathbf{x} - \mathbf{D}\mathbf{z}^*)\mathbf{D}_{\Lambda}^{\mathrm{T}}. \end{aligned}$$

We omit the derivation details due to lack of space, and refer the reader to [14] for a related discussion.

We perform the minimization of (4) again by using SGD alternating descents on $\mathbf{\Phi}$ keeping $\mathbf{\Theta}$ fixed, and on $\mathbf{\Theta}$ keeping $\mathbf{\Phi}$ fixed. $\mathbf{\Theta}$ and $\mathbf{\Phi}$ are initialized as described in Section 2. It is also worthwhile noting that the minimization of the discriminative loss (4) with respect to the matrix $\mathbf{M}$ can be viewed as a particular setting of *metric learning* – a family of problems that aim at designing task-specific metrics. In our case, we design a Mahalanobis metric $\mathbf{M}^{\mathrm{T}}\mathbf{M}$ such that the pursuit with respect to it minimizes the classification errors.

The purely discriminative objective of (4) is susceptible to over-fitting since the learned matrix $\mathbf{M}$ will not aim at producing faithful data reconstructions. In that case, the generative advantage of NMF would be lost. To avoid this problem, the minimization of (4) can be regularized by adding an data reconstruction term of the form $\|\mathbf{x} - \mathbf{D}\mathbf{z}_{\mathbf{\Theta}}(\mathbf{x})\|_2^2$.

We distinguish between two training regimes: in the *fully supervised* setting, all samples in the training set come with label information $\mathbf{y}$, and the training is performed as described above. Since label information is often difficult to obtain, in many practical cases only some of the samples in the training set are labeled. We call such a setting *semi-supervised*. Given a set of unlabeled data, $\mathcal{X}_u$, we can change the learning process by augmenting the discriminative loss (4) on the labeled data (eventually regularized with data reconstruction term), with the unsupervised term

$$\sum_{\mathbf{x}\in\mathcal{X}_u} \frac{1}{2}\|\mathbf{x}-\mathbf{D}\mathbf{z}_{\mathbf{\Theta}}(\mathbf{x})\|_2^2+\lambda_1\|\mathbf{z}_{\mathbf{\Theta}}(\mathbf{x})\|_1+\lambda_2\|\mathbf{z}_{\mathbf{\Theta}}(\mathbf{x})\|_2^2.$$

This new training scheme aims at producing a dictionary $\mathbf{D}$ that is good for classifying (and reconstructing) the labeled data but that can also used to sparsely represent the unlabeled data. Note that this regime can also be used as a way to *adapting* the system to unseen testing data. Both factorization- and classification-based approaches suffer a performance drop when the testing data are not well represented by the training samples. In this way, our system can be adapted to new unseen (and unlabeled) data.

4. FAST APPROXIMATION

The proposed approach relies on solving optimization problem (1) using an iterative method. One of the drawbacks of such iterative schemes is their relatively high computational complexity and latency, which is furthermore data-dependent. For example, non-negative ISTA typically requires hundreds of iterations to converge. However, while the classical optimization theory provides worst-case (data-independent) convergence rate bounds for many families of iterative algorithms, very little is known about their behavior on *specific* data, coming e.g., from a distribution supported on a low-dimensional manifold – properties often exhibited by real data. Common practice of sparse modeling concentrates on creating sophisticated data models, and then relies on computational and analytic techniques that are totally agnostic of the data structure.

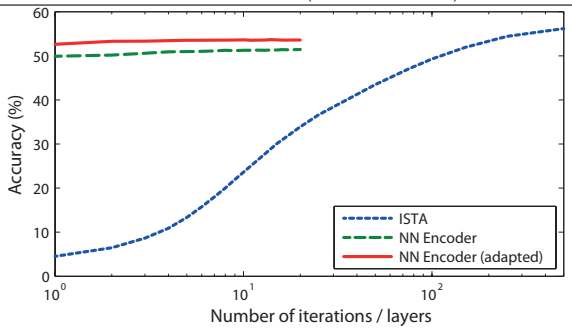


Figure 1. Accuracy of the optimization-based and neural network encoders as a function of the number of iterations or layers. Evaluation was performed on dataset from [17]. The networks were trained in the unsupervised regime.

From the perspective of the pursuit process, the minimization of (1) is merely a proxy to obtaining a highly non-linear map between the data vector $\mathbf{x}$ and the corresponding feature vector $\mathbf{z}$. Adopting ISTA as the iterative algorithm, such a map can be expressed by unrolling the iterations into the composition $f \circ f \circ \cdots \circ f(\mathbf{0}, \mathbf{G}\mathbf{x})$ of T elementary operations of the form $f : (\mathbf{z}, \mathbf{b}) \mapsto (\sigma_{\mathbf{t}}(\mathbf{b}), \mathbf{b} + \mathbf{H}(\sigma_{\mathbf{t}}(\mathbf{b}) - \mathbf{b}))$, where T is a sufficiently large number of iterations required for convergence. By fixing T, we obtain a fixed-complexity and latency encoder $\hat{\mathbf{z}}_{T,\mathbf{\Psi}}(\mathbf{x})$, parametrized by $\mathbf{\Psi} = \{\mathbf{H}, \mathbf{G}, \mathbf{t}\}$ (recall that ISTA defines the latter parameters as functions of $\mathbf{\Theta} = \{\mathbf{D}, \mathbf{M}\}$). Such an encoder can be thought of as a time-recurrent neural network, or a feed-forward network with T identical layers.

Note that for a sufficiently large T, $\hat{\mathbf{z}}_{T,\mathbf{\Psi}} \approx \mathbf{z}_{\mathbf{\Theta}}$. However, when complexity budget constraints require T to be truncated at a small fixed number, the output of $\hat{\mathbf{z}}_{T,\mathbf{\Psi}}$ is usually unsatisfactory, and the worst-case bounds provided by the classical optimization theory are of little use. However, within the family of functions $\{\hat{\mathbf{z}}_{T,\mathbf{\Psi}}\}$, there might exist better parameters for which $\hat{\mathbf{z}}$ performs better *on relevant input data*. These ideas advocated by [11], have been recently shown very effective in sound separation problems [21].

Adapted to our problem, the encoder $\hat{\mathbf{z}}_{T,\mathbf{\Psi}}$ can be trained in lieu of the iterative pursuit process in one of the discussed regimes, using the standard backpropagation techniques to compute the gradients of the network with respect to its parameters [11]. The training of the encoder can be achieved by minimize the dis-

criminative objective

$$\min_{\mathbf{\Psi},\mathbf{\Phi}} \frac{1}{|\mathcal{X}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{X}} \ell(\mathbf{y}_{\mathbf{\Phi}}(\hat{\mathbf{z}}_{T,\mathbf{\Psi}}(\mathbf{x})),\mathbf{y}) + \mu||\mathbf{\Phi}||_{\mathrm{F}}^2 \quad (6)$$

Similar ideas can be used in the unsupervised setting. Figure 1 shows, as a function of T, the performance of the exact pursuit (ISTA truncated after T iterations), and its approximation using the neural networks. About two orders of magnitude of speedup is observed.

This perspective bridges between two popular approaches to music transcription: those based on explicit data modeling and relying on optimization to solve some kind of a representation pursuit problem, and those relying on pure learning of a neural network. However, while training $\hat{\mathbf{z}}_{T,\mathbf{\Psi}}$ is technically a pure learning approach, it is very much rooted into the underlying data model. First and most importantly, the training objective is solving a matrix factorization problem. Second, the architecture of the neural network is derived from an iterative process that is guaranteed to minimize a meaningful objective. Third, the network comes with a very good initialization of the parameters (as prescribed by ISTA). Since neural network training is a highly non-convex optimization problem, such an initialization is crucial.

5. TEMPORAL REGULARIZATION

Independent analysis of spectral frames fails short of exploiting the temporal structures and dependencies of music signals. This prior knowledge can be incorporated by temporally regularizing the output of the classifiers. A popular way to achieve this is by adding a post-processing stage based on hidden Markov models (HMMs) [18]. Following [2, 17], in this work we smooth the classifier outputs using an independent two-state HMM for each pitch. We now think of the output of the sparse coding $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_k)$ as a sequence of k input observations. For each pitch p, the states are represented as a sequence of hidden variables $\mathbf{q}^p = (q_1^p, \ldots, q_k^p)$ that take the value of $+1$ in the presence, or -1 in the absence of the pitch, following the convention used throughout the paper. The HMM aims at finding for each pitch, the optimal sequence of states minimizing

$$\min_{\mathbf{q}^p} \; p(\mathbf{z}_1|\mathbf{q}_1^p)p(\mathbf{q}_1^p)\prod_{i=2}^{k} p(\mathbf{z}_i|\mathbf{q}_i^p)p(\mathbf{q}_i^p|\mathbf{q}_{i-1}^p), \quad (7)$$

where the initial probabilities $p(\mathbf{q}_1^p)$ and the transition probabilities $p(\mathbf{q}_i^p|\mathbf{q}_{i-1}^p)$ are learned from the data. The probability of observing a sparse code given a pitch state, $p(\mathbf{z}_i|\mathbf{q}_i^p)$, can be obtained naturally from the output of the classifiers. The logistic classifiers can be thought as generalized linear predictors for Bernoulli variables, leading to $p(\mathbf{z}_i|\mathbf{q}_i^p = y) = 1/(1+e^{-y\mathbf{W}^{\mathrm{T}}\mathbf{z}_i})$. Hence, maximizing the classifier's performance can be thought as maximizing the likelihood of the true state in the HMM. Problem (7) is solved using the Viterbi algorithm [18].

Table 1. Precision, recall, F1 and accuracy in percent on the test data presented in [17] for the proposed approach under different training regimes. For reference, the accuracy obtained for three recent alternative methods is: 57.6 % for [2], 56.5 % for [17], and 47.0 % for [3].

Training regime	Pre.	Rec.	F1	Acc.
Supervised	81.6	69.8	74.3	60.0
Supervised+Fitting	79.7	69.9	73.7	59.2
Semi-supervised	79.7	70.0	73.7	59.2
Semi-supervised+Fitting	82.0	70.9	75.1	61.0

6. EXPERIMENTAL EVALUATION

Similarly to the factorization-based methods, the proposed model can be trained to transcribe pieces containing mixtures of instruments by appropriately training the initial dictionaries. However, since the scope of the paper is rather a proof-of-concept than the design of a full-featured AMT system, we limit the experimental evaluation to piano recordings only.

Data. The system was tested on the Disklavier dataset proposed in [17]. For training of the initial dictionaries (pitch templates), two different piano types were used from the MAPS dataset [10]. Then, the classifier and the dictionary were trained in the supervised regime using the first 30 seconds of 50 songs from MAPS and the training data in [17]. Spectral frames were represented using CQT with 48 frequency bins per octave.

Performance measures. We adopted the frame-based accuracy measure proposed in [17], $\mathrm{Acc} = \mathrm{TP}/(\mathrm{FP}+\mathrm{FN}+\mathrm{TP})$, where TP (true positives) is the number of correctly predicted pitches, and FP (false positives) and FN (false negatives) are the number of pitches incorrectly transcribed as ON or OFF, respectively. We also include the standard frame-based precision, recall and F1 measures.

Evaluation. The proposed system was evaluated under different training regimes. Training the system by solving the bilevel optimization problem (4) is referred to as *Supervised*, and *Supervised+Fitting* when using the additional data fitting term described in Section 3. We also tested the capability of our system to use unlabeled data to unsupervisedly adapt to the test data (the *Semi-supervised+Fitting* settings). The obtained performance was compared against one successful representative approach from each of the

main existing philosophies. We used [2] to represent the factorization-based approaches, and [17] for the classification-based ones. Both of these methods include a post-processing stage very similar to the one used in the work. We also included a third method based on onset detection [3]. It is worth mentioning that the training of [2] does not use the Disklavier training data given in [17]; the authors train their system using samples of three different piano types of the MAPS dataset.

Table 1 summarizes the obtained results. The proposed method is competitive with the alternative approaches. The inclusion of unlabeled data allows a significant improvement in the system performance. Adding the fitting term seems to have more impact when unlabeled data are available. We attribute this to the reduction of over-fitting risks that such a regularization offers. In both semi- and fully-supervised regimes, the reconstruction properties of the dictionary are much better preserved with the mentioned fitting regularization.

7. CONCLUSION

We showed a trainable bilevel non-negative sparse model model for polyphonic music transcription. Our model can be interpreted as a supervised variant of NMF as well as a flavor of metric learning. We also showed that the original iterative optimization-based approach can be efficiently approximated by fixed-complexity feed-forward architectures that give a two-order-of-magnitude speedup at little expense of accuracy. This creates an interesting relation between the optimization-based transcription methods, and those relying on pure learning, which are traditionally dealt with by two, practically disjoint, communities. The approach can naturally benefit from the inclusion of unlabeled data via a semi-supervised training scheme.

8. REFERENCES

[1] S. Abdallah and M. Plumbley. Polyphonic music transcription by non-negative sparse coding of power spectra. In *ISMIR*, pages 10–14, 2004.

[2] E. Benetos and S. Dixon. Multiple-instrument polyphonic music transcription using a convolutive probabilistic model. In *Sound and Music Computing Conference*, pages 19–24, 2011.

[3] E. Benetos and S. Dixon. Polyphonic music transcription using note onset and offset detection. In *ICASSP*, pages 37–40. IEEE, 2011.

[4] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: Breaking the glass ceiling. In *ISMIR*, 2012.

[5] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in Bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Trans. Audio, Speech, and Language Proc.*, 18(3):538–549, 2010.

[6] S. Bock and M. Schedl. Polyphonic piano note transcription with recurrent neural networks. In *ICASSP*, pages 121–124, 2012.

[7] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*, pages 177–187, August 2010.

[8] J. C. Brown. Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*, 89:425, 1991.

[9] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004.

[10] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Trans. Audio, Speech, and Language Proc.*, 18(6):1643–1654, 2010.

[11] K. Gregor and Y. Lecun. Learning fast approximations of sparse coding. In *ICML*, 2010.

[12] A. Klapuri. Multipitch analysis of polyphonic music and speech signals using an auditory model. *IEEE Trans. Audio, Speech, and Language Proc.*, 16(2):255–266, 2008.

[13] D. D. Lee and H. S. Seung. Learning parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[14] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *IEEE Trans. PAMI*, 34(4):791–804, 2012.

[15] M. Marolt. A connectionist approach to automatic transcription of polyphonic piano music. *IEEE Trans. Multimedia*, 6(3):439–449, 2004.

[16] J. Nam, J. Ngiam, H. Lee, and M. Slaney. A classification-based polyphonic piano transcription approach using learned feature representations. In *ISMIR*, 2011.

[17] G. E. Poliner and D. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP J. Adv. in Sig. Proc.*, 2007, 2006.

[18] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE*, 77(2):257, 1989.

[19] P. Smaragdis and J. Brown. Non-negative matrix factorization for polyphonic music transcription. In *WASPAA*, pages 177–180, 2003.

[20] P. Smaragdis, B. Raj, and M. Shashanka. A probabilistic latent variable model for acoustic modeling. In *NIPS*, volume 148, 2006.

[21] P. Sprechmann, A. Bronstein, and G. Sapiro. Real-time online singing voice separation from monaural recordings using robust low-rank modeling. In *ISMIR*, 2012.

[22] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Trans. Audio, Speech, and Language Proc.*, 18(3):528–537, 2010.

JPRODUCTIONCRITIC: AN EDUCATIONAL TOOL FOR DETECTING TECHNICAL ERRORS IN AUDIO MIXES

Cory McKay
Marianopolis College and CIRMMT
cory.mckay@mail.mcgill.ca

ABSTRACT

jProductionCritic is an open-source educational framework for automatically detecting technical recording, editing and mixing problems in audio files. It is intended to be used as a learning and proofreading tool by students and amateur producers, and can also assist teachers as a timesaving tool when grading recordings.

A number of novel error detection algorithms are implemented by jProductionCritic. Problems detected include edit errors, clipping, noise infiltration, poor use of dynamics, poor track balancing, and many others.

The error detection algorithms are highly configurable, in order to meet the varying aesthetics of different musical genres (e.g. Baroque vs. noise music). Effective general-purpose default settings were developed based on experiments with a variety of student pieces, and these settings were then validated using a reserved set of student pieces.

jProductionCritic is also designed to serve as an extensible framework to which new detection modules can be easily plugged in. It is hoped that this will help to galvanize MIR research relating to audio production, an area that is currently underrepresented in the MIR literature, and that this work will also help to address the current general lack of educational production software.

1. INTRODUCTION

Audio production is a broad field that essentially involves recording and creating music. Important aspects include:

- *Recording:* configuring an acoustic environment, microphone selection, microphone placement, etc.
- *Editing:* shifting segments of audio in time within a track, or moving them between tracks.
- *Mixing:* combining multiple tracks with appropriate gains, panning and EQ settings, applying effects, etc.
- *Synthesis:* artificially generating audio.
- *Sampling:* incorporating pre-existing audio.
- *Mastering:* preparing a mix for final distribution via specific audio formats.

DAW (Digital Audio Workstation) software has come to play a central role in production. Such software ranges from recording-oriented tools like Avid Pro Tools, to live performance-oriented software such as Ableton Live, to free tools like Audacity.

Improved functionality, better user interfaces and decreasing costs have made audio production more and more accessible in recent years. This has helped to cause an explosion of content created in home studios, ranging from amateur mashups to recordings by professional musicians. While this has certainly resulted in a great deal of interesting music, it has also led to error-prone technical work on the part of overconfident amateur producers who lack the professional training that was previously necessary to be involved in production at all.

This problem is part of the motivation behind the jProductionCritic software, which automatically detects technical production errors, especially those relating to editing and mixing. It can help students and amateur producers check their work for unnoticed errors, much as one might use a grammar checker when writing prose. This is beneficial from an educational perspective, as it teaches users to notice problems that they might not otherwise have known to look for. This in effect trains them to improve their listening skills, which are arguably a producer's greatest asset, and pushes them to learn how to avoid or correct the detected problems, thus improving not only their current work, but also the skills that they will be able to apply in the future.

Such error checking software is also useful to those teaching audio production, as it can greatly facilitate grading. While it would certainly be ill-advised to rely exclusively on automated marking, as expert humans are needed to fully evaluate the difficult-to-quantify aesthetics involved in the art of production, simply automating the painstaking task of enumerating and time-stamping basic technical errors can be a great time saver.

Finally, error checking software could even be of some use to professional audio engineers as a final verification tool, just as professional writers make use of spellcheckers. It is not unheard of to find technical errors in professional work, often due to rushed production schedules and the high costs of studio time.

Aside from such practical benefits, developing algorithms for detecting production errors can also have important research value. As discussed in Section 2, not

only is there currently a surprising dearth of production-oriented research in the MIR community specifically, but those tools that do exist in the general audio world tend to be closed-source black boxes or based on disappointingly naïve algorithms. This presents an exciting research opportunity, particularly considering the importance of production from both commercial and artistic perspectives. jProductionCritic has therefore been designed not only as a ready-to-use application, but also as a modular framework for developing and deploying new error detection and analysis algorithms in the future.

2. RELATED RESEARCH

Commercial DAW software like Pro Tools tends to offer some basic error detection functionality, and extensive additional functionality can be added via plug-ins. Unfortunately, with certain important exceptions, such functionality tends to be relatively simplistic in implementation and based on proprietary closed-source code, making it expensive to use and difficult to extend. Also, such software tends to emphasize correcting problems rather than detecting where they occur (the latter is not always necessary for the former), something that is of limited educational value. Finally, DAW applications and plug-ins tend to address specific problems independently, with limited functionality for presenting errors to users via integrated interfaces.

Moving outside the domain of DAW software, there are a few commercial integrated error detection systems, such as Fraunhofer IDMT's A/V Analyzing Toolbox [4] and Quadriga Audiofile-Inspector [12]. Unfortunately, as with commercial DAWs, such software is closed-source and thus difficult for independent researchers to extend. This software is also limited in the range of errors detected and in the sophistication of its processing.

In terms of open-source integrated systems, the very basic Digital Audio Error Detection [13] is the only one available. A few open-source error detection DAW plug-ins can also be found, but they are isolated algorithms that each only look for individual errors, and have no integration with one another. Furthermore, they are largely intended for professional use, and typically require a significant amount of knowledge to use, limiting their usefulness in educational contexts.

With respect to research in the MIR community, surprisingly little work has been done relating directly to audio production, even though many of the audio features and metrics used in MIR research are highly relevant to this domain. Scott and Kim [9] and Montecchio and Cont [6] provide good examples of the kind of production-oriented MIR research has been done, but even this high-quality work focuses on automating production tasks rather than finding errors. Such automation is certainly very useful in practice, but it does not address the educational needs emphasized by jPoductionCritic.

There has been a substantial amount of research done outside the MIR community on detecting errors in audio signals. However, this focuses mainly on techniques associated with specific problems rather than general integrated systems. Furthermore, much of this research relates to domains such as broadcasting and audio compression, with less focus on production-oriented problems, and with almost no attention paid to addressing the issue from an educational perspective. Having noted this, there are many technical papers than can each be very useful in detecting specific production problems, particularly in AES (Audio Engineering Society) and IEEE (Institute of Electrical and Electronics Engineers) publications. There are also a number of important general references on audio production, including books by Barlett and Barlett [1], Vaseghi [10], Owsinki [7] and Huber and Runstein [3], the first of which includes a particularly useful chapter on the kinds of defects that one can encounter in an improperly prepared mix.

3. DESIGN AND FUNCTIONALITY

The first main design objective of jProductionCritic is that it be useful and accessible to music students, amateur producers and teachers, all of whom may have little or no experience with software development. To this end, jProductionCritic is distributed with a detailed manual in order to make it as easy to learn as possible. Its basic interface is also designed to be minimalistic and direct so that users can avoid being distracted by anything superfluous. Users simply need to specify an audio file or batch of files to check and where reports are to be saved, and the software automatically takes care of the rest.

Of course, it is also important that the software be highly configurable for those who desire flexibility. There is therefore a separate extensive configuration file that advanced users can modify in order to control which errors are checked for, what error thresholds are used for each error, and so on. It is thus possible to customize jProductionCritic for certain styles of music (e.g. metal vs. jazz), or to simply use the provided general-purpose default settings without worrying about the details.

jProductionCritic processes final mixes in the form of single mono or stereo files rather than DAW projects with tracks still separated out. Although this does make certain errors much harder to detect, it also ensures that no new unchecked errors are incorporated during final mixing and mastering. This also makes it possible to process any standard audio recording with jProductionCritic, and avoids tying it to any particular DAW framework.

Three types of error reports can be generated by jProductionCritic for each audio file. The first is simply an enumeration of the errors that were detected, annotated with time stamps indicating either an instantaneous time or time range, as appropriate. Errors are also marked as being mild, moderate or severe.

The second type of report consists of a series of Audacity Label Tracks, one for each type of time-specific error. These are metadata tracks that Audacity displays alongside audio and MIDI tracks. This can be very useful for visually demonstrating to users where errors occur in a waveform or spectrogram. Audacity was chosen in particular because it is free and thus accessible to all users, whether or not they used it to prepare the audio being checked for errors.

The third type of error report consists of error annotations in Weka ARFF [11] or ACE XML [5], two file formats related to machine learning that are used by the MIR community. Although not directly applicable to the educational context targeted by jProductionCritic, these reports could be helpful to MIR researchers who might want to use the output of jProductionCritic in research involving machine learning. These formats can also be useful when performing experimental validations.

jProductionCritic is implemented in Java, in order to help make it as cross-platform and accessible as possible. This avoids forcing users to buy proprietary environments such as Matlab, and avoids the installation and linking problems that one can encounter with languages like C++.

The second main design objective of jProductionCritic is that it serve as a framework under which new error detection algorithms can be developed and deployed, and not only as a ready-to-use application. This is an important priority in encouraging future MIR research and development focusing on audio production. A strong emphasis was therefore put on designing jProductionCritic using a modular and highly extensible architecture to which new error checking algorithms can be easily added as plug-ins, with virtually no changes needed to the overall jProductionCritic processing infrastructure. Special attention was also paid to extensively documenting the code.

jProductionCritic is distributed as an integrated part of the jMIR [5] suite of MIR research software. This allows researchers to easily combine jProductionCritic's functionality with other jMIR components, such as the jAudio feature extractor or the ACE meta-learning system.

As with all jMIR components, jProductionCritic is free and open-source.

4. TECHNICAL ERRORS ASSESSED

Due to limited space, only broad overviews of jProductionCritic's main error detection algorithms are provided in the sub-sections below. Those wishing to read more details on any particular algorithm are encouraged to view the jProductionCritic manual or the Java class associated with the error type, both of which are available at http://jmir.sourceforge.net.

It is important to emphasize here that there are many important subtle subjective and artistic qualities that must be considered if one is to truly evaluate the production quality of a mix. Performing such an evaluation is well beyond the current technological capabilities of any automated system, and is best left to human experts, such as professional producers and instructors.

jProductionCritic therefore only attempts to detect clear objective technical errors, which many students and amateurs can still unfortunately produce many of. jProductionCritic is intended as a supplement and aid to human experts, not as a replacement for them.

Of course, even with this policy there can still be ambiguity with respect to certain error types. What might be considered unwanted noise in a classical flute recording, for example, might be part of a desirable production aesthetic in a flute sample used in an electronic dance track. Fortunately, jProductionCritic's diverse range of configuration settings makes it possible to easily modify the detection thresholds of given error types, or to disable them entirely, in order to match the various production aesthetics of different musical styles.

4.1 Digital Clipping

Digital clipping occurs when a signal exceeds the representational limits of its bit depth. Clipped signals are characterized by flat peaks and troughs, as samples are rounded to maximum and minimum values. Digitally clipped signals sound rough and distorted, and are almost never aesthetically desirable. Analog clipping, in contrast, can be desirable in certain styles of music, and is characterized by more curved peaks and troughs.

Digital clipping tends to occur in two main ways in student work: either the gain is set too high during recording or synthesis of an individual track, or the gains on individual tracks mixed onto the same channel are too high, such that the combined signals clip, even if none of the source signals are themselves clipped individually.

Although clipping detection is a common software feature, the popular implementation of simply flagging any samples at the representational limits is surprisingly naïve. This approach has two major problems. Firstly, a sample that actually should have a value at the representational limit is not in fact clipped, and such samples are to be expected in normalized signals. Secondly, students may attempt to hide clipping by reducing the master gain in the final mix, such that sample values fall below representational limits (and are thus not flagged) but the signal distortion caused by the clipping remains.

The approach used by jProductionCritic can overcome these two problems: if a number of adjacent samples beyond a threshold have an identical signal value (whether or not it is at the representational limit), then report clipping. The number of such consecutive samples gives an indication of clipping severity.

Despite its simplicity and effectiveness, other uses of this technique were not found in the literature, although

related counting techniques *at* the representational limit have been used. It should be noted that the literature also includes spectral approaches for detecting clipping, but these can be too sensitive for styles of music where analog clipping (or its digital simulation) is desirable.

4.2 Edit Clicks

An edit click occurs when an improperly treated edit is made, and can result in a discontinuity in the waveform that typically sounds like a click. This can happen when two signals are spliced together, or at the beginnings and ends of tracks (due to a sudden jump in the signal from or to silence). Although there are a number of techniques that can be used to avoid edit clicks, students and amateurs often neglect to use them.

Although the literature includes a substantial number of techniques for detecting instantaneous noise like clicks in general, it largely neglects edit clicks in particular. This is problematic from an educational perspective, as it is useful for students to know where imperfections in their work come from.

jProductionCritic uses a simple technique to detect edit clicks based on windows of only four samples: report an edit click if a signal jumps in value beyond a threshold from samples 2 to 3, but does not change in value beyond another threshold when progressing from samples 1 to 2 or 3 to 4. This approach is sensitive to improperly executed edits, is relatively impervious to false positives, and can also provide a severity measurement. This technique is also surprisingly absent from the literature, although related techniques considering much broader spreads than four samples are used for detecting instantaneous noise in general. Clicks at the beginnings and ends of tracks are simply found by looking for first and last samples far from zero, respectively.

It should be noted that this algorithm focuses only on a particular kind of edit error. It does not detect edit errors in general, of which there are many other types (e.g. a splice involving two segments of audio recorded under very different reverberant conditions).

4.3 Other Clicks, Pops and Instantaneous Noise

There are also many other types of undesirable instantaneous noise. Plosive pops due to the improper micing of a singer or noise when a needle jumps on a record are just two examples amongst many.

Although, there are a number of established techniques for detecting such problems, many of them tend to produce false positives. jProductionCritic's approach, which also produces some false positives but was still found to be the most effective during comparative experiments, is to high-pass filter the signal (due to the common assumption that unwanted noise will stand out most clearly against the musical signal in the high frequency range) and look for sudden and unusual peaks in the filtered signal's spectral flux.

4.4 Hums and Other Background Noise

Tracks can also be infiltrated by various types of sustained noise (as opposed to the more sudden and short-lived types of noise discussed above). Ventilation systems in recording environments and faulty cable shielding are two of the many possible sources. Detecting such noise in general can be particularly difficult, as it can be hard to distinguish from the musical signal. Although the literature does include certain sophisticated techniques, including approaches based on Hidden Markov models [8], these tend to be too limited in the styles of music to which they can be applied, so it was decided to use a simpler and more general technique11.

jProductionCritic's basic approach is to calculate the power spectrum of the audio and look for sustained peaks in particular frequency regions that are present in all or most of the audio. Extra weighting is applied if these peaks are still present in otherwise quiet parts of the signal. This approach tends to work reasonably well for detecting loud noise, but can miss quieter noise, and can result in false positives for those styles of music that feature sustained drones.

jProductionCritic also has specialized detectors that look for electrical noise (e.g. ground loops), a common problem in imperfectly configured or used studios. Such noise consists of a hum at the AC frequency of the power supply (and its integer multiples), which is generally either 50 Hz or 60 Hz, depending on where one is.

4.5 Phasing

Phasing is a problem that occurs when a signal is mixed with another signal that includes a phase delayed version of itself. This can occur, for example, when two omnidirectional microphones mapped to the same channel are too close to each other, or a single microphone is too close to an acoustically reflective surface. This results in cancellation or reinforcement of various frequencies, depending on the phase offset, which can result in a muddy tone.

Although the literature specifies several effective ways to detect phasing before mixing is carried out, it is much more difficult to automatically detect afterwards, and is easily confused with sometimes desirable comb filter effects like flanging. jProductionCritic's (admittedly limited) approach is to look for consistent troughs in the power spectrum of a track.

4.6 Dynamic Range

A common mistake made by students is to keep gains excessively low due to fear of clipping. Students then sometimes exacerbate this by forgetting to normalize their work during mastering (which can be desirable in order to achieve relatively consistent volumes). Another potential problem is that some tracks are insufficiently dynamically compressed (a desirable "hot" aesthetic in

pop styles) or, conversely, do not have enough dynamic range (a problem for styles such as classical music).

To address the first issue, jProductionCritic reports an error if the maximum absolute sample value is too far below the representational maximum. To address the other two problems, optional style-specific configuration settings can be specified to generate errors if the standard deviation of the windowed RMS across a track is too high or too low, respectively.

4.7 Stereo Balance and Channel Similarity

Some students do not include enough channel separation in their recordings to create a sufficient sense of stereo space, or even forget to specify panning settings at all. Additionally, students sometimes fail to properly balance the stereo channels, with the result that one stereo channel is consistently louder than the other.

jProductionCritic compares the left and right stereo channels and generates an error if the signal correlation is too high. It was found that this works better in general than spectral approaches. An error is also generated if the RMS of one channel as a whole is too high relative to the RMS of the other channel as a whole.

4.8 Other Errors Assessed

There are several additional errors that can be reported by jProductionCritic if desired. These include, among others:

- Too much silence (either absolute or at the noise floor) at beginnings and ends of tracks.
- Audio dropout.
- DC signal offset.
- Poor encoding parameters (e.g. low sampling rate or bit depth, lossy compression, etc.) in cases where high-quality masters should be used.

jProductionCritic also reports basic summary metadata (e.g. track length, audio encoding parameters, etc.).

5. VALIDATION EXPERIMENTS

Much of the error detection processing described above is based on thresholds, which the user has the option of specifying via configuration settings. However, it is important that it also be possible to apply jProductionCritic easily and effectively to arbitrary types of music without any user tweaking. To this end, experiments were performed to first arrive at good default configuration settings, and to then validate these settings' effectiveness.

In order to do this, music technology assignments were collected from multiple sections of three different courses over four semesters at Marianopolis College. Most but not all of the students involved were enrolled in the music program. Some of these assignments required students to make classical or jazz recordings using Pro Tools (in studio and live), and others required students to make mashups in any musical style using Audacity. The instructor's original (and later re-verified) corrections to the assignments served as the ground truth. In all, 110 assignments were collected.

Forty-four of these assignments were randomly selected and used to experimentally choose the error detection algorithms and tune their configuration settings. Once this was done, the remaining 66 assignments were then processed by jProductionCritic in order to verify that the configuration settings had not been overfitted to the tuning set. The results of this validation experiment are shown in Table 1:

	True Positives	False Positives	False Negatives
Human	499	0	8
jPC	452	38	55

Table 1. Results of the validation experiment comparing jProductionCritic's performance with expert human correction. Values indicate the total number of errors detected combined across all 66 validation assignments.

It is interesting to note that 8 true technical errors were detected by jProductionCritic that were wrongly missed during original human correction (they were found to be true errors upon manual secondary verification). It was also found upon secondary verification that, unlike jProductionCritic, the original corrector did not wrongly indicate any false errors.

Overall, it can be seen that jProductionCritic performed quite well. It found 89% of the true errors, compared with 98% found by the expert course instructor. Furthermore, 92% of the errors detected by jProductionCritic were in fact true errors. This is impressive when one recalls that the assignments were in a variety of musical styles, and were all processed using the same default configuration settings. It should be noted, however, that these results would be even more meaningful if students at different institutions with different instructors had been involved in the study.

With respect to the relative performance of the different error types, the algorithms for detecting phasing, background noise and, to a lesser degree, instantaneous noise (other than edit clicks) were by far the worst performers. It was difficult during the tuning stage to find configuration settings for them that would minimize false positives while also maximizing true positives, and this was reflected in the validation stage, where these three types of error detectors were responsible for 73% of all jProductionCritic's false positives and false negatives. The other algorithms performed relatively similarly (and successfully).

While jProductionCritic is still not as good as a human expert, it did perform well enough to be at least comparable, and it certainly caught many errors missed by the students. It is sufficiently good to serve as a time-

saving and verification tool for teachers, and can effectively provide students and amateur producers with valuable feedback for improving their work.

6. CONCLUSIONS AND FUTURE RESEARCH

It is hoped that jProductionCritic will help to address several underserved needs: the absence of integrated open-source production error checking software in general; the absence of software intended to meet the educational needs of audio production students in particular; and the relatively limited attention given to both production and education software in the MIR community to date.

From a research perspective, jProductionCritic has the advantages of including a number of original error detection algorithms and of being fully open-source. Unlike almost all other integrated production error detection software, its algorithms are not proprietary black boxes. Furthermore, jProductionCritic has a modular and easily extensible design that is intended to encourage its use as a framework for future MIR research on developing additional error checking algorithms.

From an applied perspective, jProductionCritic is the only known production-oriented software that is intended to meet the specific needs of education, and looks for many more errors than any other known general integrated system. Moreover, the validation experiments found that the software performs more than well enough to be used successfully in practice.

The first priority for future work is to port jProductionCritic to a standard DAW plug-in format, such as VST or Nyquist. This will greatly increase its accessibility to students. Another priority is to implement it as a Vamp plug-in so that it can be used with Sonic Visualiser [2], which would increase the scope and clarity of information that could be shown to users by supplementing the Audacity Label Tracks currently used.

It would also be useful to implement an interface with which teachers could specify a grading scheme, so that assignments could be marked more easily, and students could have an idea what grades they will receive before submitting. Of course, it is important to reserve room for the instructor's subjective judgment when doing this.

There are also many other useful error detection algorithms that remain to be implemented, including detection of poor EQ, too much or too little reverberation, excessive performance artifacts, etc. There is also still plenty of potential to refine and improve the existing algorithms, especially those that performed poorly in the validation tests, perhaps with the ultimate goal of making jProductionCritic more useful in professional contexts.

jProductionCritic, its code and documentation can all be downloaded for free from: http://jmir.sourceforge.net.

7. REFERENCES

[1] Barlett, B. and J. Barlett. 2009. *Practical recording techniques: The step-by-step approach to professional audio recording.* Burlington, MA: Focal Press.

[2] Cannam, C., C. Landone, M. Sandler, and J. P. Bello. 2006. The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. *Proceedings of the International Conference on Music Information Retrieval*. 324–7.

[3] Huber, D. M., and R. E. Runstein. 2009. *Modern recording techniques.* Burlington, MA: Focal Press.

[4] Kühhirt, U. 2008. *A/V Analyzing Toolbox.* Retrieved 8 May 2013, from http://www.idmt.fraunhofer.de/en/Service_Offerings/technologies/a_d/av_analyzing_toolbox.html.

[5] McKay, C. 2010. Automatic music classification with jMIR. *Ph.D. Dissertation*. McGill University, Canada.

[6] Montecchio, N., and A Cont. 2011. Accelerating the mixing phase in studio recording productions by automatic audio alignment. *Proceedings of the International Society for Music Information Retrieval Conference.* 627–32.

[7] Owsinski, B. 2013. *The mixing engineer's handbook.* Independence, KY: Course Technology PTR.

[8] Sabri, M., J. Alirezaie, and S. Krishnan. 2003. Audio noise detection using hidden Markov model. *Proceedings of the IEEE Workshop on Statistical Signal Processing*. 637–40.

[9] Scott, J., and Y. E. Kim. 2011. Analysis of acoustic features for automated multi-track mixing. *Proceedings of the International Society for Music Information Retrieval Conference.* 621–6.

[10] Vaseghi, S. V. 2009. *Advanced digital signal processing and noise reduction.* Chichester, West Sussex: Wiley.

[11] Witten, I. H., E. Frank, and M. A. Hall. 2011. *Data mining: Practical machine learning tools and techniques.* New York: Morgan Kaufman.

[12] *Audiofile-Inspector & Digital Error Checker*. Retrieved 8 May 2013, from http://www.cube-tec.com/products/quadriga/quadriga-features-system-integration/quadriga-features-audiofile-inspector-and-digital-error-checker.

[13] *Digital Audio Error Detection.* Retrieved 8 May 2013, from http://digauderrodetec.sourceforge.net.

COMBINING TIMBRIC AND RHYTHMIC FEATURES FOR SEMANTIC MUSIC TAGGING

Nicola Orio
Department of Cultural Heritage
University of Padua, Italy
orio@dei.unipd.it

Roberto Piva
Department of Information Engineering
University of Padua, Italy
piva.roberto.88@gmail.com

ABSTRACT

In this paper we propose a novel approach to music tagging. The approach uses a statistical framework to model two acoustic features: timbre and rhythm. A collection of tagged music is thus represented as a graph where the states correspond to the songs and the models probabilities are related to the timbric and rhythmic similarity. Under the assumption that acoustically similar songs have similar tags, we infer the tags of a new song by adding it to the graph structure and observing the tags visited in acoustically meaningful random walks. The approach has been tested using the CAL500 dataset, with encouraging results in terms of precision.

1. INTRODUCTION

The ability of humans to associate tags or generic metadata with multimedia content is a difficult task to simulate, because it relies on subjective judgments and on the identification of connections between abstract concepts. In the case of music content, tagging has always been a feature of online streaming services like LastFM, Apple Genius, Pandora or Grooveshark, since these services rely on music descriptors to deliver the right songs to the right user. Their tags have different origins though: while Pandora pays music experts to annotate music with reliable and expressive terms, the other services rely on user generated tags and playlists and exploit statistics tools like collaborative filtering [8] for annotating and recommending music. A typical problem of manual tagging regards the annotation of new items. While songs by renowned artists may easily get proper tagging by their advertisers, there are thousands of tracks – for instance produced by small independent labels – that are likely to be unreachable because of the lack of good descriptors.

Another interesting problem is the long tail distribution: the analysis of listening charts highlights that the distribution of play counts over artists follows a power law. This means that a restricted number of artists gets the majority of play counts. However, the total play counts of the long tail largely outnumbers the one of the top artists. This situation is also common for the global charts of a music streaming service and has a reflection on how these services make money [6]. The problem is related to tagging because in most of the times the long tail consists of poorly tagged music pieces.

Early works on automatic music tagging addressed the recognition of the music genre [11, 12, 25]. The subjectiveness of "genre" classification led researchers to propose a novel genre taxonomy [18] and to identify music tagging as a more broad concept that includes other types of information – like track's pace and dance-ability. To this end, tagging has also been defined as "music semantic annotation" [10, 24]. Other approaches have explored the association between related tags [16], or the inclusion of mined social tags to infer some relation between tags and audio features [2].

A variety of features were considered in search of the right combination to correctly annotate novel songs. First attempts started with Mel Frequency Cepstral Coefficients (MFCC), which were successfully used for music classification while improvements were obtained with the aid of other sources such as social tags [5], or temporal features [15]. More recent work investigates the improvement of MFCC by using Principal Component Analysis [13] while learning frameworks have been applied as well, such as Support Vector Machines [3] and Artificial Neural Networks [9].

This paper describes a method for semantically tagging music clips by exploiting timbre and rhythm features represented in a single statistical framework, where audio features are related to different model parameters that have been developed on top of a previously defined retrieval model based on content and context descriptors [17].

2. THE TAGGING MODEL

Our basic assumption is that the relation between tags and music features can be better exploited by using multiple features in a single tagging model. Our framework, inspired by Hidden Markov Models (HMMs), accounts for music similarity in terms of two different audio features: timbre represented by MFCC and rhythm represented by Rhythm Histograms (RH).

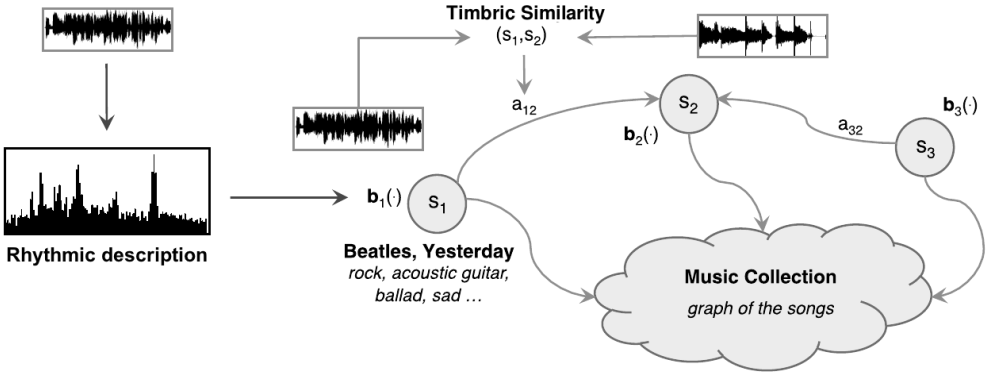


Figure 1. A graphical representation of the tagging model

2.1 Model Elements

Given the close relationship with HMMs, we introduce the different elements of the model using, when possibile, the same notation introduced in [21].

States: A song is represented as a state S_i from a set $\mathcal{S} = \{S_1, \ldots, S_N\}$, and state at time t is denoted as $s(t)$.

Prior Probabilities: The query song to be tagged is added to the model as state S_q, requiring the computation of the transition-wise similarity with each song in the collection to obtain a_{qi} for all i. Each path in the model is forced to start from the query song.

Transition Probabilities: They are related to the acoustic similarity between the songs using one of the two features, thus they can be related either to timbre or to rhythm similarity. Referring to the usual naming convention for HMMs, a_{ij} stands for the acoustic similarity of song S_i with song S_j.

Observation Probabilities: They are related to the acoustic similarity between the songs and the query song, computed from a different music dimension than the on used for transitions. We introduce symbol $\phi_i(q)$ as the probability for song S_i of emitting the audio features of query song S_q.

Tags: The information regarding the tags of the known songs is stored in a separate variable, named $b_i(w)$, that associate tag w to song S_i in the collection. We assume that the vocabulary of tags consists in $|\mathcal{V}|$ different symbols belonging to $\mathcal{V} = \{w_1, w_2, \ldots, w_{|\mathcal{V}|}\}$. The vector b_i is in binary form, it contains a 1 in position j if tag j is present, 0 otherwise.

The model is then exploited to generate random walks across the collection, starting from the song to be tagged. Transition probabilities guarantee acoustic consistency of the songs in the paths, while observation probabilities guarantee acoustic similarity with the query song. A graphical representation of the model can be found in Figure 1.

Although in principle $a_{ij} > 0$ for each (i, j) pair, we artificially keep, for each song S_i, only a subset $\mathcal{R}(S_i)$ of non-zero transitions. That is, we keep the top P highest transition probabilities for each state, and we set to 0 the remaining transitions, with an increase in model scalability. The value of P has been set to 10% of the global number of the songs. Transitions are normalized to maintain the stochastic properties.

As a consequence of the analogy with HMMs, the observation's feature plays a more important role than the transition's feature in discriminating the songs. This happens because observation's feature is compared to the query at each step in a random walk across the model, while the influence of transition similarity decreases as the path gets longer. Having the tags in a separate structure let us grow the model easily if there is a novel tag.

2.2 Acoustic Features

We use MFCC to capture the timbre signature of each song. MFCC are computed using 23 ms, half-overlapping windows over 6 seconds of music after the first 30 seconds. We thought 6 seconds may be a good amount of data to be used for automatic tagging for online services and for fast tagging of large collections. The output is a 13-sized vector for each window, but since we include the first and second derivatives, we end up with a matrix of 39 elements multiplied by the number of windows.

We use Rhythm Histograms (RH) and Rhythm Patterns (RP) to hold information regarding the rhythm of a given piece of music, as described in [19, 22] as *psychoacoustically motivated Rhythm Patterns*. Rhythm Patterns themselves are a variation of Fluctuation Pattern [7]. RH are computed using the same 6 seconds of audio as MFCC. Simplifying some psychoacoustic adjustments steps in the definition of RP and slightly varying the calculation parameters, we ended up with a matrix descriptor of 120 modulation frequencies by 24 frequency bins.

2.3 Similarity Measures

We adopted two approaches to compute the similarity between two songs. The first one considers each frame as a fixed-width word in a dictionary [14]. We calculate mean

and covariance of the (letters of the) words over the length of the song and the result is a representative multivariate Gaussian distribution of the song. To compare these two distribution we use the Kullback-Leibler divergence (KL divergence):

$$KL(p\,||\,q) \equiv \int p(x) \log \frac{p(x)}{q(x)}\,\mathrm{d}x, \tag{1}$$

where $p(x)$ and $q(x)$ are the two distributions. The KL divergence is then transformed to a $[0, 1]$ value (with 1 representing two identical songs) by exponentiating the divergence:

$$sim(p, q) = e^{-\gamma KL(p\,||\,q)}, \tag{2}$$

where γ is a parameter to be tuned (for this work it has been set to 0.009, see [20]). In this paper this similarity is referred to as "single gaussian".

We investigated also a number of alternative similarity measures [4]: the Cosine similarity and its varied version Modified Chord, a similarity based on Euclidean distance, two biology-based similarities (Bray-Curtis and Ruzicka), the Similarity Ratio, the Kulczynski similarity and the exponentiated version of the discrete KL-divergence (very similar to the one from Equation 2). This second group of measures is based on a vectorized version of the matrix representations. For MFCC we keep the absolute value of the time-wise sum, obtaining a 39-sized vector representing the average timbre across the piece of music. For the RH we sum along the frequency bins (barks) to obtain a 118 sized vector representing the influence of each modulation frequency in the $[0, 10]$Hz range.

2.4 Querying the model

We developed a modified version of the Viterbi algorithm, which application to HMMs can be found in [21] where δ represents the probability of the optimal path and ψ is used to keep track of its actual state sequence. We refer to the query song using the subscript q (e.g. S_q for the song, $b_q()$ for the tags, and so on).

Initialization: As introduced in Section 2.1 the song to be tagged S_q is inserted in the model and the initialization step is, for all $i = 1, \ldots, N$:

$$\delta_1(i) = \begin{cases} 1 & i = q \\ 0 & i \neq q \end{cases} \tag{3}$$

$$\psi_1(i) = 0. \tag{4}$$

Recursion: Here we present a variation from the original Viterbi. For $t = 2, \ldots, T$, and $i = 1, \ldots, N$:

$$\delta_t(i) = \max_{1 \leq i \leq N} [\delta_{t-1}(i)\, a_{ji}]\, \phi_q(i) \tag{5}$$

$$\psi_t(i) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i)\, a_{ji}] \tag{6}$$

$$a_{ri} = \frac{a_{ri}}{\eta} \text{ for } r = \psi_t(i),\ \eta = 10. \tag{7}$$

Where $\phi_q(i)$ is the similarity between S_q and S_i and can be computed using one of the possible similarity measures introduced in Section 2.3. Equation 7 aims at preventing looping by lowering the probability of performing a transition twice [20].

Decoding: The most probable path is computed as in the original Viterbi algorithm:

$$s(t)^* = \begin{cases} \arg \max_{1 \leq i \leq N} [\,\delta_t(i)\,] & \text{if } t = T \\ \psi_{t+1}(s(t+1)^*) & \text{if } t = T-1, \ldots, 1, \end{cases} \tag{8}$$

$$p(t)^* = \begin{cases} \max_{1 \leq i \leq N} [\,\delta_t(i)\,] & \text{if } t = T \\ \delta_t(s(t)^*) & \text{if } t = T-1, \ldots, 1. \end{cases} \tag{9}$$

Where $p(t)^*$ represents the probability of the optimal path at each step. The parameter T is the maximum length of the subpath, which is discussed in the following section.

Tagging: The tags of states belonging to the optimal path are used to tag the query song, because they are likely to be acoustically similar to the query and to share some of their textual descriptors. We take the T songs extracted by Equation 8 and we calculate a vector of tag weights $b_q(w)$ for all tags in $j = 1, \ldots, M$ as in:

$$b_q(w) = \sum_{t=1}^{T} b_t(w)\omega(t). \tag{10}$$

Where $\omega(t)$ is a decreasing monotonic function of the path position i. That is, tags from songs acoustically similar – reached early in the optimal path – to the query should have more importance than songs that are not that similar – observed at the latter steps of a random walk.

Iteration: To keep a high transition-wise similarity, the procedure is split in substeps. After having computed a path of length T, with $T = 4$ giving the best experimental results, we restart Viterbi decoding from the initialization step without resetting the modified transitions (i.e. keeping the effects of Equation 7). The procedure is iterated a number of times until enough songs are visited in order to correctly infer the tags for the query song. The final weight of a tag is computed as the sum of the weights computed at each iteration.

It has to be noted that there is no real control of the paths extracted at each iteration, in a sense that there can be songs that appear multiple times on different iterations or maybe multiple times on the same iteration. This as a desired behavior: if a song is chosen multiple times we assume that it is particularly relevant to the query, and so its tags may be better related to the it. In fact, we sum the tags contribution of these songs every time they are chosen, regardless of the number of times and the position in the path where they appear.

The proposed approach allows us to exploit the influence of two features at once. It can be noted that we could calculate query's tag by simply using all the neighbor songs computed from a forward exploration of the model starting from the query. The advantage of using Viterbi decoding relies on the fact that it finds an acoustically meaningful "music path" from the query, which helps us avoiding non-related songs. Another important advantage is that we can

also decide which weight we want to assign to each song in the path, which could lead to better results.

2.5 Weighting the Tags

As of the weighting function in Equation 10 we explored different options, that have been tested experimentally. For each time step $t = 1, \ldots, T$ the function $\omega(t)$ can be computed according to:

Path probability: As a first option, the path probability at step t can be used directly a the tag weight, using $\omega(t) = p(t)^*$.

Linear decay: The relevance of tags can decrease linearly with the length of the path required to obtain them, according to $\omega(t) = 1 - m(t-1)$, with $0 < m < 1$.

Exponential decay: Since the probability of a path across HMMs decreases exponentially with the length of the path, tag weight can be computed also according to $\omega(t) = a^{(t-1)}$, with $0 < a < 1$.

Hyperbolic decay: In order to obtain, for small values of T, intermediate weights between linear and exponential, tag weight can be also computed as $\omega(t) = 1/t$.

3. RESULTS

An automatic tagging system is expected to put meaningful tags on novel songs in a reliable way. We have already seen the importance of automatic tagging in the introduction and we wanted to test how our model performs in this difficult task.

3.1 Data Source

For the experimental evaluation we focus on the quality of the source data, as we need to rely on it to put the right tags to songs. For these reasons we have chosen to use the CAL500 dataset [23], which consists of 502 popular songs of Western music by different artists. Songs from this dataset have been tagged, through a controlled survey, by at least three human annotators each. The semantic vocabulary consists of 149 tags spacing from genre classification (e.g. "rock", "pop") to vocal and acoustic characteristics (e.g. "female lead vocals"), as well as emotions (e.g. "aggressive") and song usages (e.g. "studying"). The survey results is a binary annotation of each song.

Acoustic features (i.e. MFCC and RH) have been computed from a degraded version of the clips in the dataset, with an encoding quality which was still high enough for our purposes. The availability of the audio motivates the choice of CAL500. Moreover, we are more interested in the reliability of the tagging procedure, so we prefer to evaluate our approach with a small yet controlled collection. Experimental evaluation with larger collections, such as CAL10k or Magnatagatune, will be part of our future work.

3.2 Evaluation Measures

What the users expect from an automatic tagging system is that proposed tags are relevant, in a sense that they truly describe the content of the song, and the tags are complete, so there is no lack of information. Any extra tag is counted as an error, or at least as noise. We also expect that the system has to be concise, that is, if the output of the system is a ranked list of tags, where the rank is a relevance measure, we may want to keep only the top most tags as the query result. In other words, we need to measure whether the system is able to propose the most relevant tags at the top of the ranked list, while other tags (not-so-relevant ones and wrong ones) should have lower ranks. With this aim in mind we choose to use the precision metric: we measured the precision at 10 (P@10), which reports the fraction of relevant tags of the top 10 results from the ranked list, and the mean average precision (MAP), which averages the precision at each point of the ranked list of tags.

3.3 Testing Procedure

Since the role of this model is to tag a new song we can test it using the ground truth (the CAL500 dataset) in a leave-one-out fashion on all the songs minus one. What we have done is calculating the acoustic similarity for each song in the dataset, and, in turns, we simulated the querying of each song against the rest of the dataset.

To this end, the tagging procedure ignores the tag contribution from the query song as it assumes that it does not have tags on it.

3.4 Parameters Tuning

The tagging model as proposed in this thesis has some parameters which have to be tuned.

We tested some combinations of the values of T (length of the optimal path) in relation to the number of effectively retrieved songs per path and the total number of iterations (see section 2.4). What we have seen is that short paths give better results, and the number of effectively retrieved songs should be as close as possible to the length of the path: this could mean that the conservation of acoustic similarity is preferred over the number of retrieved results. We ended up choosing a path length of 4 with 3 retrieved song per iteration. This approach can produce iterations where as little as 1 song are retrieved, that is, there can be a loop of length 3 with the query and one song. Other combinations of path length and retrieved songs led us to worse results MAP-wise.

Regarding the total number of iterations of the Viterbi algorithm we have seen that, for our data, the best results were obtained with 9 iteration. Of course the influence of these parameters should be further discussed and optimal values may change for different datasets or as the model grows integrating the tagged songs.

3.5 Experimental Results

The approach was tested using MFFC for transitions and RH for observations and viceversa. Moreover, we wanted to evaluate the effect of individual features and compare them with a baseline approach. To measure this we tried three strategies: first, we measured the influence of the observation similarity by imposing uniform transitions. Then

we measured the influence of the transition similarity by uniforming the observation probabilities. The last test simulated a completely random walk by uniforming both probabilities, in order to obtain a true baseline.

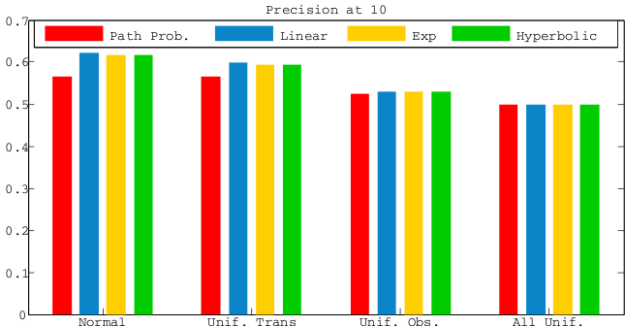


Figure 2. P@10 from MFCC on transitions with KL (single gaussian) similarity, RH on observations with Modified Chord similarity, comparing the effects of different weighting and the single features. Note: The span of this and the following figures is set in the range $[0; 0.7]$ in order to better appreciate the differences between the values

In Figure 2 are shown the results of P@10 comparing different combinations of uniform probabilities and tag weighting schemes. We can see that the exploitation of both features gives consistently better results than uniform probabilities. In turn, using only observations gives better results than using only transitions while the baseline gives always poorer results. It can also be observed that linear weighting performs slightly better than all weighting strategies and that the simple use of path probability – although a natural choice – gives the lowest results.

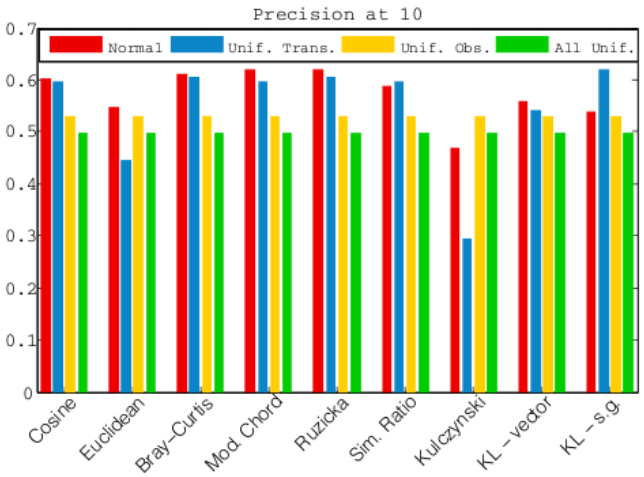


Figure 3. P@10 from MFCC on transitions with KL (single gaussian) similarity, RH on observations with multiple similarities, with linear tag weighting

In Figure 3 and Figure 4 are shown the results of P@10 comparing different similarity measures, using linear tag weighting. Our best results lead to over 0.6 precision, which means that if we pick a random song and we keep only the top 10 proposed tags, we can expect 6 of them to be correct on average, which is a good result for the first implementation of a novel approach. As it can be seen, not all similarities combinations have the same ratio between "normal" approach and uniform probabilities approach. For some combinations "normal" approach has worse results than "uniform transition probabilities" which underlines the importance of the choice of the distance measure.

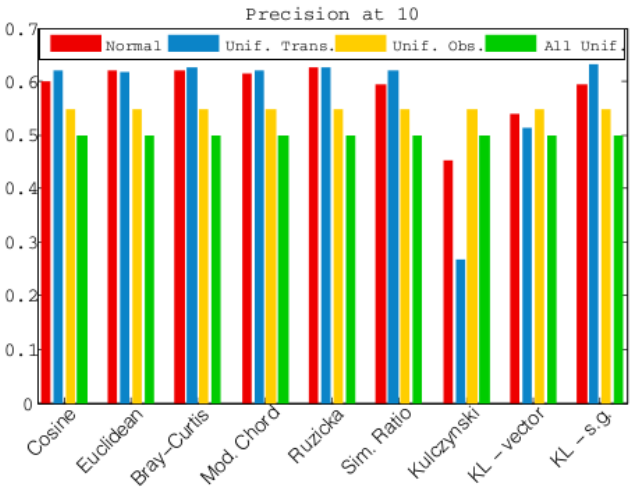


Figure 4. P@10 from RH on transitions with KL (single gaussian) similarity, MFCC on observations with multiple similarities, with linear tag weighting

Our MAP performance is in line with the one of other works at initial stage described in the literature. Table 1 shows a MAP results summary for the same configuration as Figure 3 and Figure 4 respectively.

MFCC on transitions (KL - single gaussian), RH on observations (Modified Chord)				
	Normal	**Un. Tr.**	**Un. Obs.**	**All Un.**
Path P.	0.481	0.474	0.442	0.442
Linear	**0.536**	0.507	0.450	0.442
Exp.	0.533	0.504	0.451	0.453
Hyperb.	0.533	0.504	0.452	0.450
RH on transitions (KL - single gaussian), MFCC on observations (Euclidean)				
	Normal	**Un. Tr.**	**Un. Obs.**	**All Un.**
Path P.	0.478	0.489	0.447	0.446
Linear	**0.528**	0.523	0.465	0.442
Exp.	0.526	0.520	0.468	0.453
Hyperb.	0.526	0.520	0.467	0.450

Table 1. Mean average precision summary table

4. CONCLUSIONS

We describe a novel approach for semantic music tagging based on a statistical framework that combines two music features: timbre and rhythm. Tagging is based on a modified Viterbi algorithm to carry out iterated random walks in the graph that represents a collection of tagged songs. The approach was evaluated using the CAL500 dataset. Experiments have shown encouraging results in terms of precision at 10 and Mean Average Precision of the ranked lists of tags. Performance contribution of each feature has been also measured separately and compared to

a random baseline. The effects of different similarity measures for observations have been tested as well, together with four approaches to weight tags according to the number of steps required to obtain them. To the best of our knowledge, we think that our performances are in line with other early stage approaches. As pointed out in [1], purely audio-based approaches are instrinsically limited because they cannot capture all the music dimensions perceived by listeners. We think that additional parameter tuning, possibly using other collections, will give further improvements before the "glass ceiling" is reached.

One issue that will be addressed in future work is the effect of loops in the optimal path, which at the moment is minimized with the modified Viterbi algorithm (Equation 7) but can be improved by alternative strategies to modify the transition probabilities. The current tagging procedure suggests a way to grow the collection by adding the newly tagged song to the graph, thus we aim also at measuring how performances degrade with the increase of the collection size.

5. REFERENCES

[1] J.-J. Aucouturier and Pachet F. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1), 2004.

[2] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: A model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2):115–135, 2008.

[3] S. Bourguigne and P.D. Agüero. Audio tag classification using feature trimming and grid search for SVM. In *Proc. of MIREX*, 2011.

[4] S.-H. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.

[5] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. *Advances in neural information processing systems*, 20(20):1–8, 2007.

[6] A. Elberse. Should you invest in the long tail? *Harvard business review*, 86(7/8):88, 2008.

[7] H. Fastl. Fluctuation strength and temporal masking patterns of amplitude-modulated broadband noise. *Hearing Research*, 8(1):59–69, 1982.

[8] D. Goldberg, D. Nichols, B.M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[9] P. Hamel. Multi-timescale pmscs for music audio classification. In *Proc. of MIREX*, 2012.

[10] M. Hoffman, D. Blei, and P. Cook. Easy as cba: A simple probabilistic model for tagging music. In *Proc. of ISMIR*, pages 369–374, 2009.

[11] T. Li, M. Ogihara, and Q. Li. A comparative study on content-based music genre classification. In *Proc. 26th ACM SIGIR conference*, pages 282–289. ACM, 2003.

[12] T. Lidy and A. Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proc. of ISMIR*, pages 34–41, 2005.

[13] S.-C. Lim, K. Byun, J.-S. Lee, S.-J. Jang, and Moo Young Kim. Music genre/mood classification. In *Proc. of MIREX*, 2012.

[14] M.I. Mandel and D.P.W. Ellis. Song-level features and support vector machines for music classification. In *Proc. of ISMIR*, pages 594–599, 2005.

[15] M.I. Mandel and D.P.W. Ellis. Multiple-instance learning for music information retrieval. In *Proc. of ISMIR*, pages 577–582, 2008.

[16] R. Miotto and G. Lanckriet. A generative context model for semantic music annotation and retrieval. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(4):1096–1108, 2012.

[17] R. Miotto and N. Orio. A probabilistic model to combine tags and acoustic similarity for music retrieval. *ACM Transactions on Information Systems (TOIS)*, 30(2):8, 2012.

[18] F. Pachet, D. Cazaly, et al. A taxonomy of musical genres. In *Proc. Content-Based Multimedia Information Access (RIAO)*, pages 1238–1245, 2000.

[19] E. Pampalk. Islands of music: Analysis, organization, and visualization of music archives. *Master's thesis, Vienna University of Technology*, 2001.

[20] R. Piva. Combining timbric and rhythmic features for semantic music tagging. *Master's thesis, University of Padua, Padova, Italy*, 2013.

[21] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[22] A. Rauber, E. Pampalk, and D. Merkl. The som-enhanced jukebox: Organization and visualization of music collections based on perceptual models. *Journal of New Music Research*, 32(2):193–210, 2003.

[23] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Towards musical query-by-semantic-description using the CAL500 data set. In *Proc. of ACM-SIGIR*, pages 439–446. ACM, 2007.

[24] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):467–476, 2008.

[25] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *Speech and Audio Processing, IEEE transactions on*, 10(5):293–302, 2002.

THE AUDIO DEGRADATION TOOLBOX AND ITS APPLICATION TO ROBUSTNESS EVALUATION

Matthias Mauch **Sebastian Ewert**
Queen Mary University of London, Centre for Digital Music
{matthias.mauch, sebastian.ewert}@eecs.qmul.ac.uk

ABSTRACT

We introduce the Audio Degradation Toolbox (ADT) for the controlled degradation of audio signals, and propose its usage as a means of evaluating and comparing the robustness of audio processing algorithms. Music recordings encountered in practical applications are subject to varied, sometimes unpredictable degradation. For example, audio is degraded by low-quality microphones, noisy recording environments, MP3 compression, dynamic compression in broadcasting or vinyl decay. In spite of this, no standard software for the degradation of audio exists, and music processing methods are usually evaluated against clean data. The ADT fills this gap by providing Matlab scripts that emulate a wide range of degradation types. We describe 14 degradation units, and how they can be chained to create more complex, 'real-world' degradations. The ADT also provides functionality to adjust existing ground-truth, correcting for temporal distortions introduced by degradation. Using four different music informatics tasks, we show that performance strongly depends on the combination of method and degradation applied. We demonstrate that specific degradations can reduce or even reverse the performance difference between two competing methods. ADT source code, sounds, impulse responses and definitions are freely available for download.

1. INTRODUCTION

Advances in audio and music processing technology depend on the ability to evaluate the success of different methods against each other, and to show in which ways they behave differently. Consequently, better evaluation methods and the creation of new, bigger ground-truth data sets are topics of lively research and debate [1, 7]. Considerably less attention is usually devoted to the question of whether the processing methods are robust against degradations in audio quality. For example, the robustness of automatic speech recognition systems is often only tested by adding noise or a limited set of additive environmental sounds [6], or by filtering the audio [11], e.g. using the FaNT toolkit [1]. Similarly, in music informatics only a few studies consider insufficient or variable audio quality, exceptions including two studies on the influence of MP3 compression on genre classification [2] and onset detection [12], and a study of onset detection methods under reverberation [21].

Real-world applications often require robustness against a broad range of signal degradations, as the quality of audio recordings is often entirely unknown. For example, mobile music-processing apps suffer from low quality microphones in smartphones [3], in addition to noisy recording environments and unpredictable non-linear processing in the phones' DSP chips. Similarly, guaranteeing a consistent audio quality in a series of recordings is almost impossible in exploratory ethnomusicology [13, 18]. Audio effects applied at radio stations can lead to significant challenges for audio identification methods. Many such applications are likely to receive input audio of mixed levels of degradation. Therefore, studying the influence of various signal degradations on a given method can aid the detection of problems and the development of appropriate countermeasures.

In this paper, we describe the Audio Degradation Toolbox, which offers a well-defined way of applying a wide range of audio degradations, from basic sanity checks over simple parameter sweeps to highly-specific degradations imitating audio sourced from radio, smartphones or vinyl records. The breadth of degradations goes beyond the application of noise and MP3 degradations currently used in the literature. Using four typical music informatics tasks as an example, we show how the toolbox can be used to analyse the behaviour of methods under degradation, compare the performance of competing approaches in various scenarios, locate conceptual weaknesses in methods and develop suitable countermeasures. To facilitate integration into existing evaluation workflows, the toolbox provides means to adjust ground-truth annotations to compensate for temporal distortions resulting from degradations. The combination of audio and ground-truth transformation functionality makes the toolbox an easy-to-use instrument to boost the informative value of systematic evaluations.

The rest of the paper is organised as follows: Section 2 describes the components of the toolbox itself and how they work together. In Section 3 we examine the behaviour of several music informatics methods under audio degradation. Section 4 summarises and concludes the paper.

[1] http://dnt.kr.hs-niederrhein.de/download.html

```
audio_out = degradationUnit_addNoise(
      audio, samplingFreq)
```

a) only audio processing

```
parameter.noiseColor = 'brown';
[audio_out, timestamps_out] =
   degradationUnit_addNoise(audio, samplingFreq,
                            timestamps, parameter)
```

b) with timestamps and a parameter variable

Figure 1: Example calls to a degradation unit function.

Room Impulse Responses [20]: Great Hall, Classroom, Octagon.
Smartphone Impulse Responses (recorded for the ADT: exponential sine sweep, inverse filtering):
Google Nexus One microphone, Google Nexus One speaker.
Vinyl (recorded for the ADT: inverse filtered sine sweep output from iZotope Vinyl 1.73b plugin): Vinyl Player 1960 (raw and smoothed)

(a) Impulse Responses

Pub Sound Environment [9].
Vinyl: recorded for the ADT using the iZotope Vinyl plugin.

(b) Sounds

Table 1: Data included in the ADT.

2. THE AUDIO DEGRADATION TOOLBOX

The Audio Degradation Toolbox (ADT) consists of Matlab code for the controlled degradation of audio signals, and for the adaptation of ground-truth to the degraded audio. Additionally, some scripts are provided to facilitate batch processing. The source code is hosted on SoundSoftware.org [2] and is freely available under a GPL license. The ADT is based on two concepts: *(a) degradation unit*: a Matlab function with optional parameters that performs a simple audio (and ground-truth) transformation, *(b) degradation*: parametrisation of several degradation units into a chain. The remainder of this section details the rationale and implementation of the degradation units and degradations.

2.1 Degradation Units

The ADT contains 14 basic degradation units implemented as Matlab functions. All degradation units use the same calling conventions. Figure 1a illustrates the simplest case, in which a degradation unit is called with two arguments: an audio data matrix (`audio`) and a scalar providing the audio sampling frequency (`samplingFreq`); the function returns a matrix of degraded audio data (`audio_out`) with the same sampling frequency as the input audio.

Some degradations change the speed of the audio or warp time in other ways (see detailed descriptions, below) with the result that timestamps annotated with respect to the original audio will no longer be valid. In particular, ground-truth annotated on the original audio would be meaningless on such degraded audio. Therefore, all degradation units implement a timestamp transformation that maps the input timestamps to the timeline of the output audio. The transformed ground-truth can then be used for evaluation on degraded audio data. Figure 1b illustrates how the user can provide additional timestamps (`timestamps`) and receive an output variable containing adjusted timestamps (`timestamps_out`). For convenience, all degradation units have a default setting, but they will typically be called with customised parameter settings, as in Figure 1b. More on parameters in Section 2.2.

What follows are short descriptions of the individual degradation units contained in the ADT. Degradation units that introduce a time warping are marked with the letter W. More documentation can be found as comment headers in the respective source code files.

[2] `http://code.soundsoftware.ac.uk/projects/audio-degradation-toolbox`

Add Noise. Adds artificial random noise of different 'colours' (brown, pink, white, blue, violet) to the signal at a specified signal-to noise ratio (SNR). Implemented as white noise plus filter.

Add Sound. Adds an arbitrary signal to a given signal at a specified SNR and takes care of sampling rate conversions (the ADT comes with two sounds, see Table 1b).

Attenuation. Makes the signal quieter by a specified number of decibels or a factor specified.

Aliasing. Purposeful violation of the sampling theorem: the signal is down-sampled to a specified sampling rate without lowpass filtering. The original sampling rate is restored using a regular resampling method (with filtering).

Clipping. Normalises the audio signal such that a specified number or percentage of samples is outside the interval $[-1, 1]$, and each resulting sample x is clipped to $\mathrm{sign}(x)$.

Delay (W). Pads the beginning of the input audio with a specified number of zero samples. The timestamps are delayed accordingly.

Dynamic Range Compression (W). Applies a signal-dependent normalisation to the audio signal, reducing the energy difference between soft and loud parts of the signal. The implementation is adapted from [22], allowing specification of several parameters including attack, release, delay and slope. As the delay parameter introduces a constant time delay, the output timestamps are adjusted accordingly.

Apply Impulse Response (W). Filters the signal using one of six natural impulse responses (IR) provided with the toolbox (see Table 1a). As this process introduces an IR-dependent delay, timestamps are adjusted using the IR's mean group delay.

High-pass Filter. Applies a linear-phase high-pass filter constructed using the Hamming window method. Parameters are stop and pass band edge frequencies. The output is cropped to achieve zero phase, hence no time-warp.

Low-pass Filter. Analogous to High-pass Filter.

MP3 Compression. Compresses the audio data to an MP3 file with a specified bit rate using the Lame encoder [3]. The encoder and decoder delays are compensated for by also using Lame as the decoder.

Saturation. Applies the transformation $x \leftarrow \sin(\pi x)$ multiple times (as specified) to each sample $x \in [-1, 1]$, which drives the sample towards the margins of -1 or 1, resulting in a simple imitation of a saturation effect.

[3] `http://lame.sourceforge.net`

```
function degr_cfg = demoDegradation()
  degr_cfg(1).methodname = 'degradationUnit_1';
  degr_cfg(1).parameter.someParam1 = 3;
  degr_cfg(2).methodname = 'degradationUnit_2';
  degr_cfg(2).parameter.someParam2 = 4;
```

a) Specifying a degradation.

```
audio_out = applyDegradation(
           'demoDegradation',audio,samplingFreq);
```

b) Applying a degradation to audio data.

Figure 2: Demo degradation: specification and application.

Speed-up (W). The signal is resampled at a specified sampling rate but returned using the original sampling rate, which results in a speed-up (or slow-down). Timestamps are adjusted accordingly.

Wow Resampling (W). Similar to Speed-up, but the resampling frequency is time-dependent: it oscillates around the original sampling rate at a specified frequency and amplitude, imitating non-constant speed in record players or tape machines. Timestamps are non-linearly warped to correspond to the output audio.

These degradation units implement audio and ground-truth transformation and thus form the building blocks for higher-level degradations, the subject of the next subsection.

2.2 Degradations

A degradation is a chain of degradation units with fixed parameters. The purpose of daisy-chaining degradation units is to allow the creation of more complex degradations than would be possible by using the degradation units alone. A degradation is defined in a Matlab function that acts as a configuration file describing the order and parameters of all degradation units used. An example is given in Figure 2a: the demo degradation specifies a first degradation unit `degradationUnit_1` with the parameter `someParam1` set to 3, and a second degradation unit `degradationUnit_2` with the parameter `someParam2` set to 4. In this way, the audio processing chain can be precisely specified. Any degradation thus defined can be applied to audio using the Matlab function `applyDegradation` provided by the ADT, see Figure 2b. Based on the degradation name, given as the first argument, the function retrieves the degradation definition in the struct array `degr_cfg` and cascades the degradation units in the specified order. Optionally, timestamp data can be supplied, which is also sequentially transformed to match the output audio, which is useful to make ground-truth usable on the degraded audio (see Section 2.1). The ADT comes with a range of pre-defined degradations. For the purpose of this paper we focus on the subset of six *'Real World Degradations'* that cover a variety of scenarios (more precise definitions come with the ADT source code).

Live Recording. Based on two degradation units: 1. Apply Impulse Response, using an IR of a large room ('Great Hall', taken from [20] and included in the ADT), 2. Add Noise: adding light pink noise.

Radio Broadcast. Based on two degradation units: 1. Dynamic Range Compression at a medium level to emulate the high loudness characteristic of many radio stations, 2. Speed-up, by 2%, which is commonly applied to music in commercial radio stations to shorten the music to create more advertisement time.

	correct	incorrect	not identified
Original	**100**	0	0
Live	0	0	100
Radio	3	3	94
PhonePlay	0	1	99
PhoneRec	5	7	88
MP3	**100**	0	0
Vinyl	4	0	96

Table 2: EchoNest audio ID results for 100 test songs.

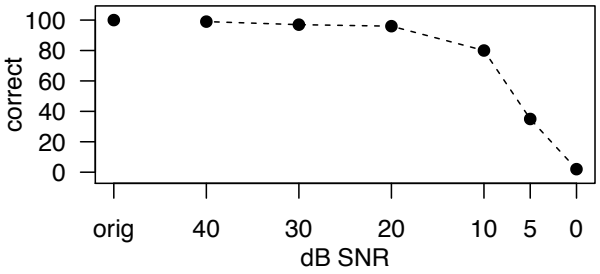


Figure 3: EchoNest audio ID results with pink noise.

Smartphone Playback. Based on two degradation units simulating a user playing back audio on a smartphone: 1. Apply Impulse Response, using the IR of a smartphone speaker ('Google Nexus One', Table 1a), which has a high-pass characteristic and a cutoff at ≈ 500Hz. 2. Add Noise, adding light pink noise.

Smartphone Recording. Based on four degradation units, simulating a user holding a phone in front of a speaker: 1. Apply Impulse Response, using the IR of a smartphone microphone ('Google Nexus One', Table 1a), 2. Dynamic Range Compression, to simulate the phone's auto-gain, 3. Clipping, 3% of samples, 4. Add Noise, adding medium pink noise.

Strong MP3 Compression. Based on one degradation unit: MP3 Compression at a constant bit rate of 64 kbps.

Vinyl. Based on four degradation units: 1. Apply Impulse Response, using a typical record player impulse response (Table 1a), 2. Add Sound, adding record player crackle (Table 1b), 3. Wow Resample, imitating wow-and-flutter, with the wow-frequency set to 33 rpm (speed of Long Play records), 4. Add Noise, adding light pink noise.

3. APPLICATIONS

In order to illustrate the insights that can be gained by using the ADT we evaluated several methods for standard music informatics tasks on suitable audio data. The results and brief discussions are given below.

3.1 Audio Identification Service

As a proof of concept we used the free web-based audio identification (audio ID) service from the Song API [4] pro-

[4] `http://developer.echonest.com/docs/v4/song.html`

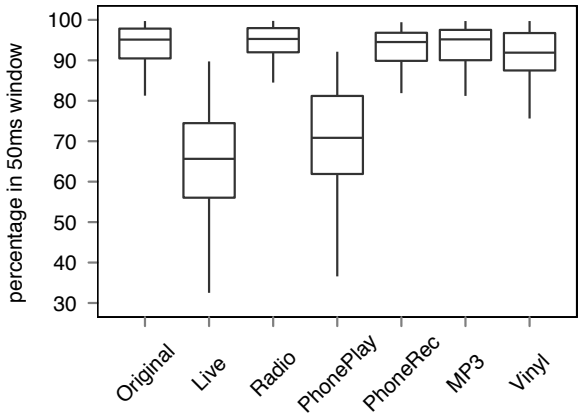


Figure 4: Score-to-Audio alignment accuracy under *Real World Degradations*. The boxes indicate the 1st, 2nd (median) and 3rd quartiles, the whiskers extend to 'the most extreme data point which is no more than 1.5 times the interquartile range' (R software [19]).

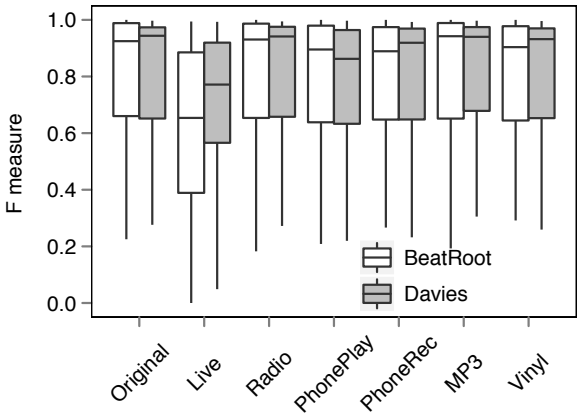


Figure 5: Comparison of beat-tracking performance under *Real World Degradations*.

vided by the company EchoNest. A user can compute an audio fingerprint using a dedicated program (ENMFP method) and query the EchoNest database for a corresponding identification number, artist and track name. If the system cannot identify a recording, that information is also returned.

We queried the database using 100 original rock and pop songs [5] taken from commercial CDs as well as degraded versions using the *Real World Degradations* defined in Section 2.2. The returned metadata was manually validated. The design goals for the EchoNest Audio ID service are clearly reflected in the results in Table 2: all 100 original tracks are correctly identified, as were all 100 files with strong MP3 compression. In contrast, all other degradations led to a clear failure with at most five recordings identified correctly, and low precision with up to seven recordings identified incorrectly. [6] However, as illustrated in Figure 3, an additional test showed that the system is reasonably robust against added pink noise up to a signal-to-noise ratio (SNR) of 10dB, for which 80 pieces were still correctly recognised (Figure 3). The poor real-world results are not surprising, since the service is supposed to *discriminate* between versions of the same song, not to detect similar songs. By contrast, the music informatics tasks below are concerned with the extraction of musical attributes that should persist even in degraded audio.

3.2 Score-to-Audio Alignment

Given a score and an audio recording for a piece of music, the aim of score-to-audio alignment is to find, for every position in the score, the corresponding position in the recording. In contrast to the audio ID task, which assumes a particular recording, the task is meant to work on any rendition or recording of the same musical work, and hence we expect a higher robustness against our real-world degradations. We use all 50 pieces from the Saarland Music Data [16], which contains audio recordings and corresponding MIDI files, both recorded using a Yamaha Disklavier. Our experimental setup is similar to the one described in [8]: the MIDI files are temporally distorted by randomly changing their tempo in 10-second intervals by up to 50%, faster or slower. We compute the alignment between the distorted MIDI files and the original audio recordings using the method described in [8], which combines chroma features with onset features. To measure the alignment accuracy, we computed for each recording the percentage of notes with an alignment error of less than 50ms for the onset position. The distribution of these values over all files is shown in the form of box-and-whisker plots in Figure 4.

This score-to-audio method is more robust than the EchoNest audio ID retrieval. The median over all files remains greater than 90% for all degradations, with two exceptions: *Live* and *Smartphone Playback*. Here, it is interesting to investigate the underlying reasons. The method employs a relatively simple onset detector to refine the alignment. The room IR used in the *Live* setting contains several early reflections, which generates several closely located 'copies' of onsets. These can easily be confused with the original onset. In the *Phone Playback* scenario, the significantly lower performance might be a result of applying the impulse response for the phone's speaker, which strongly attenuates all frequencies below 500 Hz including all fundamental frequencies up to B4. This leads to substantial differences between the observed audio and audio expected based on the score.

3.3 Beat-tracking

The aim of beat-tracking is to automatically find the timestamps of all beats in a piece of music. We compare two beat-trackers: *BeatRoot* [7] [5] and *Davies* [4] (QM Vamp Plugins implementation). *BeatRoot* first estimates note onset times, and forms a large number of tempo and beat hypotheses based on these onsets. A multiple-agent architecture is then used to determine the final beat estimates from the hypotheses. The *Davies* beat-tracker does not directly work on onsets but uses a continuous mid-level representation of onset salience, on which a comb filter is used to calculate the salience of different beat periods and beat alignments. Dynamic programming is used to retrieve

[5] A list of files is available on the project's website.

[6] In all cases, the songs are still easily recognisable to human listeners.

[7] http://www.eecs.qmul.ac.uk/~simond/beatroot, vers. 0.5.8

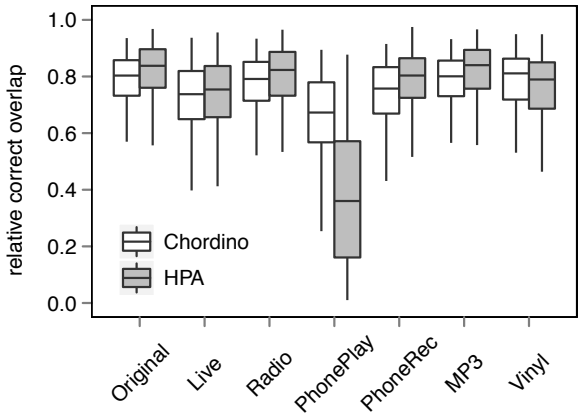


Figure 6: Chord detection performance: *Real World Degradations*

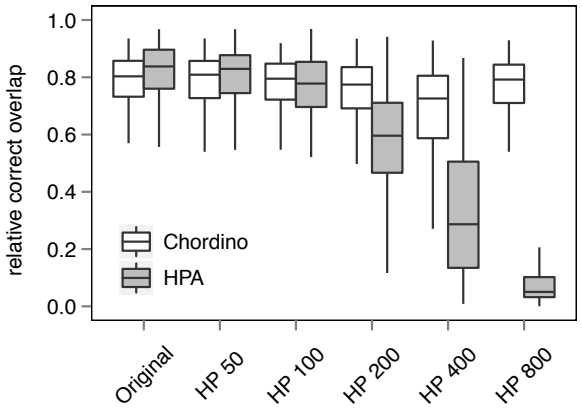


Figure 7: Chord detection performance: *High-pass filter* degradations.

the final beat estimate. Due to its dependence on onsets we would expect *BeatRoot* to be particularly susceptible to the *Live* setting (see Section 3.2).

We prepared 180 songs by the Beatles by degrading them using the *Real World Degradations*, resulting in 1260 wav files. Beats were extracted with both beat-trackers. We used a $\pm$70ms tolerance window to calculate the F measure for every song against human annotated ground-truth [14]. Figure 5 shows box-and-whisker plots of the F measure distributions, by degradation and beat-tracking method. For the original audio and most of the *Real World Degradations*, both beat-tracking methods show good performance, with median F measures always exceeding 0.85. With similar medians and inter-quartile ranges neither method has a clear advantage. The obvious exception is the *Live Recording* degradation, where the median F measure of both methods is substantially lower. *BeatRoot*: 0.65 (original: 0.92); *Davies*'s: 0.77 (original: 0.94). As explained in the case of score-to-audio alignment (Section 3.2), the likely cause are spurious onsets introduced by the impulse response; the *Davies* beat-tracker, which does not work on discrete onsets, is less affected.

3.4 Chord Detection

Chord detection is concerned with the transcription of the chord sequence in a piece of music. We test two different chord detection tools: *Chordino* [8] [15] and *HPA* [9] [17]. *Chordino* uses *NNLS Chroma* as a low-level feature, then matches manually defined chord templates to the chroma. Chords are modelled as hidden states in a hidden Markov model, and smoothing is achieved using Viterbi-decoding. *HPA* uses the same basic architecture, with some distinct differences: a beat-quantised, perceptually-inspired chroma representation (*HPA chroma*); a more complex probabilistic model that involves key and bass context; machine-learned chord profiles and transition parameters.

We continue to use the 180 songs by the Beatles from our beat-tracking experiment, as chord annotations are also available for them [10]. The chord detection outputs are evaluated by calculating the relative correct overlap for every song using a *MIREX-style* major/minor scheme [15].

Figure 6 shows box-and-whisker plots of the song-wise results by degradation and method. For the original audio and most degradations, *HPA* consistently outperforms *Chordino*, possibly due to its advanced exploitation of musical context and machine learning. Unlike the beat-trackers, both chord detection methods are relatively robust to the *Live Recording* degradation, with medians dropping less than 10 percentage points: *Chordino* 0.74 (original: 0.80), *HPA*: 0.75 (original: 0.84). Instead, they falter on the *Smartphone Playback* degradation: *Chordino* 0.67 (original: 0.80), *HPA*: 0.36 (original: 0.84). In order to understand whether this drop was caused by the degradation's high-pass characteristic (compare Section 3.2), we calculated five further degradations using the *High-pass Filter* degradation unit with the stop band edge parameter set to 50, 100, 200, 400 and 800 Hz, respectively. Figure 7 shows that the methods react very differently. The *Chordino* method remains relatively robust with the lowest median, 0.73, for a 400Hz stop band edge. The *HPA* method's advantage over *Chordino* is maintained for the 50Hz filter, but increasingly fails for higher cutoff frequencies with median values of 0.60 (200Hz), 0.29 (400Hz) and 0.05 (800Hz). In order to locate the reason for the strong drop-off, we studied the *HPA* chroma feature. Figure 8 shows an example of how the high-pass filter strongly affects the character of the feature, obfuscating the clear C major and A minor patterns.

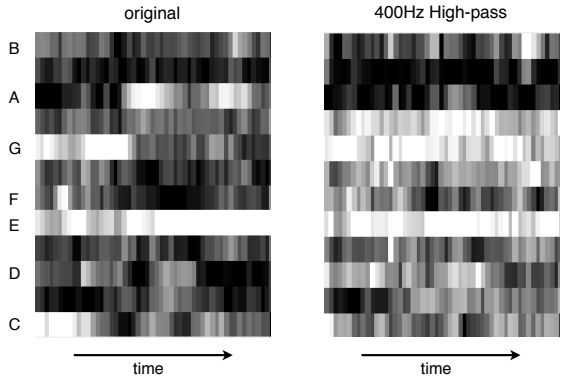


Figure 8: *HPA chroma* for the original and a high-pass filtered version of a snippet from 'Misery' by the Beatles.

[8] `http://isophonics.net/nnls-chroma`, Version 0.2.1

[9] `https://patterns.enm.bris.ac.uk/hpa-software-package`, Version 1.0

3.5 Summary of Experiments

As demonstrated by these examples, the ADT can be a valuable tool for investigating the robustness of music processing methods against a range of audio degradations. The *Real World Degradation* set in particular helps to provide an overview of the main problem areas for a method. For example, it has revealed that the otherwise excellent chord detection method *HPA* is not very robust to high-pass filtering (see Section 3.4). This gives an interesting example for how evaluating audio processing methods under multiple degradations can reveal their weaknesses and how it could drive the development of improved algorithms. While the ADT cannot replace 'looking at the data', experiments like the ones conducted in Section 3 can lead to interesting directions for further investigation.

4. CONCLUSIONS

In this paper, we have introduced and described the Audio Degradation Toolbox. The toolbox is written in Matlab and consists of 14 degradation units which can be chained into more complex degradations. Several predefined degradations can be used to simulate real-world degradations. With all required sounds, impulse responses and code modules included, the toolbox is entirely self-contained. Existing ground-truth can easily be transformed to account for time distortions introduced by the degradations.

We used the toolbox to analyse several methods in four music informatics tasks: audio ID, score-to-audio alignment, beat-tracking and chord detection. Our analysis offers insights into the robustness of different methods in various scenarios. In particular, we showed that a performance advantage of one method over another can be substantially reduced or even reversed by quasi-real-world degradations. We demonstrate how to track down particular weaknesses and argue that using the toolbox in this way makes it a powerful tool for developing more robust audio technology.

We are currently working on extending the toolbox with new sounds and degradation definitions. Furthermore, we plan to conduct user studies to investigate whether the degradations we propose have an influence on the way a human annotates an audio recording.

Acknowledgements: Matthias Mauch is funded by a Royal Academy of Engineering Research Fellowship, Sebastian Ewert is funded by EPSRC grant EP/J010375/1.

5. REFERENCES

[1] J. Burgoyne, J. Wild, and I. Fujinaga. An expert ground-truth set for audio chord recognition and music analysis. In *Proc. 12th Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2011.

[2] M. Casey, B. Fields, K. Jacobson, and M. Sandler. The effects of lossy audio encoding on genre classification tasks. In *Proc. 124th Convention of the Audio Engineering Society*, 2008.

[3] V. Chandrasekhar, M. Sharifi, and D. A. Ross. Survey and evaluation of audio fingerprinting schemes for mobile query-by-example applications. In *Proc. 12th Intl. Society for Music Information Retrieval Conf. (ISMIR)*, pages 801–806, 2011.

[4] M. Davies and M. Plumbley. Context-dependent beat tracking of musical audio. *IEEE Transactions On Audio Speech And Language Processing*, 15(3):1009–1020, 2007.

[5] S. Dixon. Evaluation of the audio beat tracking system BeatRoot. *Journal of New Music Research*, 36(1):39–50, 2007.

[6] C.-T. Do, D. Pastor, and A. Goalic. A novel framework for noise robust ASR using cochlear implant-like spectrally reduced speech. *Speech Communication*, 54(1):119–133, 2012.

[7] J. Downie, A. Ehmann, M. Bay, and M. Jones. The Music Information Retrieval Evaluation eXchange: Some observations and insights. In *Advances in Music IR*. 2010.

[8] S. Ewert, M. Müller, and P. Grosche. High resolution audio synchronization using chroma onset features. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, 2009.

[9] D. Giannoulis, E. Benetos, D. Stowell, and M. D. Plumbley. IEEE AASP CASA challenge: Public dataset for scene classification task, 2012.

[10] C. Harte, M. Sandler, S. A. Abdallah, and E. Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proc. 6th Intl. Conf. on Music Information Retrieval (ISMIR)*, pages 66–71, 2005.

[11] H.-G. Hirsch and D. Pearce. The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *ISCA Tutorial and Research Workshop (ITRW)*, 2000.

[12] K. Jacobson, M. Davies, and M. Sandler. The effects of lossy audio encoding on onset detection tasks. In *Proc. 125th Convention of the Audio Engineering Society*, 2008.

[13] P. v. Kranenburg, J. Garbers, A. Volk, F. Wiering, L. Grijp, and R. Veltkamp. Towards integration of music information retrieval and folk song research. Technical Report UU-CS-2007-016, Utrecht University, 2007.

[14] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. OMRAS2 metadata project 2009. In *Late-breaking session, 10th Intl. Conf. on Music Information Retrieval (ISMIR)*, 2009.

[15] M. Mauch and S. Dixon. Approximate note transcription for the improved identification of difficult chords. In *Proc. 11th Intl. Society for Music Information Retrieval Conf. (ISMIR)*, pages 135–140, 2010.

[16] M. Müller, V. Konz, W. Bogler, and V. Arifi-Müller. Saarland music data (SMD). In *Late-breaking session, Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2011.

[17] Y. Ni, M. McVicar, R. Santos-Rodriguez, and T. De Bie. An end-to-end machine learning system for harmonic analysis of music. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(5):1771–1783, 2012.

[18] P. Proutskova and M. Casey. You call that singing? Ensemble classification for multicultural collections of music recordings. *Proc. 10th Intl. Conf. on Music Information Retrieval (ISMIR)*, pages 759–764, 2009.

[19] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2013.

[20] R. Stewart and M. Sandler. Database of omnidirectional and b-format room impulse responses. In *Proc. IEEE Intl. Conf. on Acoustics Speech and Signal Processing (ICASSP)*, pages 165–168, 2010.

[21] T. Wilmering, G. Fazekas, and M. Sandler. The effects of reverberation on onset detection tasks. In *Proc. 128th Convention of the Audio Engineering Society*, 2010.

[22] U. Zölzer, editor. *DAFx—Digital Audio Effects*. John Wiley & Sons, 2002.

DO ONLINE SOCIAL TAGS PREDICT PERCEIVED OR INDUCED EMOTIONAL RESPONSES TO MUSIC?

Yading Song, Simon Dixon, Marcus Pearce
Centre for Digital Music
Queen Mary University of London
firstname.lastname@eecs.qmul.ac.uk

Andrea Halpern
Department of Psychology
Bucknell University
ahalpern@bucknell.edu

ABSTRACT

Music provides a powerful means of communication and self-expression. A wealth of research has been performed on the study of music and emotion, including emotion modelling and emotion classification. The emergence of online social tags (OST) has provided highly relevant information for the study of mood, as well as an important impetus for using discrete emotion terms in the study of continuous models of affect. Yet, the extent to which human annotation reveals either perceived emotion or induced emotion remains unknown. 80 musical excerpts were randomly selected from a collection of 2904 songs labelled with the Last.fm tags "happy", "sad", "angry" and "relax". Forty-seven participants provided emotion ratings on the two continuous dimensions of valence and arousal for both perceived and induced emotion. Analysis of variance did not reveal significant differences in ratings between perceived emotion and induced emotion. Moreover, the results indicated that, regardless of the discrete type of emotion experienced, listeners' ratings of perceived and induced emotion were highly positively correlated. Finally, the emotion tags "happy", "sad" and "angry" but not "relax" predicted the corresponding experimentally provided emotion categories.

1. INTRODUCTION

Music provides a powerful means of conveying and evoking feeling, and has attracted increasingly significant research interest in the past decades [5]. People report that their primary motivation for listening to music lies in its emotional effects [10]. A study of recreational activities (watching television, listening to music, reading books, and watching movies) indicated that people listen to music more often than any of the other activities [18]. The ability of identifying emotional content is established at very early age; a 5-year-old child can discriminate happiness and sadness by tempo and mode [2]. Although a wealth of research has been performed on the study of music and emotion [5], many problems remain unsolved.

In the study of emotion and music listening, one fundamental distinction is between induced emotion (also known as felt emotion), which is the emotion experienced by the listener, and perceived emotion (also known as expressed emotion), which is the emotion recognised in the music [8]. However, separating induced emotion from perceived emotion is not always straightforward. Previous studies have suggested that music induces emotions similar to the emotional quality perceived in the music. Generally, induced emotion is more subjective and perceived emotion tends to be more objective [7, 11] and it was found that the agreement of listener ratings of joy and sadness was higher than those for anger and fear [22].

To describe musical emotions, two well-known and dominant models have arisen: the discrete model (also known as the categorical approach) and the dimensional model (also known as the valence-arousal model). The discrete, or categorical, model describes all emotions as being derived from a limited number of universal and innate basic emotions such as anger, happiness, sadness and fear [6,17]. In contrast, the dimensional model considers all affective terms as arising from independent neurophysiological systems: one related to valence (a pleasure-displeasure continuum) and the other to arousal (activation-deactivation) [19]. On one hand, the dimensional model has been criticised for its lack of differentiation when it comes to emotions that are close neighbours in the valence-activation space, and for the limitation that participants can express their responses in terms of only two dimensions. On the other hand, the discrete model is criticised as being inadequate to describe the richness of emotional effects. Both theoretical frameworks, the categorical model and dimensional model, have however received empirical support [12, 24], and a comparison of the two models was presented by Eerola [4]. In recent years, a novel music-specific model derived from the Geneva Emotion Music Scale (GEMS), has been developed for specifically music-induced emotions, which consists of 9 emotional scales - wonder, transcendence, tenderness, nostalgia, peacefulness, power, joyful activation, tension and sadness [28].

Alongside the emergence of music discovery websites such as Last.FM [1] in the past decade, social tags have received increasing interest for the study of music and emotion [3, 15, 25]. Social tags are words or groups of words

[1] http://www.last.fm

supplied by a community of internet users. They are more and more commonly used to aid navigation through large media collections [27], allowing users to get a sense of what qualities characterise a song at a glance [9]. Compared with traditional human annotation by experts, semantic tags provide large-scale, cost-efficient, rich and easily accessible source of metadata [23]. In addition, the information they provide is highly relevant to music information retrieval, including genre, mood and instrument, which account for 70% of the tags [13].

Though the use of social tags is a powerful tool which can assist searching and the exploration of music [14], several problems with tags have been identified, such as the "cold start" problem (new or unknown music has no tags), noise, malicious tagging, and bias towards popular artists or genres [13]. There are a number of incentives and motivations for tagging, such as to aid memory, provide context for task organisation, social signalling, social contribution, play and competition, and opinion expression [1]. However, we know very little about the criteria on which tagging is based.

To our knowledge, the two facets of emotion communication (perceived emotion and induced emotion) in music have rarely been studied in combination with semantic tags. The purpose of this paper is to explore the association between human-annotated tags and emotional judgements in perceived emotion and induced emotion based on the dimensional model. This study also helps the mapping and modelling of mood tags on the valence-arousal space. In this paper, the following research questions are examined: (1) How do induced emotion and perceived emotion differ from each other in the ratings of valence and arousal for a 2-dimensional model of emotion? (2) How well do semantic emotional tags reflect listeners' perceived emotion and induced emotion? (3) To what degree can the emotional tags be used to select stimuli for the study of music and emotion?

2. METHOD

2.1 Participants

Forty-seven English-speaking participants (male: 20; female: 27) took part in this study. They were recruited through various email lists (e.g. school lists, professional lists, and social media), and had ages ranging from 15 to 54 years (age <18: 1; age 18-24: 22; age 25-34: 21; age 35-44: 1; age: 45-54: 2) with various educational (e.g. undergraduate, postgraduate), cultural (e.g. British, American, French, Chinese, Canadian, Italian, Greek, Sri Lankan) and musical training backgrounds (musician and non-musician).

To assess the participants' musical expertise, the Goldsmiths Musical Sophistication Index questionnaire (GOLD-MSI) was given [16] (see section 2.3). Participants' *musical training* (life history of formal musical training) was calculated using a provided template [2] giving a scale from 9 to 63 (no formal training to formal training). A summary of the responses can be found in Table 1.

[2] http://www.gold.ac.uk/music-mind-brain/gold-msi/

Skill	Min	Max	Median	SD
Musical Training	17	42	29	6.2915

Table 1. Summary of musical training

2.2 Stimuli

The stimuli were selected from a collection of 2904 excerpts retrieved from Last.FM and 7Digital [3] which have been used previously in music and emotion studies [20,21].

Each excerpt had been tagged on Last.FM with one of the four words "*happy*", "*sad*", "*angry*" and "*relax*". We randomly chose a total of 80 excerpts from these four categories (n=20 from each category). The musical excerpts ranged from recent releases back to 1960s, and covered a range of Western popular music styles such as pop, rock, country, metal and instrumental. Each excerpt was either 30 seconds or 60 seconds long (as provided by 7Digital), and it was played from a standard mp3 format file (bitrate: 128 kbps or 64 kbps; sample rate: 22050 kHz or 44100 kHz). This 80-excerpt dataset will be made available [4], to enable further studies with this data and comparisons with the current work.

In order to minimise the effect of song sequence and rating conditions (perceived and induced emotion), four different list conditions were constructed. The order of presentation of the two rating conditions and two song blocks (m=40, 10 for each emotion category) was counterbalanced across subjects. The songs in each block were randomly distributed across participants [26]. See Table 2 for the group allocation. To remind the subjects of two different rating conditions, the questions were highlighted in different colours (blue and red) and they were also counterbalanced across groups.

Group	Block 1	Block 2
Group 1	Induced emotion	Perceived emotion
Group 2	Perceived emotion	Induced emotion
	Block 2	**Block 1**
Group 3	Induced emotion	Perceived emotion
Group 4	Perceived emotion	Induced emotion

Table 2. Group allocation among participants

2.3 Procedure

The study was approved by the Research Ethics Committee (REF: QMREC1019). The listening test was conducted online [5]; participants only required internet access and a speaker or headphones for this experiment. First, the participants were asked to read the instruction page:

1. Listen to the songs (they will last either 30 or 60 seconds)

[3] http://www.7digital.com/

[4] https://code.soundsoftware.ac.uk/projects/emotion-recognition

[5] http://isophonics.net/dimensional/test/

2. After listening, please rate each piece on two dimensions: Valence (happy-sad continuum) and Arousal (excited-relaxed continuum)

3. For each track, you may click the "stop" button of the audio player if required

4. Be careful, do not press too quickly, since you can only listen to each song once

5. Please answer all the questions; the test will take about 40 mins to complete

The participants filled in a demographic form including name, age, gender, "type of music they are most familiar with", nationality, and "music culture they grew up with" as well as a selected Goldsmiths Musical Sophistication Index (GOLD-MSI) questionnaire (9 questions) to measure participants' level of musical training. They responded to each excerpt (n=10 per page) and rated them on two dimensions, valence (sad to happy) and arousal (relaxed to excited). According to the session they chose, a pop-up window would appear reminding them, based on which condition they needed to answer, "*How would you describe the emotional content of the music itself? (expressed emotion)*" , and "*What emotion do you feel in response to the music? (felt emotion)*". The whole experiment lasted about 40 minutes without any breaks. However, the participants were able to stop whenever they wanted.

3. RESULTS

The data analysis was conducted using the Matlab 2012 Statistics Toolbox. The results were aggregated across people for song level analysis or aggregated across item for individual level analysis.

3.1 Song level analysis

3.1.1 Comparison of valence and arousal ratings for perceived and induced emotion

To understand the effects of rating conditions (perceived emotion and induced emotion) and emotions (happy, sad, relax [6] and angry) on the ratings of valence and arousal, a two-way analysis of variance (ANOVA) was conducted. No significant difference was found for the ratings for the two conditions and the interaction of emotion and condition. On the other hand, the emotion tag had a significant effect on the ratings for valence and arousal (see Table 3).

3.1.2 Correlation between the ratings for perceived emotion and induced emotion

Section 3.1.1 showed that there was no significant difference between ratings for perceived and induced emotion. Correlation analyses were performed to study the relationship between valence (respectively arousal) ratings for perceived and induced emotion. Regardless of the discrete emotion tag, the listeners' valence and arousal ratings were highly positively correlated between perceived and induced emotion cases (valence: $r = 0.9357$, $p < 0.0001$; arousal: $r = 0.9590$, $p < 0.0001$). The correlation of valence is shown in Figure 1.

[6] The term "relax" was used in the data collection, which represented the emotion "relaxed"

Source	SS	df	MS	F	Prob >F
Valence					
Condition	1.93	1	1.93	1.14	0.29
Emotion Tag	163.26	3	54.42	31.97	**0**
Interaction	0.27	3	0.09	0.05	0.98
Arousal					
Condition	0	1	0.0002	0	0.99
Emotion Tag	231.55	3	77.18	25.73	**0**
Interaction	0.99	3	0.33	0.11	0.95

Note: SS - the sums of squares, df - degrees-of-freedom MS - mean squares (SS/df), F - F statistics

Table 3. Two-way analysis of variance on the ratings of valence and arousal

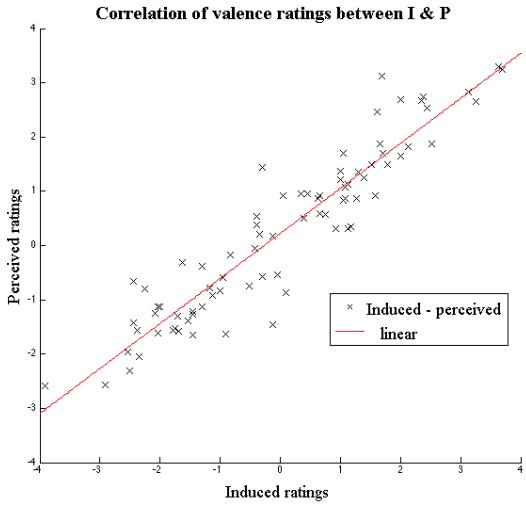


Figure 1. Correlation between the valence ratings for induced and perceived emotion

3.1.3 Correspondence between ratings and tags

The four emotion tags were chosen such that each occupies a unique quadrant of the valence-arousal plane, as shown in Figure 2. Considering that these four basic emotions are widely accepted across different cultures, we are able to assess the agreement between tags and participant ratings according to the extent that participants' ratings correspond with the quadrant belonging to the song's tag.

For each song, the average of participants' valence and arousal ratings were calculated for both perceived and induced emotion, to give a centroid for each song. The quadrant of this song centroid was then compared with the expected quadrant based on the emotion tag associated with the song. The fraction of songs for which the centroid quadrant corresponded with that of the tag is shown in Table 4. In addition, the standard deviations (SD) of the valence and arousal ratings for songs in each emotion category were calculated.

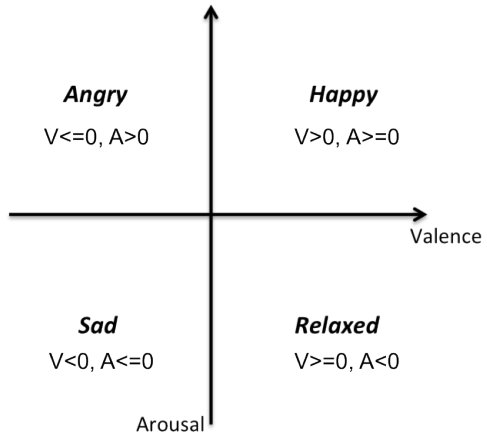


Figure 2. Valence-Arousal model showing the quadrants of the four emotion tags used in this experiment.

The results are shown with the highest values shown in bold in Table 4. Apart from the excerpts tagged with "relax", more than 55% of the average valence and arousal ratings lie in the song's corresponding tag quadrant. Fewer than 20% of mean ratings for songs tagged "relax" were located in the correct quadrant. Moreover, the high standard deviation of valence-arousal ratings for both perceived and induced emotion were found, indicating that the ratings of "relax" excerpts were not consistent across songs. For all emotion tag categories, and for both perceived and induced emotion, it was found that valence ratings were more consistent than arousal ratings.

	Happy	**Sad**	**Relax**	**Angry**
Induced emotion				
Rating=Tag	0.70	0.70	0.20	0.60
Valence SD	1.05	1.41	**1.81**	1.20
Arousal SD	1.56	1.45	**2.46**	1.82
Perceived emotion				
Rating=Tag	0.80	0.65	0.15	0.55
Valence SD	1.10	1.25	**1.47**	0.93
Arousal SD	1.34	1.32	**2.04**	1.56

Table 4. Agreement of valence-arousal ratings with tag quadrants, and spread of per-song ratings.

Figure 3 presents the average participant ratings of each song using the 2-dimensional model of emotion. The figure shows a high level of agreement between ratings for perceived and induced emotion (note that each participant rated each song for only one of perceived and induced emotion), compared to the spread of songs corresponding to an emotion tag.

3.1.4 Spread of participants' ratings

For each song, the spread of participant ratings was computed as the mean distance between points (centroid of the averaged ratings across participants and the ratings of each participant) in the valence-arousal plane. These values were then averaged across songs labelled with each emotion tag, as shown in Table 5. The spread of ratings for perceived and induced emotion were similar; likewise the spreads for each emotion category were similar, with a slightly higher spread found for "sad" songs.

	Happy	**Sad**	**Relax**	**Angry**
Induced emotion	2.59	**2.75**	2.45	2.65
Perceived Emotion	2.56	**2.76**	2.54	2.52

Table 5. The spread of the participant ratings across songs in each emotion category.

3.2 Individual level analysis

3.2.1 Individual agreement of participant ratings with the emotion tags

The analysis and results in section 3.1.3 were based on ratings for each excerpt that were averaged across participants. To analyse the relationship between individual ratings and emotion tags, we compute the fraction of ratings that are in the same quadrant as the emotion tag for the song, and compare this with the baseline of 25% for random choice of quadrants. The results are shown in Table 6. Chi-square tests were used to test whether the agreement with the emotion tag was significantly above chance level.

	Happy	**Sad**	**Relax**	**Angry**
Induced emotion	***0.56	*0.47	0.25	**0.49
Perceived emotion	**0.57	0.42	0.25	**0.48

Table 6. Agreement of participant ratings with the quadrant of the emotion tag for each category. Values above chance level according to χ^2 tests are shown for the following significance levels: $*p < 0.05$; $**p < 0.01$; $***p < 0.001$.

It was found that the songs labelled with "happy" had the highest agreement 55% (p <0.001). Significant results were also found for tags "sad" (p <0.05, induced emotion) and "angry" (p <0.01). However, the agreement of participant ratings and the expected quadrant for songs labelled with "relax" was at the level of chance.

3.2.2 Agreement among participants

Since the tags associated with a song are not an absolute ground truth, but are also generated by users, under unknown conditions, we also look at the agreement of rating quadrants among the participants. The level of participant agreement is defined as the fraction of participants whose ratings are in the quadrant with the highest number of participant ratings. This value has as a lower bound the agreement with the tag quadrant, but can be higher if a greater number of participants agree on a quadrant other than that of the tag. The agreement of the individual dimensions of valence and arousal was also computed. The Wilcoxon

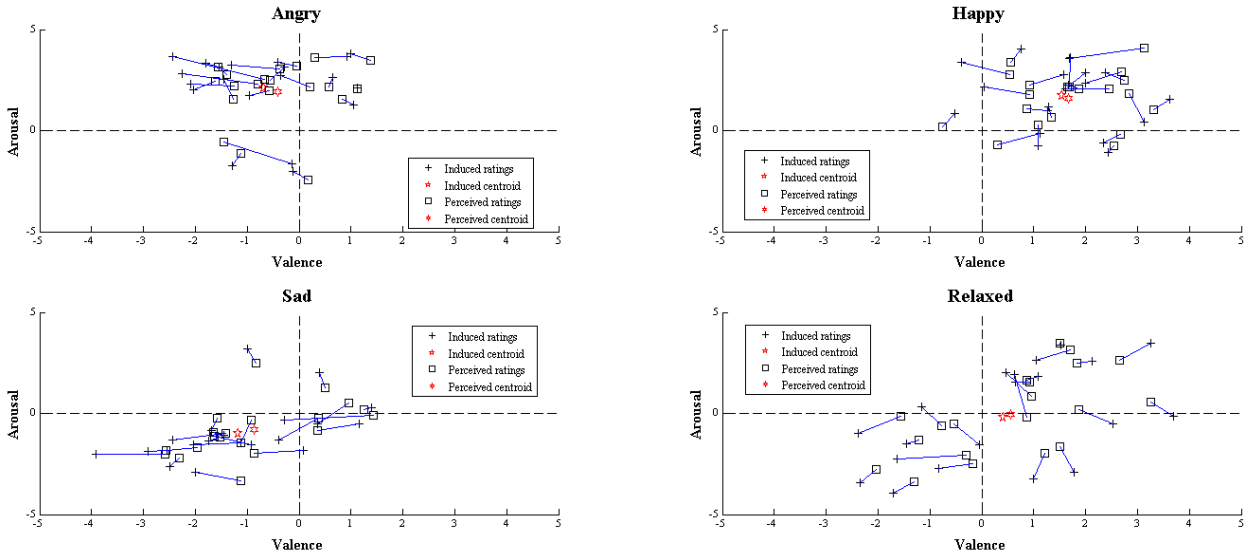


Figure 3. Map showing the average valence-arousal ratings for each excerpt. Each pane shows the ratings and centroid for one emotion tag. For each song, the points representing average ratings for perceived and induced emotion are connected by a line segment.

Signed Rank test was used to compare the difference of agreement for both perceived and induced emotion. Results are shown in Table 7. In comparison with Table 6, the levels of participant agreement are higher, suggesting that at least some of the tags do not correspond either with participants' perceived or induced emotion.

	Mean	SD	P-value
Induced emotion	0.60	0.14	0.0032
Perceived emotion	0.57	0.15	
Induced Valence	0.68	0.14	0.0211
Perceived Valence	0.66	0.15	
Induced Arousal	0.77	0.14	0.0001
Perceived Arousal	0.73	0.16	

Table 7. Participant agreement among themselves

3.2.3 The spread of song ratings for each emotion category

Similar to section 3.1.4, the spread of the song ratings for each person was computed for each emotion tag. Once again, the highest values were found for the tag "relax", because the responses for songs in this category vary more than for songs in other categories. This also helps to explain the low agreement with tags in section 6.

	Happy	Sad	Relax	Angry
Induced emotion	2.67	2.82	**3.49**	2.82
Perceived emotion	2.67	2.61	**3.15**	2.55

Table 8. The spread of the song ratings for each emotion category.

4. DISCUSSION

The main purpose of the paper was to investigate the associations between categorical emotion tags (happy, sad, relax and angry) and musical judgements of perceived emotion and induced emotion using a 2-dimensional (valence-arousal) model of emotion. First, two-way analysis of variance was used to test the effects of rating conditions and emotion categories on the ratings of valence and arousal. No significant difference was found between rating conditions, perceived and induced emotion, nor for the interaction of emotion categories and conditions. A correlation analysis on the ratings for induced and perceived emotion showed strong positive correlations between perceived and induced emotion, for ratings of both valence ($p<0.0001$ $r=0.94$) and arousal ($p<0.0001$ $r=0.96$). This suggests that listeners will typically feel the emotions expressed by the song, so if a song expresses happiness, it is likely that the listener will feel happy as well [21].

Second, we studied the reliability of the mean value among participants' ratings for predicting the emotion tag (happy, sad, relax and angry). The average valence and arousal ratings for both induced and perceived emotion were consistent with more than 55% of the happy, sad and angry tagged songs. However, the ratings of the songs labelled with "relax" tended to be less consistent, and did not agree with the quadrant of the tag. This can be seen in the mapping of songs to the valence-arousal plane (Figure 3), which shows the average ratings for songs labelled "relax" spread across happy, sad and relax quadrants. Interestingly, although 60% of the songs labelled with "sad" lie in correct quadrant, the spread of participants' ratings was higher than for the other emotion categories.

Third, we analysed the relationship between emotion tags and individual participant ratings. The results were

very similar to the average ratings in section 3.1.3; the songs labelled with "happy", "sad" and "angry" had ratings in the corresponding quadrants of the valence-arousal plane at a level that was significantly above chance. For songs tagged "relax", however, the agreement of ratings with the positive-valence, negative-arousal quadrant was at the level of chance for both perceived and induced emotion. Further, the spread of the ratings was also higher than for the other categories. Comparing these four tags, regardless of song or person, the excerpts tagged with "happy" are most likely to produce responses in the corresponding quadrant of the valence-arousal plane.

Finally, the agreement among participants were computed. Gabrielsson noted a higher agreement among listeners for perceived emotion, due to its objectivity, over induced emotion [8], and this was confirmed in a study using the categorical model and the same musical excerpts as the present study [21]. In this paper, we found that agreement among participants was significantly higher for induced emotion than for perceived emotion ($p<0.01$). However, we concede that the overall levels of agreement (60% for induced emotion and 57% for perceived emotion) are not high. As a partial explanation of discrepancies between emotion tags and user ratings, we observed lower levels of agreement among listeners for songs tagged "relax" than for the classes which had higher agreement between tags and participant ratings.

In future studies we plan to investigate the different responses in perceived and induced emotion, and compare the use of the categorical and dimensional models of emotion.

5. ACKNOWLEDGEMENTS

We acknowledge the support of the China Scholarship Council. We would like to thank the listeners and reviewers for their participation and comments.

6. REFERENCES

[1] M. Ames and M. Naaman. Why we tag: motivations for annotation in mobile and online media. In *SIGCHI conference on Human Factors in Computing Systems*, 2007.

[2] S. Dalla Bella, I. Peretz, L. Rousseau, N. Gosselin. A developmental study of the affective value of tempo and mode in music. *Cognition*, 80(3):B1–10, 2001.

[3] D. Eck, P. Lamere, T. Bertin-Mahieux, S. Green. Automatic generation of social tags for music recommendation. *Advances in Neural Information Processing Systems*, 2007.

[4] 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, 2010.

[5] T. Eerola and J.K. Vuoskoski. A review of music and emotion studies: approaches, emotion models, and stimuli. *Music Perception*, 30(3):307–340, 2013.

[6] P. Ekman. An argument for basic emotions. *Cognition and Emotion*, 6:169–200, 1992.

[7] P. Evans and E. Schubert. Relationships between expressed and felt emotions in music. *Musicae Scientiae*, 12(1):75–99, 2008.

[8] A. Gabrielsson. Emotion perceived and emotion felt: same or different? *Musicae Scientiae*, 123–147, 2002.

[9] M. Hoffman, D. Blei, et al. Easy as CBA: A simple probabilistic model for tagging music. In *10th International Society for Music Information Retrieval*, 2009.

[10] P.N. Juslin and P. Laukka. Expression, perception, and induction of musical emotions. *Journal of New Music Research*, 33(3):217–238, 2004.

[11] K. Kallinen and N. Ravaja. Emotion perceived and emotion felt: Same and different. *Musicae Scientiae*, 191–213, 2006.

[12] G. Kreutz, U. Ott, D. Teichmann, P. Osawa, D. Vaitl. Using music to induce emotions: Influences of musical preference and absorption. *Psychology of Music*, 36(1):101–126, 2007.

[13] P. Lamere. Social tagging and music information retrieval. *Journal of New Music Research*, 37(2):101–114, 2008.

[14] M. Levy and M. Sandler. A semantic space for music derived from social tags. In *8th International Society for Music Information Retrieval*, 2007.

[15] M. Levy and M. Sandler. Music information retrieval using social tags and audio. *IEEE Transactions on Multimedia*, 11(3):383–395, 2009.

[16] D. Müllensiefen, B. Gingras, L. Stewart. Goldsmiths Musical Sophistication Index (Gold-MSI) *Technical report*, 2012.

[17] J. Panksepp. Affective neuroscience: The foundation of human and animal emotions. *Oxford University Press*, 1998.

[18] 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):1236–1256, 2003.

[19] J.A. Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178, 1980.

[20] Y. Song, S. Dixon, M. Pearce. Evaluation of musical features for emotion classification. In *13th International Society for Music Information Retrieval*, 2012.

[21] Y. Song, S. Dixon, M. Pearce, G. Fazekas. Using tags to select stimuli in the study of music and emotion. In *3rd International Conference on Music & Emotion*, 2013.

[22] M. Terwogt and F. Van Grinsven. Musical expression of moodstates. *Psychology of Music*, 90–109, 1991.

[23] D. Turnbull, L. Barrington, G. Lanckriet. Five approaches to collecting tags for music. In *9th International Society for Music Information Retrieval*, 2008.

[24] S. Vieillard, I. Peretz, et al. Happy, sad, scary and peaceful musical excerpts for research on emotions. *Cognition & Emotion*, 22(4):720–752, 2008.

[25] D. Wang, T. Li, M. Ogihara. Tags better than audio features? the effect of joint use of tags and audio content features for artistic style clustering. In *11th International Society for Music Information Retrieval*, 2010.

[26] N. Welch and J.H. Krantz. The World-Wide Web as a medium for psychoacoustical demonstrations and experiments: Experience and results. *Behavior Research Methods, Instruments,& Computers*, 28(2):192–196, 1996.

[27] X. Wu, L. Zhang, Y. Yu. Exploring social annotations for the semantic web. In *15th International Conference on World Wide Web*, 2006.

[28] M. Zentner, D. Grandjean, et al. Emotions evoked by the sound of music: characterization, classification, and measurement. *Emotion (Washington, D.C.)*, 8(4):494–521, 2008.

A VIDEO COMPRESSION-BASED APPROACH TO MEASURE MUSIC STRUCTURE SIMILARITY

Diego F. Silva[1], **Hélène Papadopoulos**[2], **Gustavo E.A.P.A. Batista**[1] **and Daniel P.W. Ellis**[3]

[1]Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo
[2]Laboratoire des Signaux et Systèmes (L2S), CNRS UMR 8506, France.
[3]Department of Electrical Engineering - Columbia University

diegofsilva@icmc.usp.br, helene.papadopoulos@lss.supelec.fr,
gbatista@icmc.usp.br, dpwe@columbia.edu

ABSTRACT

The choice of the distance measure between time-series representations can be decisive to achieve good classification results in many content-based information retrieval applications. In the field of Music Information Retrieval, two-dimensional representations of the music signal are ubiquitous. Such representations are useful to display patterns of evidence that are not clearly revealed directly in the time domain. Among these representations, self-similarity matrices have become common representations for visualizing the time structure of an audio signal. In the context of organizing recordings, recent work has shown that, given a collection of recordings, it is possible to to group performances of the same musical work based on the pairwise similarity between structural representations of the audio signal. In this work, we introduce the use of the Campana-Keogh distance, a video compression-based measure, to compare musical items based on their structure. Through extensive experiments, we show that the use of this distance measure outperforms the results of previous work using similar approaches but other distance measures. Along with quantitative results, detailed examples are provided to to illustrate the benefits of using the newly proposed distance measure.

1. INTRODUCTION

Within the last few years, time-series methods and algorithms have attracted the interest of many research communities such as Data Mining and Information Retrieval. Indeed, processes of interest generally change over time, and the study of how these changes occur is a central issue to fully understand such processes.

The choice of the distance measure between time-series representations can be decisive to achieve good classification results in many content-based information retrieval applications. Two distance measures commonly used in time-series analysis are the Euclidean distance (ED) and the Dynamic Time Warping distance (DTW). The DTW can be understood as an extension of the ED, able to provide nonlinear time scaling invariance, popularly known as warping [3]. Those simple and well-known distances have been successfully applied to various kinds of problems and have proven to be very hard to beat [9].

Some time-series features are not evident in the time domain. There is a large number of research papers that propose methods to explore alternative representations of time-series, such as autocorrelation [1] and shapelets [20], in order to clarify specific features.

In the hot context of organization and retrieval of large collections of music, the notion of similarity between music recordings is of great importance for many applications such as music summarization [8] or cover song retrieval [11]. In general, the similarity between two recordings is measured by comparing their respective time-series.

Music audio signals are highly structured at different time-scales (bar, phrase-level, sections, etc.) and exhibit repetitive segments, e.g. the so-called ABA sonata form (exposition - development - recapitulation) in classical music. Since their introduction in the domain of audio in 1999 [12], self-similarity matrices (SSMs) have become common representations for visualizing the time structure of an audio signal in terms of self-similarity and repetitions. Such two-dimensional representations are obtained by computing the pairwise similarity of an audio feature sequence (such as mel-frequency cepstral coefficients [21] or chromas [2]), and allow putting in evidence patterns that are not clearly revealed directly in the time domain. For instance, repeated patterns in the audio will appear as diagonal stripes in the SSM. For more details and a review about music structure, we refer the reader to [24].

Among the various applications based on cross-retrieval tasks, we are interested in this work in the problem of, given a piece of music as a query, to automatically retrieve from a given collection all performances (various interpretations) of the query. We focus on classical music. Two interpretations of the same piece will have a similar musical content, but they may differ in many ways. Besides articulation, phrasing and ornamentation, the global tempo may

be different from one performance to another. Local tempo variations such as *ritardandi* may also exist. Moreover, other factors such as the recording conditions, the loudness or the instrumentation may result in huge dynamical and spectral deviations between the two interpretations.

Despite these variations, recent work has shown that it is possible to accurately measure the structural similarity between two recordings by computing the pairwise similarity between their self-similarity matrices, without extracting explicitly the underlying audio structure. In [16], this approach is applied to the cover song detection problem. In [22], this approach is used to build a retrieval system that searches a database for the musical piece that best matches a given *symbolic* structural query. Of particular interest to the present article are two closely related previous work that show that it is possible robustly group *audio* performances of the same musical work by using a distance that measures the pairwise similarity between their respective structural representations [4, 14].

In order to compare two-dimensional objects directly, it is appropriate to use a distance measure specific to this purpose. An example of that is the Campana-Keogh (CK-1) distance [7], which uses the video compression as the basis for estimating the dissimilarity between two images.

In this paper, we introduce the use of CK-1 to retrieve music recordings based on its structural similarity to an audio query. We show that the use of this distance measure outperforms the results of previous work using similar approaches but other measures.

2. STRUCTURAL SIMILARITY

As mentioned above, there are at least two studies that have successfully explored the possibility to retrieve music recordings according to their structural similarity to an audio query. In this work, we introduce the use of the CK-1 measure as an efficient distance that can be used to accomplish this task using self-similarity arrays.

This section describes each step of the proposed approach. We consider the following retrieval scenario. Given a query recording and a collection of music recordings that contains various performances of the same piece as the query, along with recordings of different compositions, we aim at retrieving all the performances of the query music piece. To this end, we define the training and retrieval steps by the Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 Training phase

Require: v = A collection of music recordings
  $S \leftarrow []$
  for $i = 1 \rightarrow length(v)$ **do**
    $s \leftarrow SSM(v[i])$
    $S \leftarrow concatenate(S, s)$
  end for

2.1 Feature Extraction

The first step consists in extracting a set of features that provides relevant information about the musical structure. Among the various clues that humans use to determine the structure of a music piece, the harmonic progression is very important [6]. Since their introduction in 1999, the *Pitch Class Profiles* [13] or *chroma* features [25] became common features for describing the harmonic content of music. The chroma features are, in general, 12-dimensional vectors that represent the spectral energy of the pitch classes of the chromatic scale. They have been successfully used for various content-based retrieval tasks, especially in previous work on music similarity [4, 14].

Algorithm 2 Retrieval phase

Require: q = A query recording
  S = The collection of SSM extracted in the training phase
  $s \leftarrow SSM(q)$
  $D \leftarrow []$
  for $i = 1 \rightarrow length(S)$ **do**
    $d \leftarrow d_{ck1}(s, S[i])$
    $D \leftarrow concatenate(D, \{d, i\})$
  end for
  $L \leftarrow sort\_by\_distance(D)$
  return L

Following these approaches, we extract a sequence of chroma features, as well as two variants of chroma-like features: the Chroma Energy Normalized Statistics (CENS) and the Chroma DCT-Reduced log Pitch (CRP). CENS features involve an additional temporal smoothing and downsampling step, leading to an increased robustness of the features to local tempo changes. CRP features boost the degree of timbre-invariance. We refer the reader to [23] for more details. For chromagram computation, we used the Matlab chroma toolbox [1] . All feature vectors are normalized to have unit norm.

2.2 Self-Similarity Matrix and Recurrence Plots

In order to analyze the music structure, we used a self-similarity matrix (SSM), defined by Equation 1.

$$S(i,j) = d(\vec{x}(i), \vec{x}(j)),\ i,j \in 1 \cdots N \tag{1}$$

where N is the length of the signal, $\vec{x}(i)$ and $\vec{x}(j)$ are the feature vectors at positions i and j of the signal, respectively, and $d(\cdot,\cdot)$ is a similarity measure. In this work, we used the cosine distance defined by Equation 2.

$$d(\vec{x}(i), \vec{x}(j)) = \frac{< \vec{x}(i), \vec{x}(j) >}{||\vec{x}(i)|| \cdot ||\vec{x}(j)||} \tag{2}$$

A well-known variation of the SSM is called Recurrence Plot (RP) [10], that introduces three parameters to the SSM: an embedding dimension m, a time delay τ and a closeness threshold ϵ. The general idea is that each vector $\vec{x}$ is augmented by m observations evenly spaced in τ units of time. Therefore, we end up with a matrix $X(i) \in \Re^{m \times k}$, composed by $\vec{x}(i), \vec{x}(i+\tau), \ldots, \vec{x}(i+(m-1)\tau)$. A RP can be formally defined according to Equation 3.

$$R_{i,j} = \Theta(\epsilon - d(X(i) - X(j))), X(i) \in \Re^{m \times k}, \quad i,j = 1..N-m \tag{3}$$

where ϵ is a closeness threshold and Θ is the Heaviside function (i.e. $\Theta(z) = 0$ if $z < 0$ and $\Theta(z) = 1$ otherwise). For a better understanding of these parameters and the use of closeness threshold, we recommend the reading of [15].

[1] http://www.mpi-inf.mpg.de/resources/MIR/chromatoolbox/

In short, RP is a thresholded version of the SSM, so that the SSM is transformed into a binary matrix. The general idea of using the parameter ϵ is to discard spurious differences among observations that may hinder the visualization of relevant events. However, we note that setting such a parameter is not an intuitive task, and one frequently has to rely on adhoc techniques to do so. As we will discuss in Section 5, our experiments show that the use of SSM and CK-1 distance can outperform the results obtained in the literature with RP and Normalized Compression Distance.

2.3 Compression Distances

The CK-1 distance [7] is based on the concept of Kolmogorov complexity, proposed to quantify the randomness of discrete sequences. The Kolmogorov complexity $K(x)$ of an object x is given by the size of the smallest program capable to output x on a universal computer, such as a Turing machine [19]. Intuitively, $K(x)$ is the minimal quantity of information required to generate a string x with a program. Similarly, the Kolmogorov conditional complexity $K(x|y)$ is the size of the smallest program to generate the sequence x, given a sequence y as auxiliary input.

These concepts are the basis to a metric called Information Distance, that is universal, in the sense that it subsumes other measures [5]. Although Information Distance has attractive theoretical properties, it is uncomputable in the general case. Therefore, several research papers have proposed approximations to this distance using compression algorithms [17, 18].

A widely used distance based on Kolmogorov complexity is the Normalized Compression Distance (NCD) [18], that uses standard compression algorithms to estimate the Kolmogorov complexity. Let $C(x)$ and $C(y)$ be the sizes of sequences x and y after they have been compressed, and $C(xy)$ the compression size of the concatenation of both sequences. The NCD is defined by Equation 4.

$$NCD(x,y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}} \quad (4)$$

NCD has shown to provide good similarity estimates for various applications in discrete domain, such as strings of DNA and natural language. NCD typically uses lossless compression algorithms, which are well-suited for discrete data. In short, lossless compression relies in finding *exact* recurring sequences in data. However, images are composed by real-valued pixels and exact repeating patterns are rare. Another relevant issue is that generally the used compression algorithms work over data sequences and are not able to make use of spatial patterns present in images.

To overcome these limitations, the CK-1 distance makes use of video compression algorithms. More specifically, it uses the MPEG-1 compressor. Such an algorithm explores recurring patterns within a frame and/or between consecutive frames to compress the video. When two consecutive frames are composed of similar images, the inter frame compression step should be able to exploit this structure to produce a smaller file size, which is therefore interpreted as significant similarity. As digital video is an important commercial application, many efforts have been made to achieve high compression rates in video encoding, making it a good approximation of the Kolmogorov conditional complexity.

In order to calculate a distance between two images, x and y, we can create a fictional video with one frame for each image. Thus, CK-1 is defined by Equation 5.

$$d_{ck1}(x,y) = \frac{C(x|y) + C(y|x)}{C(x|x) + C(y|y)} - 1 \quad (5)$$

where $C(x|y)$ is the size of a two-frame MPEG-1 video composed by images y and x, in this order.

3. RELATED WORK

In [4], the author proposes an approach to retrieve music using an *intermediate representation* of musical structure, without any explicit determination of such structure. The approach uses RP as representation of music structure and NCD as similarity measure between two plots. The paper performs a large-scale evaluation of the proposed approach, including an exploration carried out to find the best parameter configuration, a comparison of front-end features, embedding and recurrence analysis strategies. Supported by such a large experimental evaluation, the author provides evidence of the effectiveness of the approach.

Subsequently, a similar method was proposed in [14], using the Euclidean distance. Due to the simplicity of such distance, the proposed method resulted in a more efficient approach. However, temporal variations may cause recurrence shifts and the Euclidean distance is very sensitive to patterns translations. To overcome this limitation, the authors used a technique to smooth the resulting plot. The results, obtained on a dataset slightly different from [4] (with a smaller number of recordings) also proved effective.

4. EXPERIMENTAL SETUP

We conducted a broad experimental evaluation to assess the effectiveness of the proposed method for music retrieval. In order to evaluate and fairly compare our method to previous work, we used the same data sets, and the same feature extraction parameter variation used in [4]. In addition, we conducted experiments with a technique similar to that presented in [14], applying the binarization and blur methods over the SSM.

4.1 Data Sets

In our evaluation phase, we used two datasets of the classical repertoire, kindly provided by the author of [4]:

- The first dataset, referred in this paper as *123tracks*, is composed by 123 recordings of 19 different works from the classical and romantic period. Among these tracks, 56 are played on piano and the remaining 67 are symphonic movements. In total there are 59 different conductors in this dataset;

- The second dataset, referred in this paper as *Mazurkas*, consists in 2919 [2] recordings of 49 Chopin's mazurkas for piano. These were recorded by 135 different pianists.

4.2 Parameter Configuration

The CK-1 distance measure is parameter-free, as well as the procedure to create the SSMs. However, CK-1 has a small caveat, it requires that all images must have the same dimensions. Since the dimensions of the SSM are proportional to the recordings durations, which are variable, it is necessary to resize the feature sequences to a fixed dimension d before extracting the SSM. In our experiments, this is achieved by a resampling procedure, resulting in five different feature dimensions, $d \in \{300, 500, 700, 900, 1100\}$.

For each chroma-based feature, Chroma, CENS and CRP, we used seven different analysis window lengths, resulting in seven feature rates, $f \in \{0.333, 0.5, 1, 1.25, 2.5, 5, 10\}$ features/second.

To facilitate the reading and writing of parameter configurations, we adopted the notation $Feat_{f=F;d=D}$, where $Feat \in \{Chroma, CENS, CRP\}$ and F and D are the values of feature frequency (f) and the feature vector dimension (d), respectively.

4.3 Evaluation

All the experiments in this work were evaluated using mean average precision (MAP). Consider a collection C consisting of M items, and a subset $Q \subset C$ containing n different performances of the same music piece. Given a query $q_i \in Q$, we build a ranked list by arranging the results in ascending order according to the calculated distance between all the pieces in the collection and the query q_i. The Average Precision (AP) is defined by Equation 6.

$$AP(q_i) = \frac{1}{n} \sum_{r=1}^{M} P(r)\Omega(r) \quad (6)$$

where r is the rank in the ordered list, and

$$P(r) = \frac{1}{r} \sum_{i=1}^{r} \Omega(i) \quad (7)$$

and $\Omega(r)$ is 0 if the work r is relevant and 1 otherwise. Finally, the MAP is defined by the mean of all AP values.

5. RESULTS

5.1 Results on the *123tracks* dataset

We start presenting the results obtained with SSM and CK-1 distance on the *123tracks* dataset. Tables 1, 2 and 3 report MAP results obtained by the proposed method, varying parameters to calculate the SSMs using Chroma, CENS and CRP, respectively. Non-parametric paired Friedman and Dunnet post-hoc tests were applied to compare the statistical difference between the results. Yellow-shaded cells are statistically equivalent to the best result in each table.

Table 1. MAP results obtained with Chroma features

Feature rate (f)	Feature dimension (d) 300	500	700	900	1100
10	0.914	0.915	0.904	0.900	0.897
5	0.918	0.910	0.902	0.894	0.896
2.5	**0.930**	0.922	0.898	0.890	0.885
1.25	0.927	0.923	0.903	0.893	0.887
1	0.929	0.928	0.913	0.903	0.889
0.5	0.930	0.928	0.917	0.907	0.890
0.33	0.930	0.927	0.918	0.907	0.888

Table 2. MAP results obtained with CENS features

Feature rate (f)	Feature dimension (d) 300	500	700	900	1100
10	0.926	0.920	0.914	0.911	0.908
5	0.943	0.929	0.923	0.917	0.915
2.5	0.946	0.945	0.941	0.935	0.930
1.25	0.943	0.943	0.937	0.933	0.930
1	0.944	0.944	0.941	0.938	0.936
0.5	**0.946**	0.944	0.940	0.940	0.942
0.33	0.942	0.940	0.934	0.932	0.932

Table 3. MAP results obtained with CRP features

Feature rate (f)	Feature dimension (d) 300	500	700	900	1100
10	0.930	0.936	0.937	0.932	0.935
5	0.933	0.933	0.937	0.934	0.935
2.5	0.933	0.940	0.936	0.935	0.934
1.25	0.935	**0.941**	0.935	0.936	0.936
1	0.933	0.936	0.932	0.934	0.937
0.5	0.930	0.937	0.929	0.933	0.932
0.33	0.929	0.937	0.929	0,932	0.931

Since the parameter settings used in these experiments are the same as those used in [4], the results can be directly compared. The results obtained with the proposed method outperformed the results of [4] for all parameter settings. For instance, the best MAP result obtained with CK-1 and SSM is 0.946 ($CENS_{f=0.5;d=300}$). The competing method best result is 0.863 ($CRP_{f=10;d=700}$), a significant difference of almost 9%. We must note that CK-1 and SSM have no internal parameters to be searched for. Meanwhile, the recurrence plots used in [4] have three parameters that need to be set. According to the author, after a computational intensive search in the parameters space varying the closeness threshold, the time delay and the embedding dimension, the best result obtained was 0.921. We note that such result is still outperformed by our parameter-free method.

Although the performance difference is too small in order to claim a statistically significant difference with high confidence, there are other aspects that should be considered. The lack of internal parameters makes our method much simpler to use and having its results replicated by the research community. Another relevant issue is that our method is very robust to changes of *external* parameters. A poor parameter choice in the feature extraction

[2] This is the number of recordings presented in the official description of the dataset. However, there are pieces that are not available, and we found a total of 2914 recordings. This condition also applies to previous work, therefore does not affect the comparison of results.

step does not affect the performance significantly. For instance, the worst result obtained by the proposed method is 0.885, using $Chroma_{f=2.5;d=1100}$, while in [4], the worst result is 0.225. Such invariability to the parameter setting is verified by the lack of statistically significant differences among our best result and our remaining results when we vary the external parameters. We also obtained an excellent mean performance of 0.926 over all parameter values.

5.2 Further Experiments

We mentioned the fact that the NCD may not be appropriate to compare real value matrices. To prove this fact, we applied the NCD on the SSMs obtained with the best parameter configuration for Chroma, CENS and CRP, using the CK-1 distance. We also applied the NCD on the matrices obtained by setting $CRP_{f=10;d=700}$, the configuration that achieved the best result in [4]. The results of this experiment are shown in Table 4, as well as the results obtained using ED in the same configurations.

Table 4. Results obtained by applying NCD and ED on the SSM in some configurations.

Configuration	NCD	ED
$Chroma_{f=2.5;d=300}$	0.322	0.279
$CENS_{f=0.5;d=300}$	0.382	0.313
$CRP_{f=1.25;d=500}$	0.276	0.257
$CRP_{f=10;d=700}$	0.271	0.262

With this simple experiment, it is possible to note that ED and NCD are not appropriate to compare SSM directly. However, in [14], it was shown that the ED can be effectively used to retrieve songs by preprocessing the SSM. We take advantage of such idea to evaluate the use of CK-1 distance in this context.

As we did not have access to the code used in [14], we just simulated similar experiments by applying threshold and blur procedures. In other words, we did not apply path enhancement technique before applying the threshold. Our goal is not to directly compare the results, because we do not even have the same dataset. However, we can compare the use of ED and CK-1 when some preprocessing operations are applied on the SSMs.

In the binarization step (application of a threshold), we used the strategy to consider that $k\%$ of the closest points in the SSM represent recurrence. Thus, these points are transformed into black (0) pixels in the resulting RP. The remaining become white (1). To evaluate different scenarios, we used three different values for the threshold: $k \in \{10, 25, 50\}$. Furthermore, we used a two-dimensional circular averaging filter (Pillbox) to blur the image, using five different filter sizes: $l \in \{1, 5, 10, 20, 30\}$. Figure 1 shows different representations of the same recording of the Chopin's Prelude Op.28 No.4. Plot (a) represents the SSM, plot (b) represents the RP (using 25% of the points) and plots $(c$ & $d)$ the result of applying a blur filter on the RP with four different sizes.

For the sake of space, we do not present all the results of our experiments. Briefly, the best results obtained by the use of distance ED in this context were better than those

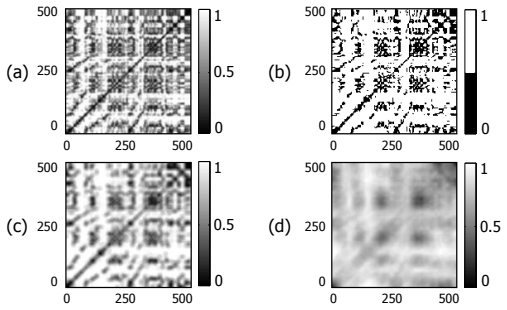


Figure 1. Four different representations of the same recording: (a) self-similarity matrix; (b) recurrence plot; (c) RP after application of a blur filter with size $l = 30$.

obtained in [4], which used the RP-NCD, in most scenarios. However, the results obtained by CK-1 distance in the same context were better than ED in all scenarios. The best results obtained for each distance and each feature are presented in Table 5.

Table 5. Results obtained after applying a threshold and a blurring effect in the recurrence plots. The value k is the percentage of black points and l is the size of the blur filter. The symbol $*$ indicates where CK-1 is statistically outperformed ED in the same parameter configuration.

	k (%)	l	Distance	MAP
$Chroma_{f=2.5;d=300}$	50	1	CK-1	0.941$*$
			ED	0.816
	10	20	CK-1	0.905
			ED	0.872
$CENS_{f=0.5;d=300}$	25	5	CK-1	0.924$*$
			ED	0.829
	25	1	CK-1	0.919
			ED	0.910
$CRP_{f=1.25;d=500}$	25	30	CK-1	0.958$*$
			ED	0.893
	10	30	CK-1	0.953
			ED	0.941

5.3 Results on *Mazurkas* Dataset

We performed experiments on the *Mazurkas* dataset to validate the results obtained in the previous experiments. We first chose the parameter configuration $CENS_{f=0.5;d=300}$ since it obtained the best classification performance in the *123tracks* dataset. However, our method achieved a MAP of 0.611, which we considered unsatisfactory. A more detailed analysis of the SSMs showed that in many cases the matrices were not able to clearly represent the recording structure. This was due to the fact that the cosine distance, used to extract matrices, resulted in short distances in many cases. Thus, the figure generated by such distances contains very dark colors when applied to a color scale between 0 and 1.

After analyzing the recordings, we can conclude that the distances with small values can be directly related to the frequency in which the features were extracted. A low feature rate corresponds to a large analysis window, resulting in mixing several structurally distinct segments of music. Since many pieces in the dataset have a short duration and numerous structural variations of short duration,

their structure can only be accurately analyzed with higher frequencies of feature extraction. To prove this fact, we performed the same experiment using $CENS_{f=1;d=300}$, achieving $MAP = 0.760$, without the need of normalizing the SSM, similar to the best result achieved in [4], $MAP = 0.767$. The Figure 2 shows an example of the difference between different feature extraction rates.

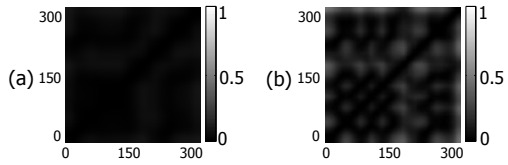


Figure 2. Self-similarity matrices of the same piece, but different feature rates: (a) $0.5f/s$; (b) $1f/s$;

Finally, we used the same dataset to test the configuration $CRP_{f=1.25;d=500}$ after the application of a threshold ($k = 25$) and a blur filter ($l = 30$). This test was performed since this configuration had the best result on the *123tracks* dataset. The effectiveness of the proposed method on this setting was proved in this experiment. When analyzing the structural similarity in this configuration using ED, we reach the result of $MAP = 0.652$. However, when we used the CK-1 measure, we obtained $MAP = 0.795$. This result is statistically superior to the results obtained by competing methods in the same dataset.

6. CONCLUSION

This paper proposes a simple and parameter-free approach to recover music by its structural similarity. The only parameters involved in our method are those required in the feature extraction step. We also show that our method is robust to a poor choice of these parameters, provided the parameter choices achieve meaningful SSM.

Through a wide experimental evaluation, we show that our method is superior to the techniques presented in previous work. Furthermore, CK-1 outperforms other distances regardless of pre-processing steps are performed in the signal or structural representation.

We believe that the contribution of this paper is not limited to the presentation of a new method for retrieving music by structural similarity. We hope this work will encourage the scientific community to analyze signals from various domains using visual representations and image-friendly distance measures.

7. ACKNOWLEDGMENTS

This work was supported by grants #2011/04054-2, #2012/-18985-0 and #2012/50714-7, São Paulo Research Foundation (FAPESP), and grant IIS-1117015, National Science Foundation (NSF).

8. REFERENCES

[1] A. Bagnall, L.M. Davis, J. Hills, and J. Lines. Transformation based ensembles for time series classification. In *ICDM*, 2012.

[2] M.A. Bartsch and G.H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Trans. Multimedia*, 7(1):96–104, 2005.

[3] G.E.A.P.A. Batista, X. Wang, and E.J. Keogh. A complexity-invariant distance measure for time series. In *SDM*, 2011.

[4] J.P. Bello. Measuring structural similarity in music. *IEEE Trans. Sp. Aud. Proc.*, 19(7):2013–2025, 2011.

[5] C.H. Bennett, P. Gacs, Ming L., P.M.B. Vitanyi, and W.H. Zurek. Information distance. *IEEE Trans. Inf. Theory*, 44(4):1407–1423, 1998.

[6] M. J. Bruderer, M. F. McKinney, and A. Kohlrausch. The perception of structural boundaries in melody lines of western popular music. *Musicae Scientiae*, 13(2):273–313., 2009.

[7] B.J. L. Campana and E.J. Keogh. A compression based distance measure for texture. In *ICDM*, pages 850–861, 2010.

[8] M. Cooper and J. Foote. Automatic music summarization via similarity analysis. In *ISMIR*, 2002.

[9] H. Ding, G. Trajcevski, P. Scheuermann, X.e Wang, and E. Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *VLDB*, 2008.

[10] J. P. Eckmann, Oliffson S. Kamphorst, and D. Ruelle. Recurrence plots of dynamical systems. *Europhysics Letters*, 4(9):973–977, 1987.

[11] D.P.W Ellis and G.E. Poliner. Identifying "cover songs" with chroma features and dynamic programming beat tracking. In *ICASSP*, volume 4, 2004.

[12] J. Foote. Visualizing music and audio using self-similarity. In *ACM Multimedia*, pages 77–80, Orlando, Florida,, November 1999.

[13] T. Fujishima. Real-time chord recognition of musical sound: a system using common lisp music. In *ICMC*, 1999.

[14] P. Grosche, J. Serr, M. Muller, and J.L. Arcos. Structure-based audio fingerprinting for music retrieval. In *ISMIR*, 2012.

[15] J.S. Iwanski and E. Bradley. Recurrence plots of experimental data: To embed or not to embed? *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 8(4):861–871, 1998.

[16] T. Izumitani and K. Kashino. A robust musical audio search method based on diagonal dynamic programming matching of self-similarity matrices. In *ISMIR*, 2008.

[17] E.J. Keogh, S. Lonardi, C.A. Ratanamahatana, L. Wei, S. Lee, and J. Handley. Compression-based data mining of sequential data. *Data Minining and Knowledge Discovery*, 14(1):99–129, 2007.

[18] M. Li, X. Chen, X. Li, B. Ma, and P.M.B. Vitanyi. The similarity metric. *IEEE Trans. Inf. Theory*, 50(12):3250–3264, 2004.

[19] M. Li and P. Vitanyi. *An introduction to Kolmogorov complexity and its applications*. Springer Verlag, second edition, 1997.

[20] J. Lines, L.M. Davis, J. Hills, and A. Bagnall. A shapelet transform for time series classification. In *SIGKDD*, 2012.

[21] B. Logan and A. Salomon. A music similarity function based on signal analysis. In *ICME*, 2001.

[22] B. Martin, M. Robine, and P. Hanna. Musical structure retrieval by aligning self-similarity matrices. In *ISMIR*, 2009.

[23] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *ISMIR*, Miami, USA, 2011.

[24] J. Paulus, M. Mueller, and A. Klapuri. Audio-based music structure analysis. In *ISMIR*, 2010.

[25] G.H. Wakefield. Mathematical representation of joint time-chroma distribution. In *ASPAAI*, 1999.

DUNYA: A SYSTEM FOR BROWSING AUDIO MUSIC COLLECTIONS EXPLOITING CULTURAL CONTEXT

Alastair Porter, Mohamed Sordo, Xavier Serra
Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
{alastair.porter,mohamed.sordo,xavier.serra}@upf.edu

ABSTRACT

Music recommendation and discovery is an important MIR application with a strong impact in the music industry, but most music recommendation systems are still quite generic and without much musical knowledge. In this paper we present a web-based software application that lets users interact with an audio music collection through the use of musical concepts that are derived from a specific musical culture, in this case Carnatic music. The application includes a database containing information relevant to that music collection, such as audio recordings, editorial information, and metadata obtained from various sources. An analysis module extracts features from the audio recordings that are related to Carnatic music, which are then used to create musically meaningful relationships between all of the items in the database. The application displays the content of these items, allowing users to navigate through the collection by identifying and showing other information that is related to the currently viewed item, either by showing the relationships between them or by using culturally relevant similarity measures. The basic architecture and the design principles developed are reusable for other music collections with different characteristics.

1. INTRODUCTION

The research being performed within the CompMusic [14] project aims to provide new technologies, interfaces and navigation methods for browsing music collections from specific music cultures. In particular we are working with several non-western music traditions and are taking advantage of their specific characteristics to promote an approach in MIR that emphasises the use of domain knowledge in every step of the research project.

Our research goal is to extract musically meaningful descriptions from audio recordings by using as much contextual information as possible. From these descriptions we can develop similarity measures between all the entities that have been identified in a particular music collection.

Several music navigation and discovery systems have been proposed in the last few years [8, 9, 15]. Although differing in visualisation aspects, they all share the notion of computing audio similarity for music discovery. Other systems, like the *Sonic Visualiser* [2], are designed for visualising low-level and mid-level audio features. One of the main advantages of Sonic Visualiser is that it allows the integration of third party algorithms through the use of plug-ins. More recently, Goto et al. proposed *Songle* [3], a system that, among other things, allows users to actively listen to audio while visualising musical scene descriptors by engaging them in correcting estimation errors of the extracted audio features.

Some of these systems make use of related semantic information from the Internet and other sources, however none use cultural-specific information to build a comprehensive music listening experience.

In this paper we introduce Dunya, a system that allows users to explore a given music collection and discover musically relevant relationships between all the items that have a musical relevance. The name Dunya means world [1] in several languages of the cultures that we are studying. In Dunya we are emphasising the concepts of exploration and similarity based on cultural specificity and the design of the interface and of the system architecture emphasises this idea.

For this first version of Dunya we are focusing on Carnatic music, the classical music tradition of the south of India. In the next sections we will describe the architecture of the Dunya system, including the types and the sources of the information that we are using. We then describe the development of our interface for a music collection of the Carnatic music tradition.

2. ARCHITECTURE

This section presents the overall architecture of the Dunya system. Dunya consists of three main aspects (Figure 1). Firstly, raw data is gathered from many different sources. This data includes audio and other information about the music and musicians who performed it. The second aspect is the development of a data schema (a structured data representation) that specifically reflects the concepts that are found in a music tradition. The data schema allows us to store relationships and similarities between items in the database. The final part of Dunya is a graphical interface

[1] Originally coming from the Arabic language, Dunya means more precisely world in a metaphysical sense.

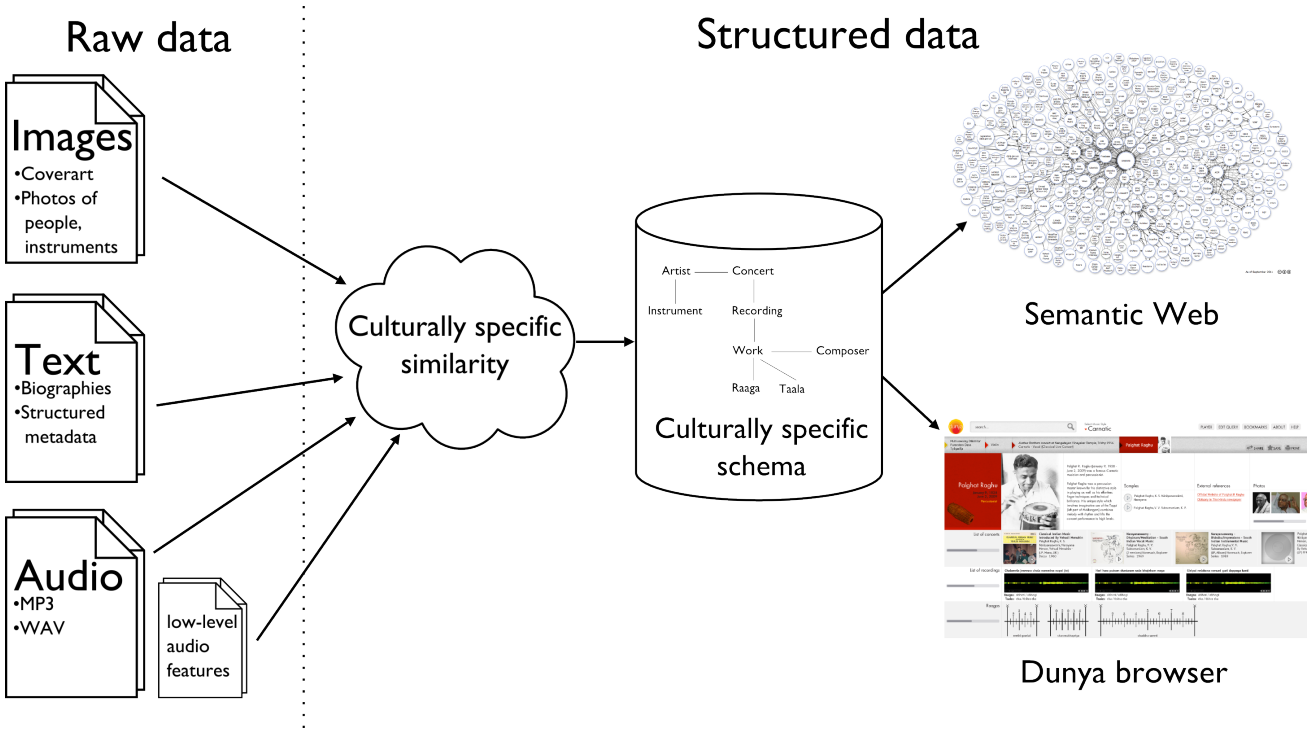


Figure 1. An architecture overview of Dunya. We gather raw data from a number of sources and combine them in a structured database.

for exploring this structured data. The following sections describe these three aspects in more detail.

3. RAW DATA

3.1 Audio recordings

The audio recordings are the core of Dunya. In order to build our audio collection, we bought 300 audio CDs, containing approximately 380 hours of audio and 1900 distinct recordings. We will be extending this collection and hope to reach 500 hours of audio in the next two years. We ripped these audio CDs, and stored the recordings in our file server.

3.2 Editorial metadata

Every audio recording of the collection is accompanied by editorial metadata. Since most audio recordings come from commercial CDs, the editorial metadata comes from the cover or the booklet accompanying the CD. We use MusicBrainz [2] to store and organise this metadata, which includes names of recordings, releases, compositions, composers, artists, and other culture-specific musical concepts. Though Musicbrainz contains metadata for a large number of audio and albums, most of the metadata of the audio recordings that we obtained was not yet in MusicBrainz and so we added it ourselves.

MusicBrainz is a very useful information source for us for a number of reasons. MusicBrainz provides stable identifiers for musical entities, in the form of MusicBrainz Identifiers (MBIDs). These IDs are guaranteed to be unique over all items in the MusicBrainz database. This makes it possible to unambiguously refer to a musician, composition, recording, or released album. For this project we use MBIDs to refer to all entities in our corpus. We see MusicBrainz as a useful tool in many fields of MIR research to explicitly identify datasets used in research. It is able to represent additional relationships in addition to the typically used artist-release-recording model. An example relationship that we preserve in our dataset is the information indicating the people who performed on albums and the instruments that they played. All factual data entered into MusicBrainz becomes freely available under the MusicBrainz project's licenses to be used for any desired use. For a research project such as ours the free availability of the data makes it a useful source for factual information.

3.3 Biographical metadata

Where they exist we use Wikipedia for biographies of artists in our database. When MusicBrainz lists a link to Wikipedia as the artists biography, we use that page. Where such a link doesnt exist we create a link between MusicBrainz and Wikipedia as part of our import process.

3.4 Images

We gather representative images for each item. We get cover art images for commercial CDs from the Cover art archive, [3] a free repository of CD cover art images. We add cover images to the Cover art archive whenever we add new CDs to MusicBrainz. Images of artists and composers

[2] http://musicbrainz.org

[3] http://coverartarchive.org

are currently obtained from Wikipedia and other community websites related to the culture whose music we are working with. Images of instruments used in the musical culture are taken from Wikipedia and from repositories containing images with a freely available license such as Flickr.

3.5 Audio features

We compute low to mid level audio features from the audio recordings in our music collection and store them with the audio recordings. We are currently focusing on melodic and rhythmic dimensions of the music and so have extracted low level audio features such as perceptual amplitude, onsets, and predominant pitch [12]. For the Indian music collections, we also extract the tonic pitch of the lead performing artist [13]. We use the Essentia library [1], an audio analysis library developed by our research group that includes most of the common low and mid level feature analysis algorithms, to compute these features. We are currently performing research on various culture specific descriptors that will be integrated into Essentia and used in Dunya as they become available. For example we are currently working on intonation analysis [6], motivic analysis [11], and *rāga* characterisation [5]. The audio features are stored in the system using the YAML format. [4]

4. CULTURALLY STRUCTURED DATA

After we have gathered all of the raw data that is part of a given corpus we organise it and store it in a database. The information is stored using a schema that is designed to reflect the culture that we are working with. We are currently working to develop cultural-specific schemas for all music cultures that are part of the CompMusic project. For this first version of the system, we have developed a basic schema for Carnatic music which we discuss in Section 5. In this section we discuss the main types of structured data that we create (on top of the raw data) and display in our corpus browsing application.

4.1 High level audio features

Carnatic music is based on well-established melodic and rhythmic frameworks, *rāga* and *tāla* respectively. It is heterophonic with the most dominant characteristics of the music arising from melody. Each melody is set in a specific *rāga*. The main characteristics of *rāga* are the constituent svaras (musical notes), their ascending and descending progressions, their role, and gamakas (ornamentations) sung/-played with them, and characteristic phrases [7]. Several descriptors are extracted from the audio for a comprehensive description of melodies. These include tonic, F0 contours, loudness and timbre of the predominant melodic source. Given a *rāga*, the intonation of each svara is a function of its role and the gamakas sung with it. A compact description of the overall intonation of svaras is extracted using pitch histogram of a given recording [6]. Melodic motives are extracted to retrieve the characteristic phrases of the *rāga* [4, 11]. Intonation description and motives are then used to establish similarity between performances. These, coupled with the pitch contours [1] and the tonic provide a good account of *rāga*.

When characterising the rhythmic structure of Carnatic music the *tāla* cycle is the basis for repetitions of phrases and motives. Tracking the progression of these cycles is essential for a rhythmic description of a piece. Each *tāla* cycle is divided into metrical positions, sometimes unequally spaced. We extract these metrical positions of the *tāla*, which are often called the beats of the *tāla*. Of all the positions, the first metrical position of the cycle, called the sama is the most important. The work towards the complete description of the *tāla* is in progress. Though tempo is not clearly defined in Carnatic music, we presently extract the rhythmic density based on the onsets, which is an indicator of the tempo of the piece.

The characterisation of rhythmic and melodic phrases in Carnatic music and the extraction of these phrases is research in progress. In addition we are also working towards the structural segmentation of the music piece at different time scales such as motives of a phrase, phrases of a piece, and pieces of a concert. These structural segmentation features provide a perspective about the structure of the performance and all these melodic description components are the building blocks for developing melodic similarity measures.

4.1.1 Analysing audio

We have developed an audio processing framework that is used to compute the high level audio features from the audio recordings in our corpus. Because the project is still under active development it is important to be able to re-analyse the audio whenever our analysis algorithm is modified, and to run the analysis on newly added audio. Our audio processing is performed on a cluster of servers. Each server in the cluster has access to a shared NFS disk that contains the entire audio corpus. A central PostgreSQL database keeps track of which audio files have been analysed by the algorithms. We use the Celery task queue system [5] to distribute the analysis tasks to the machines in the cluster. When audio is added to the corpus the cluster master directs a worker machine to analyse an album using a specific algorithm. Each worker performs analysis on its workload and stores the results with the audio files, where they become available to our browser application. By performing the analysis using a cluster of servers we are able to add more servers as our corpus grows. The result of the analysis is stored in a log, allowing an administrator to quickly check if there there was a problem with the analysis of an album.

4.2 Structured editorial metadata

Our system stores structured editorial metadata for all items in the corpus in a PostgreSQL database. This includes information from MusicBrainz as well as information that

[4] http://www.yaml.org

[5] http://www.celeryproject.org

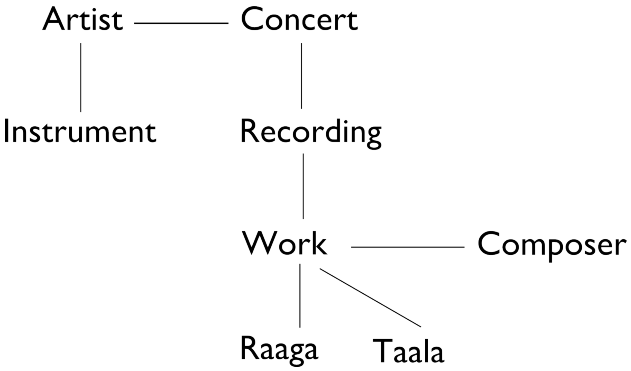


Figure 2. The main entity types in Carnatic music and their relationships.

we have obtained from other online sources such as Wikipedia or websites that contain information and discussions related to the style of music that we are working with. Some information can be represented as ontologies, explicitly expressing the relationships between concepts used in that culture. We store the RDF ontologies in a graph database allowing us to navigate through related concepts in our database. We initially attempted to access data on demand via APIs provided by each service that provides data for our corpus, however we abandoned this approach for two reasons. Firstly, the time taken to perform queries on all dependent sites in order to display the relevant information on a page would take too long. Secondly, by controlling all of the metadata in our database we are able to create relationships between items in the database that may not exist in the other data sources we are using. We keep references to all of our external sources in order to direct users to them if they wish to find more information. By storing the date at which data was taken from an external source we can periodically refresh the data by checking if the source information has changed since we last accessed it. Because we store all of the relevant information in a local database it is also possible to take the entire collection and access it locally in situations where an Internet connection is not available.

5. BROWSER

The Dunya front-end is a web-based application written in Python using the Django web framework, [6] with an interactive interface written in Javascript. The interface is designed to work on a range of computing devices.

We also plan to make the same data that we show in the browser available in a machine-readable format. To do this we will use the Music Ontology initiative [10] to provide an endpoint that lets people write systems to query the items in our database using publicly known identifiers such as MBIDs.

For the Carnatic music collection we focus on seven main concepts that are important to the culture surrounding the music. The concepts are: artist, composer, work, recording, concert, *rāga*, and *tāla*. We refer to each instance of one of these concepts as *entities*. For every entity in our database we are able to show the information that we have gathered from linked sites as well as suggestions for other entities that are related to the currently viewed one. *rāga* and *tāla* are concepts that are represented as attributes of a work. These works are performed by musicians at concerts. Concerts are a fundamental musical unit in the Carnatic culture and so we use these as a type of entity. Almost all albums of this kind of music are recordings of live concerts. We have developed ontologies for the *rāga* and instrument entity types for Carnatic music.

[6] https://www.djangoproject.com

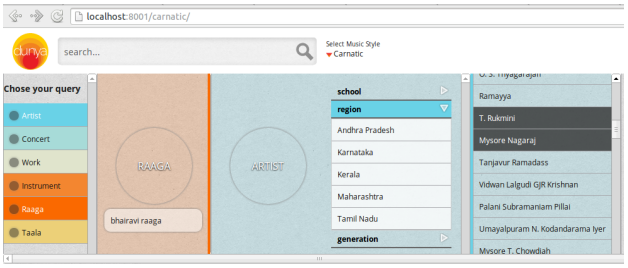


Figure 3. The search navigation interface showing a query for items related to a specific *rāga* and two artists.

5.1 Search and discovery

In Dunya we provide two different ways to find music and other items in the collection. The search interface allows a user to type in the name of any entity or concept that we store in the database. As the user types in the search box we suggest recommendations that they might wish to use. We show results of all entities that contain the text that the user has entered. When searching we also consider alternative spellings of entities due to different ways of romanising Indian text to Latin characters. We make use of alternative spellings entered in MusicBrainz, as well as a fuzzy Levenshtein distance-based measurement.

The second search interface is an interactive browser (Figure 3). For each of the core concepts (artist, composer, work, concert, *rāga*, and *tāla*) we show filters that allow users to find entities that match certain criteria. For example, an artist can be filtered by a time period, the musical school that they trained in, or the region that they are from. The data for these filters comes from online sources such as biographies. A user can query with as many concepts as they like. As more queries are added the number of relevant results returned is reduced.

5.2 Entity pages

Once the user selects an entity in the search results we show a page describing it. At the top of the page we show editorial and relationship data. This information comes from the information that is present in Musicbrainz, as well as from other linked information that we have discovered. For example, the page for an artist shows basic biographical information such as their name and birth and death dates. The page includes biographical information if we

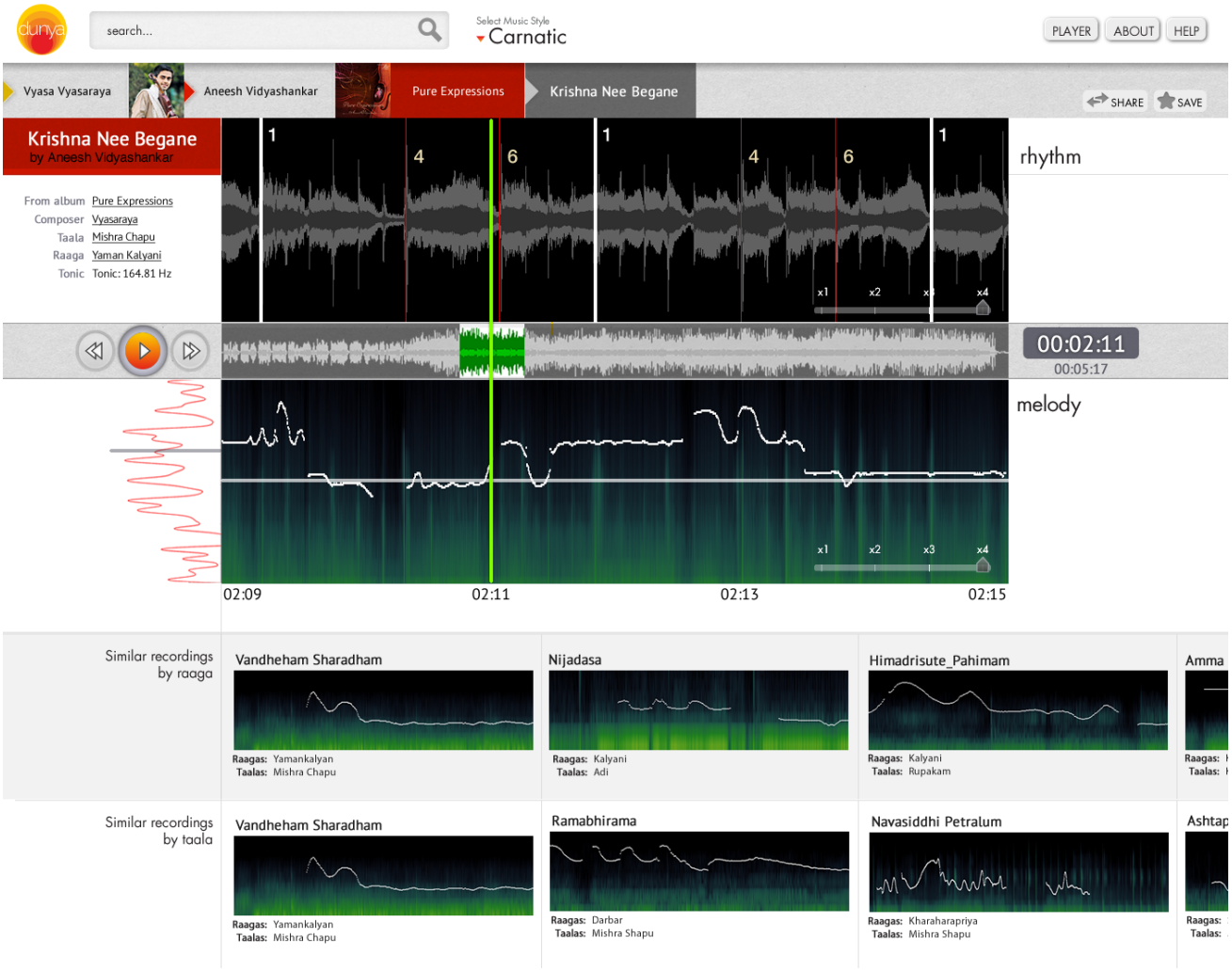


Figure 4. Visual display of a recording in Dunya. Suggestions for similar recordings are shown underneath the audio player.

have been able to source it with a link to the original entry. Where available we also show an image of the entity. Artist pages show an image of the person, while for concerts we show the cover art of a CD if available. We also show links to other related entities. For example, the page for an artist will show concerts that they have played in and the instrument that they play. The page for a *rāga* shows composers that often wrote compositions using that *rāga* and popular compositions written using it.

One core concept in Dunya is to be able to listen to a relevant sound sample for each entity. In the display of each entity there is a button that when pressed plays a sound fragment that has been automatically chosen for its relevance. From the chosen fragment a user would be able to access the entity of the full recording that the segment comes from. Audio recordings are added to a global playlist that continues playing as the user navigates around the corpus. At any point in time the user can view the playlist, edit it, or go to the recording page of the currently playing track.

5.3 Recording entity

One of the primary entity interfaces that we are developing in Dunya is the view that represents a single audio recording. Most of the technologies developed in this project result in new methods to analyse and understand audio and the recording page is where we visually show the results of the analysis while playing the recording (Figure 4). The recording interface overlays the extracted features on top of a visual representation of the recording. In this version of the system we show two types of low-level audio features and three types of higher level features on the recording page.

We show two different representations of the audio. The upper view shows a time-domain waveform, and the lower one shows a spectral view. These views can be zoomed in to see data in more detail. For performance reasons we pre-render the low-level visualisations at different zoom levels and draw the features on top of them. Between the two views there is a full-length representation of the time-domain signal. This gives an indication of the progress through the track even if the other views are zoomed in. On the time-domain waveform we show extracted rhyth-

mic information. This information indicates the metrical positions of the *tāla* of the recording. On the lower spectrogram image we show the pitch contour of the extracted melody of the recording. To the left of the spectrogram is a histogram of the predominant pitches in the entire recording. We indicate the tonic pitch of the recording with a horizontal line across this image. A playback cursor is shown on both of these visualisations as the recording is being played back. As the recording plays we indicate the current predominant pitch on the histogram to the side of the melody pane.

Below the audio player we show recordings that are similar according to the distance metrics that we have developed. Currently we show recordings that have a similar *tāla* (metrical framework) and *rāga* (melodic framework). The list of recordings is ordered by the similarity of the recordings to the current recording. The first recordings in the list may have the same *rāga* as the current recording, however following recordings may use *rāgas* that are are related based on the ontologies that we have developed.

6. CONCLUSIONS

We have presented Dunya, a music discovery system being developed as part of the CompMusic project designed to support the exploration of a music corpus using concepts relevant to the particular musical culture of the corpus. The application displays the typical information shown in music playback applications, but also includes additional information collected from a variety of sources and information that has been automatically computed from the recorded audio. It stores a complex set of relationships between the items in the database. With this information a user can explore given music collection and discover musically relevant relationships. The architecture of the system and the interface have been designed to support cultural specificity, starting in this paper with Carnatic music. In the near future we will extend the system to the rest of the music collections we are working with and we will continue integrating research results resulting from the CompMusic project to enhance the discovery capabilities of Dunya.

7. ACKNOWLEDGEMENTS

The authors would like to thank Ajay Srinivasamurthy, Gopala K. Koduri, and Sankalp Gulati for contributing to the descriptions of their respective research that is being used in the CompMusic project. Pere Esteve assisted with the design and development of the graphical interface of the Dunya Browser. The CompMusic project is funded by the European Research Council under the European Union's Seventh Framework Program (FP7/2007-2013) / ERC grant agreement 267583

8. REFERENCES

[1] 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 *Proceedings of the International Society for Music Information Retrieval Conference*, 2013.

[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 7th International Conference on Music Information Retrieval*, 2006.

[3] Masataka Goto, Kazuyoshi Yoshii, Hiromasa Fujihara, Matthias Mauch, and Tomoyasu Nakano. Songle: A web service for active music listening improved by user contributions. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 311–316, 2011.

[4] Vignesh Ishwar, Ashwin Bellur, and Hema A Murthy. Motivic analysis and its relevance to raga identification in carnatic music. In *Proceedings of the 2nd CompMusic Workshop*, 2012.

[5] Gopala K. Koduri, Sankalp Gulati, Preeti Rao, and Xavier Serra. Raga Recognition based on Pitch Distribution Methods. *Journal of New Music Research*, 41(4):337–350, 2012.

[6] Gopala K. Koduri, Joan Serrà, and Xavier Serra. Characterization of intonation in carnatic music by parametrizing pitch histograms. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 199–204, 2012.

[7] TM Krishna and Vignesh Ishwar. Carnatic music: Svara, gamaka, motif and raga identity. In *Proceedings of the 2nd CompMusic Workshop*, 2012.

[8] François Pachet, Jean-Julien Aucouturier, Amaury La Burthe, Aymeric Zils, and Anthony Beurive. The cuidado music browser: an end-to-end electronic music distribution system. *Multimedia Tools and Applications*, 30(3):331–349, 2006.

[9] Elias Pampalk and Masataka Goto. Musicsun: A new approach to artist recommendation. In *Proceedings of the 8th International Conference on Music Information Retrieval*, pages 101–4, 2007.

[10] Yves Raimond, Samer Abdallah, Mark Sandler, and Frederick Giasson. The music ontology. In *Proceedings of the 8th International Conference on Music Information Retrieval*, pages 417–422, 2007.

[11] Joe Cheri Ross, TP Vinutha, and Preeti Rao. Detecting melodic motifs from audio for hindustani classical music. In *Proceedings of the International Society for Music Information Retrieval Conference*, 2012.

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

[13] Justin Salamon, Sankalp Gulati, and Xavier Serra. A multipitch approach to tonic identification in indian classical music. In *Proceedings of the International Society for Music Information Retrieval Conference*, 2012.

[14] Xavier Serra. A Multicultural Approach to Music Information Research. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, 2011.

[15] George Tzanetakis. Musescape: An interactive content-aware music browser. In *Proceedings of the 6th Conference on Digital Audio Effects (DAFX)*, 2003.

AN ANALYSIS OF CHORUS FEATURES IN POPULAR SONG

Jan Van Balen[1] **John Ashley Burgoyne**[2] **Frans Wiering**[1] **Remco C. Veltkamp**[1]
[1]Utrecht University, Department of Information and Computing Sciences
[2]Universiteit van Amsterdam, Institute for Logic, Language and Computation
{j.m.h.vanbalen,f.wiering,r.c.veltkamp}@uu.nl, j.a.burgoyne@uva.nl

ABSTRACT

This paper presents a computational study of the perceptual and musicological audio features that correlate with the structural function of sections in pop songs, specifically the chorus. Choruses have been described as more prominent, more catchy and more memorable than other sections in a song, yet chorus detection applications have always been primarily based on identifying the most-repeated section in a song. Inspired by cognitive research rather than applied signal processing, this computational analysis compiles a list of robust and interpretable features and models their influence on the 'chorusness' of a collection of song sections from the Billboard dataset. This is done through the unsupervised learning of a probabilistic graphical model. We show that timbre and timbre variety are more strongly related to chorus qualities than harmony and absolute pitch height. A regression and a classification experiment are performed to quantify these relations.

1. INTRODUCTION

1.1 Chorus Analysis

The term *chorus* originates as a designation for those parts of a musical piece that feature a choir or other form of group performance. When solo performance became the norm in popular music, the term chorus was retained to indicate a repeated structural unit of musical form. In terms of musical content, the chorus has been referred to as the 'most prominent', 'most catchy' or 'most memorable' part of a song and 'the site of the more musically distinctive and emotionally affecting material' [4, 10].

While agreement on which section in a song constitutes the chorus generally exists among listeners, attributes such as 'prominent' and 'catchy' are far from understood in music cognition and cognitive musicology [6]. On the other hand, as a frequent subject of study in the domain of Music Information Retrieval, the notion of chorus has been shown to correlate with a number of computable descriptors. Yet when studied more closely, the chorus detection systems that locate choruses most successfully turn out to rely on rather contextual cues such as the amount of repetition and relative energy of the signal, with more sophisticated systems also taking section length and position within the song into account [4, 5]. The central research question of this paper is therefore: in which computable properties of popular music are choruses, when compared to other song sections, musically distinct?

The main motivations for a deeper study of the particularities of choruses are two-fold: first, the chorus being a central element of form in popular music, insight may be gained in the popular song as a medium, and conscious as well as unconscious choices in songwriting may be unveiled. Second, as choruses can be related to a catchy and/or memorable quality, to the notion of hooks, and perhaps to a general cognitive salience underlying these aspects, the nature of choruses may indicate some of the musical properties that constitute this salience, prominence or memorability.

This investigation relates to chorus detection as known in Music Information Retrieval, but it does not have the same goal. While chorus detection systems are built to locate the choruses given unsegmented raw audio for a song, this investigation aims to use similar and novel computational methods to improve our understanding of choruses.

1.2 Related Work

Existing work on chorus detection strongly relates to audio thumbnailing, music summarization and structural segmentation. Audio thumbnailing and music summarization refer to the unsupervised extraction of the most representative short excerpt from a piece of musical audio, and often rely on full structure analysis as a first step. An overview of relevant techniques is given by Paulus et al. [13].

Definitions of the chorus in the MIR literature characterize the chorus as repeated, prominent and catchy. Since the last two notions are never formalized, thumbnailing and chorus detection are essentially reduced to finding the most often-repeated segment or section. A few chorus detection systems make use of additional cues from the song audio, including RefraiD by Goto and work by Eronen [4, 5]. RefraiD makes use of a scoring function that favors segments C occuring at the end of a longer repeated chunk ABC and segments CC that consistently feature an internal repetition. Eronen's system favors segments that occur $\frac{1}{4}$ of the way through the song and reoccur near $\frac{3}{4}$, as well as segments with higher energy. In most other cases, heuristics are only used to limit the set of candidates from which

the most frequent segment is picked, e.g. restricting to the first half of the song or discarding all segments shorter than 4 bars.

Paulus and Klapuri use a Markov model to label segments given a set of tags capturing which segments correspond to the same structural section (e.g. ABCBCD) [12, 14]. This approach performs well on *UPFBeatles*, a dataset of annotated Beatles recordings, and fairly well on a larger collection of songs (TUTstructure07). [1] An n-gram method with $n = 3$ and a Variable-order Markov Model come out as the best techniques. The same methods have also been enhanced by using limited acoustic information: section loudness and section loudness deviation [14]. This boosts the best performance (in terms of per-section accuracy) by up to 4 percent for TUTstructure07. Whether the model could be improved with more acoustic information remains an open question that this paper aims to address.

The contribution of this paper is to introduce the notion of chorusness, and a statistical model of this measure for MIR applications, for popular music understanding and for popular music perception and cognition, using a novel and rigorous take on corpus-scale audio music analysis.

2. METHODOLOGY

The research question formulated above is addressed by means of a statistical analysis of a selection of music descriptors, computed over a dataset of pop songs. The *Billboard* dataset, described in section 2.1, will be used as the ground truth. The expert structure annotations available for these data allow for parsing audio descriptors, detailed in part 2.2 of this paper, into per-section statistics. The analysis of the resulting variables will then be formalized by learning a probabilistic graphical model from the data, as explained in section 2.3.

2.1 The Dataset

The *Billboard* dataset is a collection of time-aligned transcriptions of the harmony and structure of over 1000 songs selected randomly from the *Billboard* 'Hot 100' chart in the United States between 1958 and 1991 [2]. The annotations include information about harmony, meter, phrase, and larger musical structure. The Billboard dataset is one of the largest and most diverse popular music datasets for which expert structure annotations exist and one of few to be consistently sampled from actual pop charts. It can be expected to reflect both important commonalities and significant trends in popular music of the period of focus. It includes a wide variety of genres, and suits the goal of drawing broadly-applicable musicological conclusions, making it the best available dataset for analysis of popular music choruses. For the present study, the complete v1.2 release is used (649 songs), and of the annotations, only the structural annotations are retained.

The structural annotations in the dataset follow the format and instructions established in the *SALAMI* project [18]. The transcriptions contain start and end times for every section and section labels for almost all sections. The section labels the annotators were allowed to assign were restricted to a list of 22, some of which were not used. The most frequently recurring section labels are: verse (34% of total annotated time), chorus (24%), intro, solo, outro and bridge. The total number of sections, including the unlabeled ones, is 7762.

[1] Dataset descriptions and links at `http://www.cs.tut.fi/sgn/arg/paulus/structure.html`

2.2 Perceptual Audio Features

The proposed corpus analysis–centered study requires somewhat different kinds of descriptors than traditionally used in machine-learning applications in Music Information Retrieval. Four important constraints are applied. First, all descriptors are demonstrated correlates of a relevant perceptual or cognitive attribute of music. Second, we favor musically interpretable descriptors. A classic example of an audio feature with demonstrated perceptual correlates but low interpretability are MFCC's [1]. Third, only transparent statistics over these features are considered. Higher order statistical moments and learned codewords can be very informative from an engineering perspective, but amount to highly uninterpretable descriptions. Finally, we also limit the set to a small number of handpicked descriptors since the amount of data required for the proposed analysis grows exponentially with the number of variables. All features are one-dimensional.

Loudness The loudness descriptor is the standard psychological analogy of energy. It is obtained through comparison of stimuli spectra and a standardized set of equal loudness curves. We will make use of the implementation by Pampalk [11]. The model applies outer-ear filtering and a spreading function before computing specific loudness values (N_k in sones) per band k and summing these values over all bands to obtain the total loudness T:

$$S = \max_k(N_k) + 0.15 \cdot \sum_{k \neq \max} N_k \qquad (1)$$

where the factor 0.15 serves as a weighting that emphasizes the strongest band's contribution. For every section, the loudness *mean* is computed and stored, as well as the inter-quartile range (**Loudness IQR**), as a measure of the section dynamics.

Sharpness The sharpness descriptor is the psychoacoustic analog of the spectral centroid. It characterizes the balance between higher- and lower-band loudness. We will make use of the specific loudnesses N_k as computed by Pampalk [11] and summing as formulated by Peeters [15]:

$$A = 0.11 \times \sum_k g(k) \cdot k \cdot N_k, \quad \text{where} \qquad (2)$$

$$g(k) = \begin{cases} 1 & k < 15 \\ 0.066 \times \exp(0.171 \cdot k) & k > 15 \end{cases} \qquad (3)$$

For every section, we use the *mean* sharpness.

Roughness Like the loudness descriptor, roughness is a mathematically defined psychoacoustic measure. It

characterizes a timbral property of complex tones, relating to the proximity of its consituent partials. We will make use of the MIRToolbox implementation by Lartillot et al. [9], which is based on a model by Plomp and Levelt [16]. Since the roughness feature is very nonlinearly distributed, the roughness feature is summarized for every section by taking its *median*.

MFCC MFCCs are established multidimensional correlates of several aspects of timbre, designed to be maximally independent. Typically around 13 MFCC coefficients are used. In this model, the descriptor of interest is the variety in timbre. This will be modeled by computing the trace of the square root of the MFCC covariance matrix, a measure of the timbre *total variance*. The MFCCs are computed following [11], and the first component (directly proportional to energy) is discarded.

Chroma Variance Chroma features are widely used to capture harmony and harmonic changes. In the most typical implementation, the chroma descriptor or pitch class profile consists of a 12-dimensional vector, each dimension quantifying the energy of one of the 12 equal-tempered pitch classes. These energies can be obtained in several ways. The *Chordino* chroma features distributed along with the Billboard dataset are used here. [2]

In this study, the variety in the section's harmony will be measured by modeling the normalized chroma features $\mathbf{p}$ as a 12-dimensional random variable and estimating a Dirichlet distribution to the total of all of the section's chroma observations. Note that estimating just the total variance, as done for MFCC, would neglect the normalisation constraint on chroma vectors and the dependencies it entails between pitch classes. The Dirichlet distribution $\mathcal{D}_{12}(\boldsymbol{\alpha})$, can be written:

$$f(\mathbf{p}) \sim \mathcal{D}_{12}(\boldsymbol{\alpha}) = \frac{\Gamma(\sum_{k=1}^{12} \alpha_k)}{\prod_{k=1}^{12} \Gamma(\alpha_k)} \prod_{k=1}^{12} p_k^{\alpha_k - 1}, \quad (4)$$

where Γ is the Gamma function, and can be seen as a distribution over distributions. We use the sum of the parameters $\boldsymbol{\alpha}$, commonly refered to as the Dirichlet precision s:

$$s = \sum_{k=1}^{12} \alpha_k \quad (5)$$

The Dirichlet precision quantifies the difference between observing the same combination of pitches thoughout the whole section (high precision) and observing many different distributions (low precision) [3]. There is no closed-form formula for s or $\boldsymbol{\alpha}$, but several iterative methods exist that can be applied to obtain a maximum-likelihood estimation (e.g. Newton iteration). Fast fitting can be done using the *fastfit* Matlab toolbox by Minka. [3]

Pitch Salience The notion of pitch salience exists in several contexts. Here, it will refer to the strength of (a discrete set of) pitches, i.e. a measure of the combined strength of a frequency and its harmonics, as in [17]. The *mean* of the strongest (per frame) pitch strength will be computed for every section.

Pitch Centroid The pitch height of polyphonic audio can be defined and computationally approximated in several ways. A predominant fundamental frequency in the classic sense is not always reliably found, especially for the case of polyphonic pop music. We therefore use the more robust *pitch centroid*. We define this as the average pitch height with all pitches weighted by their salience. Note that the pitch salience profile used here spans multiple octaves and exploits spectral whitening, spectral peak detection and harmonic weighting in order to capture only tonal energy and emphasize the harmonic components.

Our feature set includes the section *mean* of the pitch centroid as well as the inter-quartile range of this measure, represented by the **Pitch Centroid IQR**.

Section Length The length of the section in seconds.

Section Position The position of the section inside the song is included as a number between 0 and 1.

The descriptors above have proven useful in a variety of other contexts and fall within the constraints listed above. The first three originate in psychoacoustics, the next three descriptors stem from MIR research. The pitch centroid is a novel descriptor designed with robustness in mind.

2.3 Analysis Method

2.3.1 PGM

The resulting descriptors make up a dataset of 7762 observations (sections) and 12 variables (descriptors) for each observation: the above perceptual features and one section label. These data will be used to model what features correlate with a section being a chorus or not. However, modeling all dependencies between a set of variables quickly leads to complex representations that are hard to manage.

Probabilistic graphical models (PGM) are graph-based representations of such complex higher-dimensional distributions that focus on modeling the direct probabilistic interactions between variables [8]. Unlike a correlation matrix, they encode which variables are conditionally dependent, i.e. correlate *given* the state of all other variables, regardless of any indirect effects from other influencing variables. Examples of the use of a PGM in music analysis can be found in [3].

Essential to a PGM is its graph structure. It is typically constructed using prior expert knowledge but, with enough data, can also be learned. Learning the PGM structure generally requires a great amount of conditional independence tests. The *PC-algorithm* optimizes this procedure and, in addition, provides information about the direction of the dependencies [7]. When not all directions are found, a partially directed graph is returned.

Given the limitations of currently available software packages, an important practical requirement for learning the

[2] http://www.isophonics.net/nnls-chroma

[3] http://research.microsoft.com/en-us/um/people/minka/software/fastfit/

graph structure is that the variables follow similar distributions, e.g. all discrete, or all continuous Gaussian. In the analysis in the next section, all data are modeled as continuous. This means that also the **Section Type** variable will have to be modeled as continuous. We do this by introducing the notion of **Chorusness**.

2.3.2 Chorusness

The Chorusness variable is derived from the **Chorus Probability** p_C, a function over the domain of possible section labels. The chorus probability $p_C(T)$ of a section label T is defined as the probability of a section with label T being labeled 'chorus' by an independent annotator. In terms of the annotations x_1 and x_2 of two independent listeners, $p_C(T)$ can be written:

$$p_C(T) = \frac{p(x_1 = C|x_2 = T) + p(x_2 = C|x_1 = T)}{2}, \tag{6}$$

where C refers to the label 'chorus'.

The Billboard dataset has been annotated by only one expert per song, therefore it contains no information about any of the $p_C(T)$. However, in the SALAMI dataset, annotated under the same guidelines and conditions, two independent annotators were consulted per song [18]. The annotators' behaviour can therefore be modeled by means of a confusion matrix $M(T_1, T_2) \in [0, 1]^{22\times 22}$:

$$M(T_1, T_2) = f(x_1 = T_1 \cap x_2 = T_2) \tag{7}$$

with frequencies f in seconds (of observed overlapping labels T_1 and T_2). Since the two annotators are interchangeable (and have in fact been randomized), M may be averaged out to obtain a symmetric confusion matrix $M^\star$:

$$M^\star = \frac{M + M^T}{2} \tag{8}$$

From here we can obtain the empirical Chorus Probability:

$$p_C(T) = \frac{M^\star(T, C)}{\sum_k M^\star(T, k)} \in [0, 1]. \tag{9}$$

Chorus Probability values for every section type were obtained from the Codaich-Pop subset of the SALAMI dataset (99 songs). Finally, the Chorus Probability is scaled monotonically to obtain the Chorusness measure $C(T)$, a standard *log odds ratio* of p_C:

$$C(T) = \log\left(\frac{p_C(T)}{1 - p_C(T)}\right) \in (-\infty, \infty). \tag{10}$$

It ranges from -8.41 (for the label 'spoken') to 0.83 (for the label 'chorus').

2.3.3 Implementation

Before the model learning, a set of Box-Cox tests is performed to check for rank-preserving polynomial transformations that would make any of the variables more normal. The Chroma Precision s is found to improve with a power parameter $\lambda = -1$, and therefore scaled as:

$$S = \frac{s^\lambda - 1}{\lambda} = 1 - \frac{1}{s} \tag{11}$$

The Section Length, Loudness IQR and Pitch Centroid IQR are found to improve with a log transform. Weeding out divergent entries in the dataset leaves us with a subset of 6462 sections and 12 variables.

The R-package *pcalg* implements the PC-algorithm. Beginning with a fully connected graph, it estimates the graph skeleton by visiting all pairs of adjecent nodes and testing for conditional independence given all possible subsets of the remaining graph. [4] The procedure is applied to the 6462×12 dataset, with 'conservative' estimation of directionalities, i.e. no directionality is forced onto the edges where no V-structures were found indicating a specific direction.

3. ANALYSIS RESULTS

The resulting graphical model is shown in Figure 1. It is obtained with $p < 3.5 \times 10^{-5}$, the significance level required to bring the overall probability of observing one or more edges due to chance, under 5 percent. In terms of the significance level α_{CI} of the conditional independence tests and α_{PGM} of the model:

$$\alpha_{\text{CI}} = 1 - (1 - \alpha_{\text{PGM}})^{1/n} \approx \frac{\alpha_{\text{PGM}}}{n} \tag{12}$$

with $\alpha_{\text{PGM}} \ll 1$ (here 0.05) and n the number of tests performed (~ 1500). Note that $p \approx 10^{-5}$ is a conservative parameter setting for an individual test. As a result, we may choose to view the model as a depiction of dependencies rather than independencies, since the latter may always be present at a lower significance than required by the α.

3.1 Discussion

At least three kinds of feature relations are expected. First, there are the correlations between features that are closely related on the signal level (black edges): Loudness and Pitch Salience, for example, measure roughly the same aspects of a spectrum (and can be expected to be proportional to roughness), and so do Sharpness and Pitch Centroid. Roughness is a highly non-linear feature that is known to be proportional to energy. The model reflects this.

The second kind of correlations are the relations between variance-based features and the Section Length variable. Musically, it is expected that longer sections allow more room for an artist to explore a variety of timbres and pitches. This effect is observed for Chroma Variance and Pitch Centroid IQR, though not for MFCC Variance and Loudness IQR. Interestingly, correlations with Section Length point *towards* it rather than away (dotted edges): the length of a section length is a result of its variety in pitch and timbre content, rather than a cause. The importance of this distinction can be debated.

Third, some sections might just display more overall variety, regardless of the section length. This would cause different variances to relate, resulting in a set of arrows between the four variance features. Four such relations are observed (lighter, orange edges).

[4] `http://cran.r-project.org/web/packages/pcalg/`

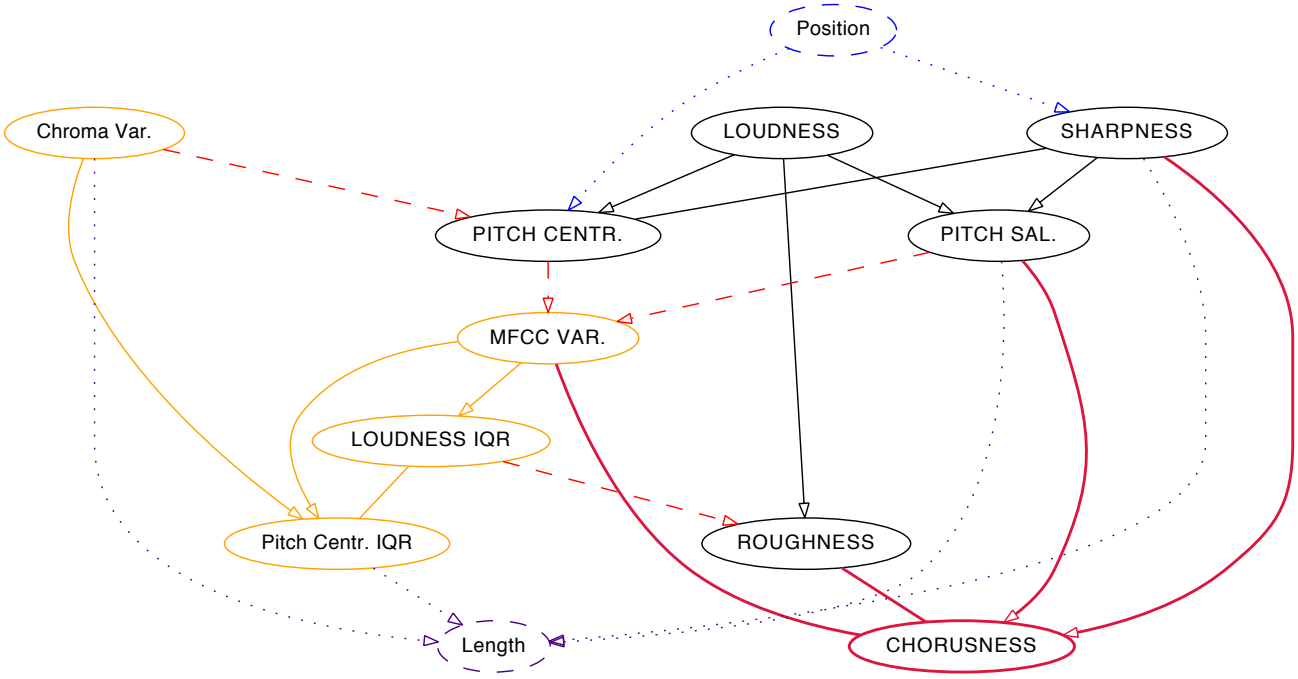


Figure 1. Graphical model of the 11 analyzed perceptual features and chorusness variable C. $\alpha_{\text{PGM}} = 0.05$.

We now note that Sharpness, Pitch Salience and Roughness predict Chorusness, as well as the MFCC Variance (bold edges). All of these can be categorized as primarily timbre-related descriptors. Section Length, Section Position and Chroma Variance are d-separated from Chorusness, i.e. no direct influence between them has been found. The status of Pitch Centroid, Loudness, and Loudness IQR is uncertain. Depending on the true directionality of the Chorusness, MFCC Variance and Roughness relations, they may be part of the Chorusness *Markov blanket*, the set of Chorusness' parents, children, and parents of children, which d-separates Chorusness from all other variables [8].

Also interesting are the more unexpected dependencies. For example, two variables depend directly on the Section Position, while Chorusness does not. This may be due to the limitations of the normal distribution by which all variables are modeled; it is fair to say that it might not reflect the potentially complex relation of Chorusness variable and Section Position. However, the Position variable does predict Sharpness and Pitch Centroid to some extent (dotted edges). A simple regression also shows both variables correlate *positively*. This suggests some kind of overtime intensification along the frequency dimension exists in the songs of the Billboard corpus.

Finally, the dashed red edges in the diagram indicate dependencies that are most unintuitive. Tentative explanations may be found, but since they have no effect on Chorusness, we will omit such speculations here.

3.2 Regression

We are more interested to see in more detail how the set of Chorusness-related features predict our variable of interest. Table 1 lists the coefficients of a linear regression on the Chorusness variable and its Markov blanket, i.e. those variables for which a direct dependency with chorusness is apparent from the model. Since there is no certainty about the exact composition of the Markov blanket, all candidates are included. Note that, having defined Chorusness as a log odds ratio, this linear regression is *de facto* a common form of logistic regression on the section's original Chorus Probability $p_C \in [0, 1]$.

		95% CI	
	Coeff.	LL	UU
Sharpness	0.11	0.10	0.13
MFCC variance	0.12	0.09	0.15
Roughness	0.12	0.08	0.16
Pitch Salience ($\times 10$)	0.04	0.03	0.05
Loudness	0.03	-0.01	0.06
Loudness IQR	-0.33	-0.48	-0.18
Pitch Centroid	0.10	0.07	0.12

Table 1. Results of a multivariate linear regression on the Chorusness' Markov blanket ($p < 10^{-15}$ for all coeff.). CI=confidence interval, LL=lower limit, UU=upper limit.

One can see that all features but the Loudness IQR have positive coefficients. We conclude that, in this model, sections with high Chorusness are louder, sharper and rougher than other sections. Chorus-like sections also feature a slightly higher and more salient pitch, a smaller dynamic range and greater variety in MFCC timbre.

4. VALIDATION

Finally, a validating experiment is performed. It consists of the evaluation of a 2-way classifier that aims to label

sections as either 'chorus' or 'non-chorus'. A k-nearest neighbour classifier ($k = 1$) is trained on half of the available sections, and tested on the other half (randomly partitioned). This procedure is repeated 10 times to obtain an average precision, recall and F-measure. The results are positive: using just the Markov blanket features of table 1, the classifier performs better than random: $F = 0.52$, 95% CI $[0.51, 0.52]$ vs. a maximum random baseline of $F = 0.36$. The classifier also performs better than one that uses all features ($F = 0.48$), or only Loudness and Loudness IQR ($F = 0.48$), the features used in [14].

5. CONCLUSIONS

This paper presents a computational study of the musically interpretable and robust audio descriptors that correlate with the 'chorusness' of sections in pop songs. A selection of existing and novel perceptual and computational features is presented. The set has been analyzed using a probablistic graphical model and a measure of chorusness that is derived from annotations and an inter-annotator confusion matrix. The resulting model was complemented with a regression on the most important variables. The results show that choruses and chorus-like sections are louder, sharper and more rough, and feature a higher and more salient pitch, a smaller dynamic range and greater variety of MFCC-measurable timbre than other sections.

The results obtained in a validating classification experiment show that our model does not reach the level of accuracy obtained by the state of the art techniques that incorporate repetition information. However, it demonstrates for the first time that there is a class of complementary musical information that, independently of repetition, can be used to locate choruses. This suggests that our model can be applied to complement existing structure analysis applications, while repetition information and section order can in turn enhance our model of chorusness for further application in popular music cognition research and audio corpus analysis.

6. ACKNOWLEDGEMENTS

This research is supported by the NWO CATCH project COGITCH (640.005.004), and the FES project COMMIT/.

7. REFERENCES

[1] J. J. Aucoutourier and E. Bigand: "Mel Cepstrum & Ann Ova: The Difficult Dialog between MIR and Music Cognition," *Proc. of the Int. Society for Music Information Retrieval Conf.*, pp. 397–402, 2012.

[2] J. A. Burgoyne, J. Wild and I. Fujinaga: "An Expert Ground Truth Set for Audio Chord Recognition and Music Analysis," *Proc. of the Int. Society for Music Information Retrieval Conf.*, pp. 633–638, 2011.

[3] J. A. Burgoyne: *Stochastic Processes and Database-driven Musicology*, PhD Thesis, McGill University, Montréal, Québec, 2011.

[4] A. Eronen and F. Tampere: "Chorus Detection with Combined Use of MFCC and Chroma Features and Image Processing Filters", *Proc. of the Int. Conf. Digital Audio Effects*, 2007.

[5] M. Goto: "A Chorus-section Detecting Method for Musical Audio Signals," *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pp. 437–440, 2003.

[6] H. Honing: "Lure(d) into listening: The Potential of Cognition-based Music Information Retrieval," *Empirical Musicology Review*, Vol. 5, No. 4, 2010.

[7] M. Kalisch et al.: "Causal Inference Using Graphical Models with the R Package pcalg," *Journal of Statistical Software*, Vol. 47, No. 11, pp. 1–26, 2012.

[8] D. Koller and N. Friedman: *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.

[9] O. Lartillot and P. Toiviainen: "MIR in Matlab (II): A Toolbox for Musical Feature Extraction from Audio," *Proc. of the Int. Society of Music Information Retrieval Conf.*, pp. 127–130, 2007.

[10] R. Middleton: "Form," in: *Continuum Encyclopedia of Popular Music of the World*, eds. J. Shepherd, D. Horn, D. Laing, Continuum Int. Publishing Group, 2003.

[11] E. Pampalk: "A Matlab Toolbox to Compute Similarity from Audio," *Proc. of the Int. Society for Music Information Retrieval Conf.*, 2004.

[12] J. Paulus and A. Klapuri: "Labelling the Structural Parts of a Music Piece with Markov Models," in: *Computer Music Modeling and Retrieval: Genesis of Meaning in Sound and Music*, Berlin: Springer, pp. 166-176, 2009.

[13] J. Paulus et al.: "Audio-based Music Structure Analysis," *Proc. of the 11th Int. Society for Music Information Retrieval Conf.*, pp. 625–636, 2010.

[14] J. Paulus: "Improving Markov Model-Based Music Piece Structure Labelling with Acoustic Information," *Proc. of the 11th Int. Society for Music Information Retrieval Conf.*, pp. 303–308, 2010.

[15] G. Peeters: *A Large Set of Audio Features for Sound Description in the CUIDADO Project*, Tech. Rep., IRCAM, Paris, France, 2004.

[16] R. Plomp and W. J. M. Levelt: "Tonal Consonance and Critical Bandwidth," *Journal of the Acoustical Society of America*, Vol. 38, No. 4, pp. 548–560, 1965.

[17] J. Salamon and E. Gomez: "Melody Extraction from Polyphonic Music Signals Using Pitch Contour Characteristics," *IEEE Trans. on Audio, Speech and Language Processing*, Vol. 20, No. 6, pp. 1759–70, 2010.

[18] J. Smith et al.: "Design and Creation of a Large-Scale Database of Structural Annotations," *Proc. of the Int. Society for Music Information Retrieval Conf.*, pp. 555–560, 2011.

VISUAL HUMDRUM-LIBRARY FOR PWGL

Mika Kuuskankare
DocMus
Sibelius Academy
mkuuskan@siba.fi

Craig Stuart Sapp
CCARH
Stanford University
craig@ccrma.stanford.edu

ABSTRACT

We introduce a PWGL Humdrum interface that integrates command-line unix tools for music analysis into a visual programming environment. This symbiosis allows users access to the strengths of each system—algorithmic composition and visual programming components of PWGL along with computational analysis and data processing features of Humdrum tools. Our novel interface for Humdrum graphical programming allows non-programmers better access to Humdrum analysis tools, particularly with the built-in music notation display capabilities of PWGL. ENP (Expressive Notation Package) data from PWGL can be exported as Humdrum data. Humdrum files in turn can be converted back into ENP data, allowing bi-directional communication between the two software systems.

1. INTRODUCTION

PWGL [10] is a visual music programming language written in Common Lisp, CLOS (Common Lisp Object System) and OpenGL. Its primary focus is on computer-assisted composition in an integrated environment with music notation, software synthesis and music theory and analysis tools. PWGL comprises several large-scale applications, such as ENP [5] for music notation, PWGLConstraints [7] for rule-based composition, and Kronos [11] for sound synthesis.

The Humdrum Toolkit is a widely used open-source software package for musicological research conceived of by David Huron [1] in the 1990's and has been developed and extended by others. It is composed of two main components: the Humdrum data format and the Humdrum Toolkit that processes music in this format. The Humdrum file format is a generalized data array. Each column represents a stream of data (such as a part in a musical score), while each row represents simultaneous events across multiple data streams. A particular advantage of this format is that it allows inclusion of analytic data streams alongside the original musical score within the same file. The primary music-content subformat in Humdrum is called *kern*, with about 50 other subformats predefined in the standard toolkit for encoding musical features such as lyrics, dynamics, tuning, harmonic analysis and perceptual data. the Humdrum file structure also allows users to define their own subformats for specific analytic or markup purposes.

The Humdrum Toolkit is a collection of command-line utilities operating on data written in the Humdrum syntax. In addition to the standard Humdrum utilities that are written in AWK, a software package called *Humdrum Extras* developed at CCARH at Stanford University extends and compliments the original toolset with additional command-line tools and a C++ code library for manipulating data in the Humdrum syntax with a focus on efficient numeric processing of the data. [1] Humdrum data processing can also done within PERL [2], and kern data can be imported into the music21 system implemented within Python. [2]

PWGL can be used to analyze scores written in the ENP format by using a built-in rule-based scripting language called ENP-script [4]. ENP offers both an extensive and extensible set of visualization devices that can be used to visualize analytic observations directly in the score [6]. It has already been used for music analysis, as reported for example in [9] and [8]; however, the current approach is not well suited for statistical analyses. Furthermore, a scant amount of musical data is readily available in the ENP format. The PWGL Humdrum interface now enables access to a vast number of pieces encoded in the kern data format that is made freely available for download from `http://kern.ccarh.org` [12]. In addition to direct translation of kern data into ENP, the interface allows access to a large number of predefined analytical tools, and allows for the batch analysis of a large musical corpus at once.

Our library attempts to make it easier to work within the Humdrum environment. First, it makes it possible for non-programmers to use and access Humdrum analysis tools through a visual interface. Many theorists, musicians, and composers may be intimidated by Humdrum's unix-flavored command-line syntax thus limiting the number of potential users. Second, for musicians it is important to be able to see the piece of music they are working on in common music notation. ENP allows an instant, editable and extendable representation of thousands of pieces of music encoded in kern notation. Finally, the analytic in-

[1] http://extras.humdrum.org and http://github.com/craigsapp/humextra
[2] http://mit.edu/music21

formation obtained via Humdrum tools can be visualized directly within an ENP score which assists interpretation of the results.

Previous attempts to create visual interfaces around Humdrum tools, such as JRing [3] have been made; however, the approach provided by PWGL is more general and user-extendable since PWGL is a generalized programming environment. The combination of Humdrum and PWGL should prove especially beneficial for musicologists, but should be of interest to MIR researchers and the like as well.

2. HUMDRUM LIBRARY

The PWGL Humdrum library is implemented in two parts: kern import is implemented as a Humdrum command-line tool called *hum2enp*, which is written in C++. The kern export and visual patch interface are implemented on the PWGL side and are written in Lisp and CLOS. The connection between the visual PWGL boxes and the Humdrum Tools is realized with the help of the PWGL Shell library, which provides for a graphical interface between PWGL and the unix command-line. Practically every unix command can be turned into a PWGL box and seamlessly used as a part of a patch along with the built-in boxes. When the Humdrum library is loaded, PWGL scans a specific directory defined by the user using the Humdrum library preferences for available Humdrum commands and creates the box interface automatically.

2.1 Kern import/export

Communication between PWGL and Humdrum tools is facilitated by two utilities that convert between ENP and Humdrum data. PWGL has a set of built-in Lisp methods that convert ENP scores into Humdrum data streams or files. Humdrum Extras reverses the process with a program called *hum2enp* to revert back to ENP data. These two converters allow for round-trip processing between ENP objects and Humdrum tools. For example, music generated algorithmically in PWGL can be exported into Humdrum data, then processed with Humdrum tools and converted back into ENP with some analytic markup.

The Humdrum and ENP representations of music are different in several ways. The Humdrum data paradigm is a spreadsheet. Each column (or more generally, a *spine* in Humdrum terminology) of data represents a stream of information such as a part, voice, dynamic, text or analytic data sequence. Each row in the data represents a simultaneity, meaning that all events in the row occur at the same time, with time progressing downwards in the file; any row above the current one occurs in its past, and rows below the current one occur in its future. In other words, horizontal lines of data can be thought of as being the total sounding harmony at any give point in time. This arrangement directly represents the configuration of notes in a musical score, but rotated 90° clockwise and without system breaks as shown in Figure 1. This also means that the lowest part in a file is the left-most spine in the data.

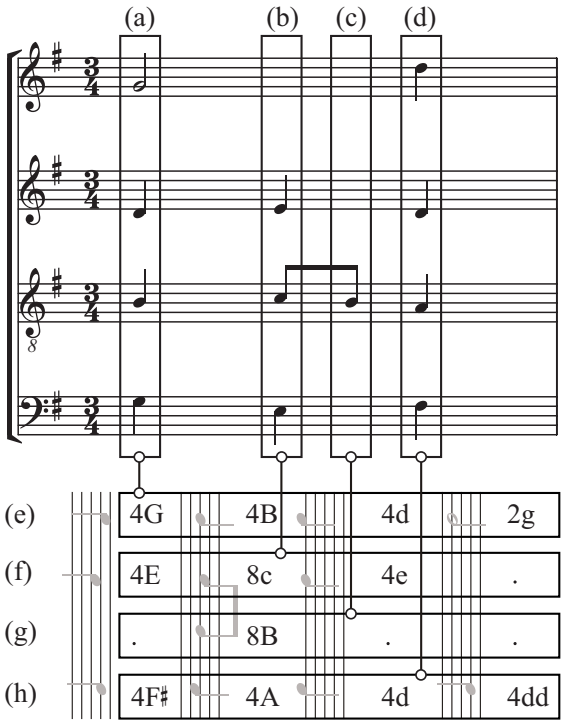
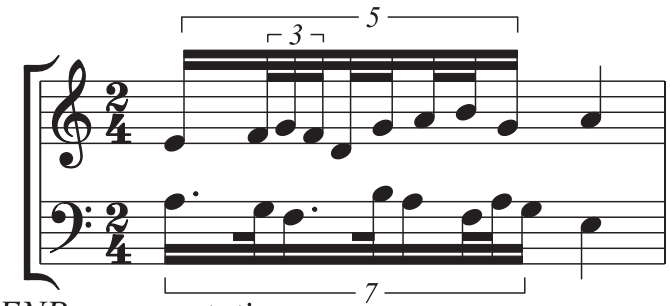


Figure 1. An ENP score displayed as graphical music notation (top) partitioned into harmonic slices (a–d) which are converted into kern records (e–h).

An ENP data file serializes the score by part from highest to lowest in the musical system with a configuration analogous to Type-1 Standard MIDI Files. While Humdrum data is arranged two-dimensionally, ENP data is stored in a hierarchical tree structure by part, voice, rhythm, chord, then note in a similar manner to most XML formats used for encoding musical data. Analytic data is typically stored as *expressions* on the chord or note level. Humdrum files, on the other hand, typically store analytic data in spines (columns) that are parallel to the main streams of musical data.

In order to convert from ENP scores into Humdrum, the scores are first partitioned into *harmonic slices* which effectively represent rows of Humdrum data (see Figure 1). A harmonic slice is created whenever there is at least one note object with a unique start time. Notes sharing start times are collected within the same harmonic slice using an identical process for harmonic analysis in PWGL, or to generate a notated score. Now the harmonic slices, which are already ordered by time, can be written as Humdrum data rows by reading each harmonic slice from bottom to top. Each harmonic slice thus becomes a Humdrum record where the lowest voice is encoded to the left-most column and the highest part in the right-most column. In general, every ENP voice becomes a kern spine within the Humdrum data. The Humdrum export is mainly intended for music analytic applications and therefore the complete ENP score structure is not necessarily preserved. In addition to rhythm and pitch, other notational properties exported from ENP to kern include instrument names, clefs,



ENP representation:

```
(((((1 ((1 :NOTES (64))
       (1 ((1 :NOTES (65)) (1 :NOTES (67)) (1 :NOTES (65))))
       (1 ((1 :NOTES (62)) (1 :NOTES (67))))
       (1 ((1 :NOTES (69)) (1 :NOTES (71))))
       (1 :NOTES (67))))
  (1 ((1 :NOTES (69)))))))
 (((((1
   ((2 ((3 :NOTES (57)) (1 :NOTES (55))))
    (2 ((3 :NOTES (53)) (1 :NOTES (59))))
    (3 ((1 :NOTES (57))
        (1 ((1 :NOTES (53))
            (1 :NOTES (57))))
        (1 :NOTES (55))))))
  (1 ((1 :NOTES (52)))))))
 :STAFF :BASS-STAFF))
```

Humdrum representation:

**kern	**kern
*clefF4	*clefG2
*M2/4	*M2/4
!LO:TUP:r=4:t=7	!LO:TUP:r=4:t=5
28.A	20e
!	!LO:TUP:r=20:t=3
.	60f
56G	.
.	60g
28.F	.
.	60f
.	40d
56B	40g
28A	.
.	40a
.	40b
56F	.
56A	.
.	20g
28G	.
4E	4a
=	=
*-	*-

Figure 2. Sample representation of complex rhythms in ENP and kern data formats.

tempos, time signatures, articulations, and optionally beam and tuplet groupings.

Once the musical data has been transformed into the Humdrum format, it can be processed with Humdrum tools and then converted back into ENP using the *hum2enp* program provided by Humdrum Extras. Humdrum data may contain separate spines for music and analytic data. As the *hum2enp* program converts the musical data into ENP, it collates analytic data into ENP expressions attached to notes or chords.

In order to correctly rebuild the rhythmic hierarchy in ENP, layout extensions have been created for kern data to demark tuplet groupings of arbitrary rhythmic complexity. Figure 2 shows ENP and Humdrum music encodings that represent the same rhythmic formations: a nested quintuplet and an irregularly subdivided septuplet. The structure of ENP is encoded within the parentheses levels starting with the outer pair for the score, then parts, voices, and measures. Rhythmic values are stored within measures as proportions in a hierarchical manner that allows nested tuplet rhythms as shown in Figure 2, where the triplet encompasses a duration of one quintuplet sixteenth note (1/3rd of 1/5th of a beat). The top level of the rhythmic structure is the beat, with subdivisions of the beat inserted as a list of proportions and notes which occur within the duration of their parent rhythmic proportion. Humdrum's linear representation of the music is more direct to the graphical music notation. Each part is a spine (column) of kern data with time progressing downwards. While ENP uses MIDI note numbers to encode pitch (with optional note attributes to resolve enharmonic spellings), kern data uses letter names for pitches. Upper case letters are for the octave below middle C, and lower case for the middle-C octave. Rhythmic values are numbers indicating the note count of that duration which are needed to create a whole note. For example, a septuplet sixteenth note is represented by the number 28, since 28 notes with this duration are needed to create a whole note. Dots following rhythmic values are augmentation dots, so "28." is a dotted septuplet sixteenth note, adding one-half of the dotless duration to the note. Dots in isolation are sequential alignment place holders indicating that the current part has no new activity while something new is occurring in the other part. The rhythmic value 60 can be interpreted as $4 \times 5 \times 3$ (a quarter note divided into 5, then each fifth divided into 3) or $4 \times 3 \times 5$ (a quarter note divided into 3, further divided into 5). The Humdrum representation in Figure 2 shows how to disambiguate between two such possible tuplet interpretations by using optional layout codes for tuplet groupings. The `r` parameter for tuplet layouts indicates the rhythmic duration of the tuplet, and the `t` parameter indicates the tuplet type. Thus the layout instruction `!LO:TUP:r=20:t=3` means that starting on the next note in the data stream (`60f`), a triplet (`t=3`) is to be applied with a total duration of a quintuplet sixteenth note (`r=20`).

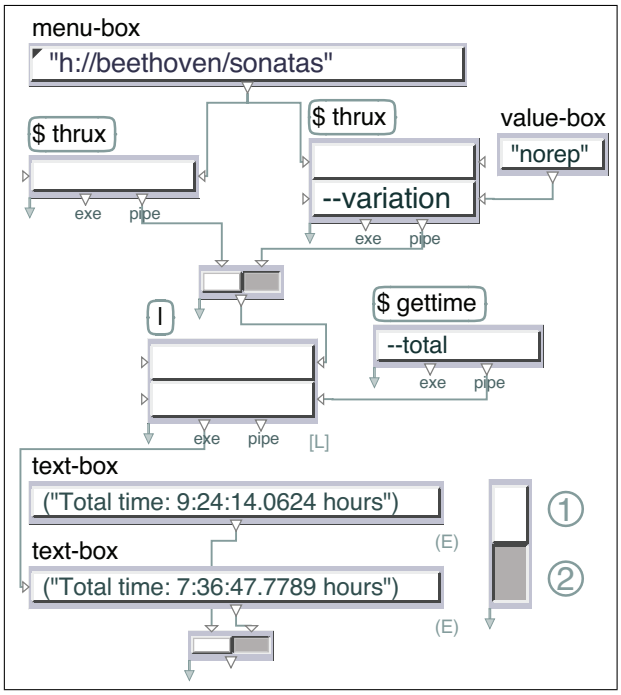


Figure 3. A PWGL Patch containing Humdrum boxes such as *thrux* that are distinguished from regular PWGL boxes by titles enclosed in rounded rectangles.

2.2 Interfacing to Humdrum tools

In addition to data musical transfer between the Humdrum and ENP representational systems, PWGL incorporates access to command-line Humdrum tools as mentioned in the introduction that allows processing of Humdrum data directly within PWGL. Figure 3 illustrates a PWGL patch that interfaces with two Humdrum Extras tools, *thrux* and *gettime*, to measure the total performance duration of a Humdrum data source (either a single work or a stream of multiple works or movements). The patch can be used to calculate the performance duration of input Humdrum data in two configurations: either the performance time when all repeats are taken, or the total amount of time when only second endings are taken. In this case the analysis shows (1) that the complete Beethoven piano sonatas would take about 9.5 hours to perform when taking all repeats (at the tempo specified within the data, taking no breaks between movements), or (2) about 7.5 hours without repeats. The master switch at the bottom right of the patch is used to choose between these two analysis methods to calculate the total performance time. Users of the patch can easily select a different repertory at the top of the patch, and the patch can be displayed in presentation mode to further simplify the interface (see Figures 4 & 5).

The menu-box object at the top of the patch stores a list of data sources, with the complete Beethoven piano sonatas currently selected. In this example the "h://" prefix indicates that the data will be downloaded dynamically from the Internet. [3] When the switch button in the patch

[3] specifically, the h://beethoven/sonatas URI maps onto the URL http://kern.ccarh.org/data?l=beethoven/sonatas

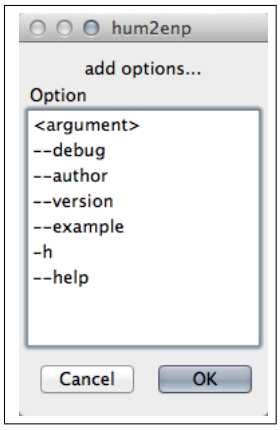


Figure 6. The options dialog for `hum2enp` command showing all the available flags.

causes data to flow through the left-hand *thrux* box, the input data stream is converted from printed ordering of the notation into performance ordering. The right-hand *thrux* box adds the `--variation norep` option to the command, which instructs *thrux* to take only second endings in the music to avoid repetitions. Output from both *thrux* commands is then piped to the *gettime* program. The `--total` option tells *gettime* to calculate the total duration of its input data. Without this option the program calculates the performance time of each harmonic slice. As a comparison, the patch can be expressed in unix shell syntax as two separate commands:

```
thrux h://beethoven/sonatas | gettime -T
thrux h://beethoven/sonatas -v norep | gettime -T
```

where `-v` is an alias for `--variation`, and `-T` for `--total`.

A more elaborate visualization of Humdrum analysis data can be seen in Figures 4 & 5 which display processed Humdrum data in two very different graphic formats on the patch. A menu-box at the top of the patch in Figure 4 is used to select a Humdrum file either from the local hard disk or downloaded via the web. The contents of this file can be pulled through two data paths in the patch. Humdrum data can be transformed into ENP data using the import-kern box and then displayed as graphical music notation within the Score-Editor box. A second path travels through the mkeyscape box which converts the Humdrum data into a graphic image. This image shows a keyscape plot [13] that summarizes the harmonic structure of the chorale, which is primarily in G major with transient key/chord regions on V, vi, and iii. Future work will implement a playback cursor in image boxes which link to the playback position when playing music in Score-Editor boxes. This will allow interactive visual feedback when playing music in the Score-Editor.

All Humdrum Extras command-line tools possess an option called `--options` that lists all options the program recognizes. The PWGL Humdrum library takes advantage of this self-documenting feature to extract a list of options that the program understands. When double-clicking on Humdrum boxes in a PWGL patch, a list of

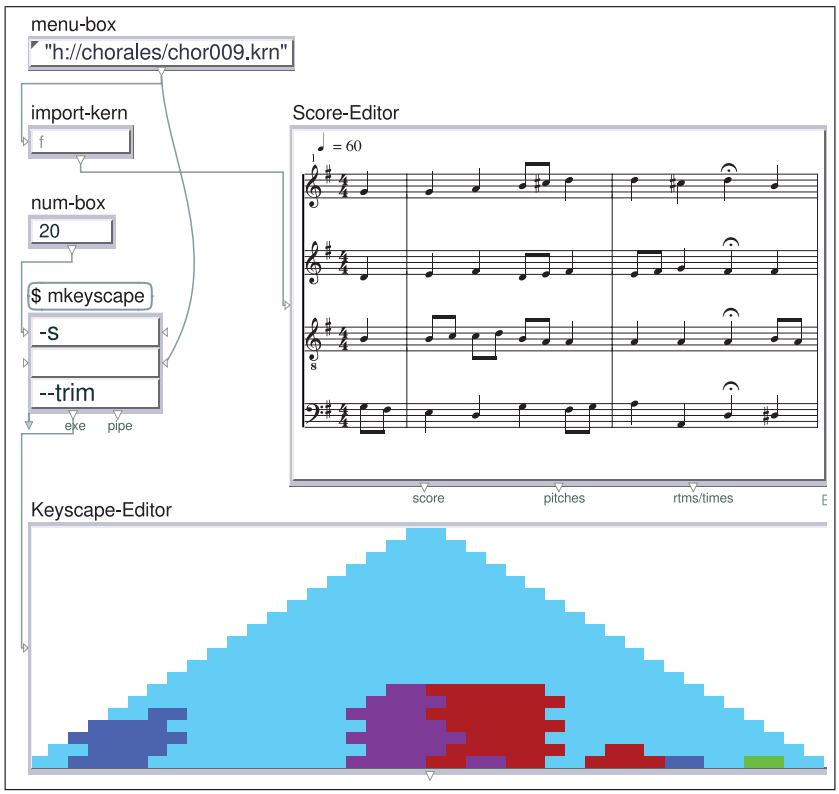


Figure 4. A PWGL patch which loads a Humdrum data file for a J.S. Bach chorale into the ENP Score-Editor as well as a visual analysis of the harmonic structure generated by the Humdrum Extras *mkeyscape* tool.

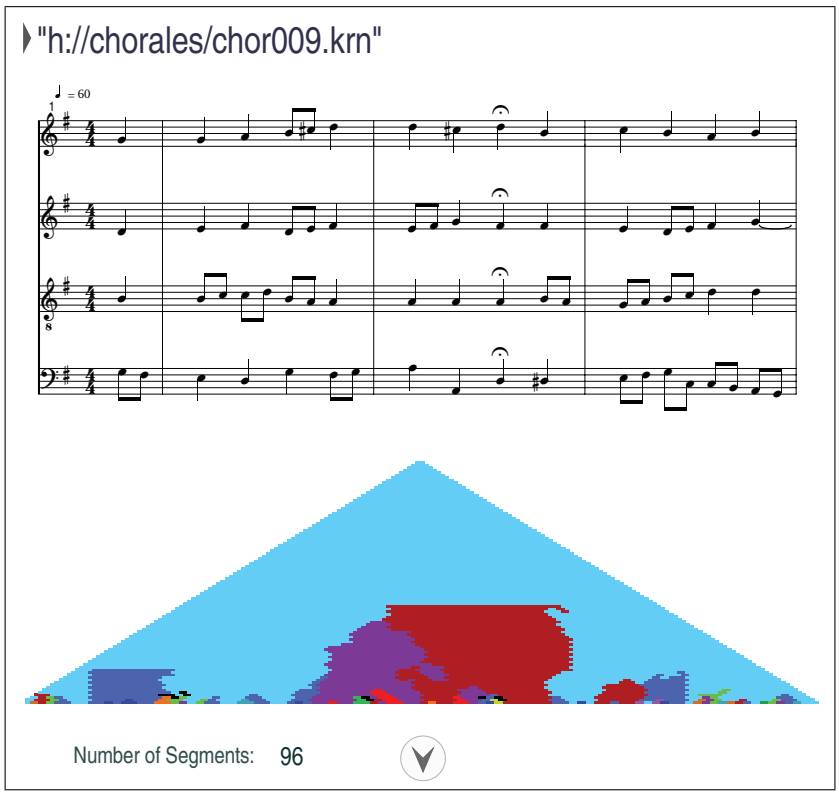


Figure 5. The patch shown in Figure 4 displayed in the presentation mode. The presentation mode allows us to hide the programming details from the end users. It also allows us to lay out the patch in a different way to create a more functional user-interface.

available command-line options appears, such as the list shown in Figure 6. Options can also be defined to accept strings or numbers as arguments, and any default value will also be displayed in the output of the program when using the `--options` option. In this example the `--help` option includes `-h` as a shorter alias.

Figure 5 shows the same patch as in Figure 4, but in *presentation mode*. The PWGL presentation mode is used to hide programming details in a patch to display it in a simplified form that focuses on inputs and outputs, suppressing intermediate details. In presentation mode the connections between boxes are hidden, and individual boxes can themselves be hidden, resized, displaced, as well as scaled or repositioned from their usual programming mode layout. The `-s` option for *mkeyscape*, which controls the number of analysis segments in the triangular image, is visible in Figure 4, taking a value of 20 from the attached num-box. In Figure 5 the call to the *mkeyscape* program is hidden, with only the number of analysis segments visible on the patch. In presentation mode the source score can be selected at the top of the patch, and the number of segments can be changed by typing a new value in place of 96. To recalculate the patch after changing the settings, the arrow at the bottom of the presentation patch is clicked. The program/presentation mode state is persistent for a patch so that it can be saved and then opened in presentation mode. The user can toggle back to the programming mode where visibility and size of all patch boxes are restored as illustrated in Figure 4.

3. CONCLUSIONS

PWGL's Humdrum library enables users of either PWGL or Humdrum to have access to tools available in the other system. For PWGL users the main benefit is access to thousands of classical music scores in the Humdrum file format, while Humdrum users have access to ENP's score editor that provides a rich palette of analytic markup for music notation. The PWGL/Humdrum interface allows users to access the full functionality of the Humdrum toolkit within PWGL and also allows for data exchange between the two systems via the Humdrum file format. The PWGL Humdrum library also offers enhancements over the standard Humdrum Toolkit by providing Humdrum users with a visual interface to Humdrum command-line tools. This is especially beneficial for non-programmers who usually feel more comfortable in a graphical programming environment. Since PWGL is composed of a generalized programming environment as well as the ENP notation editor, data obtained via Humdrum files/tools can be further processed using either pre-existing or user-defined tools within PWGL.

The PWGL Humdrum library can be downloaded at `http://www2.siba.fi/PWGL/downloads`. The `hum2enp` command-line tool is distributed as a part of Humdrum Extras and can be obtained from `http://github.com/craigsapp/humextra`.

4. ACKNOWLEDGMENTS

The work of Mika Kuuskankare has been supported by the Academy of Finland (SA137619). The authors would also like to thank both CCRMA and CCARH at Stanford University for hosting this research.

5. REFERENCES

[1] David Huron. Music information processing using the Humdrum Toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):15–30, 2002.

[2] Ian Knopke. The Perlhumdrum and Perllilypond toolkits for symbolic music information retrieval. In *International Symposium on Music Information Retrieval*, pages 147–152, 2008.

[3] Andreas Kornstädt. The JRing system for computer-assisted musicological analysis. In *ISMIR*, 2001.

[4] Mika Kuuskankare and Mikael Laurson. Intelligent Scripting in ENP using PWConstraints. In *Proceedings of International Computer Music Conference*, pages 684–687, Miami, USA, 2004.

[5] Mika Kuuskankare and Mikael Laurson. Expressive Notation Package. *Computer Music Journal*, 30(4):67–79, 2006.

[6] Mika Kuuskankare and Mikael Laurson. Survey of music analysis and visualization tools in PWGL. In *Proceedings of International Computer Music Conference*, pages 372–375, 2008.

[7] Mikael Laurson and Mika Kuuskankare. A constraint based approach to musical textures and instrumental writing. In *CP01 workshop on Musical Constraints*, Cyprus, 2001.

[8] Mikael Laurson, Mika Kuuskankare, and Kimmo Kuitunen. Introduction to computer-assited music analysis in PWGL. In *Sound and Music Computing '05*, 2005.

[9] Mikael Laurson, Mika Kuuskankare, and Kimmo Kuitunen. The Visualisation of Computer-assisted Music Analysis Information in PWGL. *Journal of New Music Research*, 37(1):61–76, 2008.

[10] Mikael Laurson, Mika Kuuskankare, and Vesa Norilo. An Overview of PWGL, a Visual Programming Environment for Music. *Computer Music Journal*, 33(1):19–31, 2009.

[11] Vesa Norilo. Introducing Kronos - A Novel Approach to Signal Processing Languages. In Frank Neumann and Victor Lazzarini, editors, *Proceedings of the Linux Audio Conference*, pages 9–16, Maynooth, Ireland, 2011. NUIM.

[12] Craig Stuart Sapp. Online database of scores in the Humdrum file format. In *Proceedings of ISMIR*, pages 664–665, 2005.

[13] Craig Stuart Sapp. *Computational Methods for the Analysis of Musical Structure*. Stanford University, 2011.

SOURCE SEPARATION OF POLYPHONIC MUSIC WITH INTERACTIVE USER-FEEDBACK ON A PIANO ROLL DISPLAY

Nicholas J. Bryan
CCRMA, Stanford University
njb@ccrma.stanford.edu

Gautham J. Mysore
Adobe Research
gmysore@adobe.com

Ge Wang
CCRMA, Stanford University
ge@ccrma.stanford.edu

ABSTRACT

The task of separating a single recording of a polyphonic instrument (e.g. piano, guitar, etc.) into distinctive pitch tracks is challenging. One promising class of methods to accomplish this task is based on non-negative matrix factorization (NMF). Such methods, however, are still far from perfect. Distinct pitches from a single instrument have similar timbre, similar note attacks, and contain overlapping harmonics that all make separation difficult. In an attempt to overcome these issues, we use a database of synthesized piano and guitar recordings to learn the harmonic structure of distinct pitches, perform NMF-based separation, and then extend the method to allow an end-user to interactively correct for errors in the output separation estimates by drawing on a piano roll display of the separated tracks. The user-annotations are mapped to linear grouping regularization parameters within a modified NMF-based algorithm and are then used to refine the separation estimates in an iterative manner. For evaluation, a prototype user-interface was built and used to separate several polyphonic guitar and piano recordings. Initial results show that the method of interactive feedback can significantly increase the separation quality and produce high-quality separation results.

1. INTRODUCTION

For many audio editing and production tasks, it is desirable to separate a single recording of a polyphonic instrument into its respective pitch tracks. One promising method to do so is that of non-negative matrix factorization (NMF), which models audio spectrogram data as a linear combination of prototypical frequency components or basis vectors over time. NMF can be defined by

$$\mathbf{V} \approx \mathbf{W}\,\mathbf{H} \tag{1}$$

where $\mathbf{V} \in \mathrm{R}_{+}^{F \times T}$ is an audio spectrogram, $\mathbf{W} \in \mathrm{R}_{+}^{F \times K}$ is a dictionary or matrix of basis vectors (columns), and $\mathbf{H} \in \mathrm{R}_{+}^{K \times T}$ is a matrix of activations or gain vectors (rows).

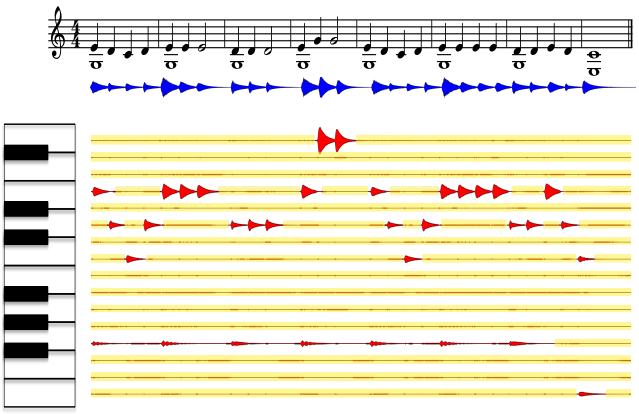


Figure 1: Polyphonic source separation of a single piano recording (blue) of *Mary Had A Little Lamb* displayed on a piano roll. Given an initial separation, a user can annotate errors (yellow overlays) in the separated outputs (red) and iteratively improve results. Note underneath the overlays, incorrect, residual energy is present.

Given a spectrogram $\mathbf{V}$, the matrices $\mathbf{W}$ and/or $\mathbf{H}$ can then be computed via an optimization problem

$$\begin{aligned} \underset{\mathbf{W},\mathbf{H}}{\text{minimize}} \quad & D(\mathbf{V} \,|\, \mathbf{W}\,\mathbf{H}) \\ \text{subject to} \quad & \mathbf{W} \geq 0, \mathbf{H} \geq 0 \end{aligned} \tag{2}$$

that minimizes the distance between $\mathbf{V}$ and $\mathbf{W}\,\mathbf{H}$, where D is a suitable divergence function (e.g. Euclidean, Kullback-Leibler, Itakura-Saito, etc.) and the inequalities are element-wise [5, 6, 10, 11, 14]. Note, (2) is non-convex, typically allowing us to only find a local optima.

When used to separate a single polyphonic recording into distinctive pitch tracks (e.g. 88 keys of the piano), typically supervised NMF is used. In this case, isolated recordings of distinct pitches are used to model the prototypical frequency content of each pitch. The pre-learned models are then collected together and used to estimate the contribution of each pitch within an unknown mixture.

These methods can sometimes produce high-quality separation results. At the same time, however, these methods can be frustrating in that the resulting separation output estimates can contain errors that are audibly and/or visually obvious. These errors are typically caused by overlapping harmonics, similarities in timbre, similarities in note attack, and other such issues, limiting the general usefulness of the method.

To overcome these limitations, we propose an extension to supervised NMF-based source separation. In particular, we allow an end-user to separate a single recording into distinctive pitch tracks and then interactively annotate errors in the output estimates by drawing on a piano roll display of the separated tracks, as shown in Fig. 1. The user-annotations are then mapped to linear grouping regularization parameters in a modified NMF-based algorithm to refine the separation estimates and iteratively improve results through user-feedback.

To initially train our supervised NMF model, we leverage a database of synthesized pianos and guitars to learn the harmonic structure of distinct pitches. Using the entire database, we learn a universal pitch model across all instruments and timbres. Additionally, we learn instrument-specific pitch models, and instrument/timbre-specific models that can be used in place of the universal model if needed. For evaluation, we built a prototype user-interface and used it to separate several polyphonic guitar and piano recordings. Initial results shows that the proposed method significantly improves separation quality and can produce high-quality separation estimates.

The complete proposed method consists of an initial pre-computation step discussed in Section 2, and two core steps in Section 3, and Section 4. Algorithmic issues, evaluation, and related works are discussed in Section 5, Section 6, and Section 7, followed by acknowledgements, and conclusions in Section 9, and Section 8.

2. LEARNING PITCH MODELS

To separate an unknown polyphonic instrument recording, we must first precompute or learn the prototypical frequency content for each pitch $p \in 1, ..., P$ we wish to separate. For a given pitch p, we learn one or more (K_p) prototypical spectra or basis vectors that capture the harmonic structure of that pitch. While this can be achieved by handcrafting specific spectra, we instead learn this from data by the following supervised NMF procedure:

1. Given isolated training data of each pitch p, compute the spectrogram $\mathbf{V}_p, \ \forall \ p \in 1, ..., P$ via the short-time Fourier Transform (STFT).

2. Factorize each spectrogram $\mathbf{V}_p$ via (2), and obtain the basis vectors $\mathbf{W}_p \in \mathrm{R}_+^{F \times K_p}$ of each source, where K_p is the number of basis vectors per pitch. Normalize each column of $\mathbf{W}_p$ to sum to one. Discard the activations $\mathbf{H}_p \in \mathrm{R}^{K_p \times T}$.

3. Concatenate the basis vectors $\mathbf{W}_p$ together to form the complete pitch model or dictionary $\mathbf{W} = [\mathbf{W}_1 \, \mathbf{W}_2 \ldots \mathbf{W}_P] \in \mathrm{R}_+^{F \times K}$, where $K = \sum_1^P K_p$.

For this work, we define D to be the Kullback-Leibler (KL) divergence and use the multiplicative NMF-update algorithm of Lee and Seung [6] to solve (2). Note, alternative divergence functions can be used instead, such as the Itakura-Saito divergence. An example set of guitar basis vectors for one octave ($K_p = 1$) is shown in Fig. 2.

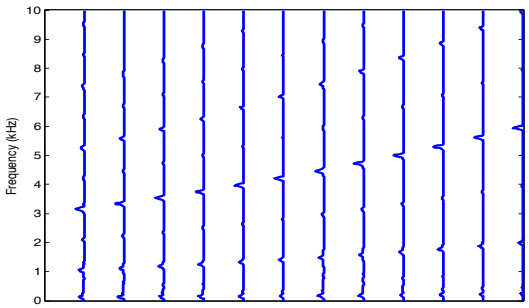


Figure 2: One octave of acoustic guitar pitch-based basis vectors for $K_p = 1$. Notice the harmonic structure.

For training data, we used 13 distinct piano synthesizers and 11 distinct guitar synthesizers and recorded two measures of quarter notes (≈ 5 seconds) for each of the 88 piano pitches of each of the 24 synthesizers. The different synthesizers have contrasting timbres, reverberation, and other effects and correspond to the guitar and piano presets of the Logic Pro software package. The piano timbres include: electric piano, grand piano, grand piano on stage, grand piano punchy, smokey clav, smooth clav, studio pop piano, swirling electric piano, whirly, yamaha piano club, yamaha piano hall, yamaha piano room, yamaha piano studio. The guitar timbres include: acoustic guitar, big electric lead, classical acoustic, clean electric, crunchy muted delays, electric tremolo, fuzzy synth guitar, heavy metal guitar, los freakos, nylon shimmer, steel string acoustic.

Given the collection of recordings, we then use various subsets of the data and the aforementioned supervised NMF procedure several times to compute various pitch models. By learning a pitch model on the complete set of recordings across all instruments and timbres, for example, we learn a form of universal pitch model (U) that generalizes across instruments and instrument timbre, similar in motivation to the work of Reynolds et al. [9] and more recently Sun and Mysore [12]. When using such a model to perform separation, however, the results might be less than ideal because of the difference in harmonic structure between instruments (e.g. guitar vs. piano). This motivates the ability to train more specific pitch models on subsets of our training data. As a result, in addition to computing a general, universal pitch model, we additionally compute a universal guitar model (UG), a universal piano model (UP), and all 24 instrument/timbre-specific models (T).

3. MIXTURE SEPARATION

Given a particular pitch model, we can proceed to separate an unknown polyphonic mixture sound. This involves using a single, complete pitch model $\mathbf{W}$ to estimate the weights or activations $\mathbf{H}$ of each pitch from the unknown recording spectrogram $\mathbf{V}$. This is done via

$$\begin{aligned} \underset{\mathbf{H}}{\text{minimize}} \quad & D(\mathbf{V} \,|\, \mathbf{W}\,\mathbf{H}) \\ \text{subject to} \quad & \mathbf{H} \geq 0 \end{aligned} \tag{3}$$

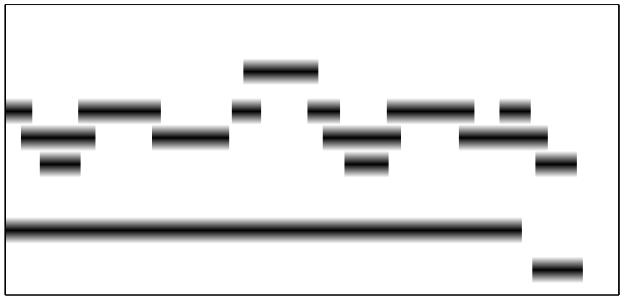


Figure 3: User-annotated penalty weight parameters $\mathbf{\Lambda}$ from Fig. 1, depicted as an image. White represents annotated regions. Black represents unannotated regions.

where we only optimize over $\mathbf{H} \in \mathrm{R}_{+}^{K \times T}$. Assuming D is convex, this optimization problem is convex, allowing us to find a global minimizer. We again use the KL divergence (which is convex) and the multiplicative update algorithm of [6] to solve (3), holding $\mathbf{W}$ fixed. We then use the given pitch model and corresponding activations to estimate the magnitude spectrogram of each pitch within the mixture. The estimated pitch spectrograms are then converted to the time-domain using the mixture phase and inverse STFT according to standard practice (see Section 5).

4. INTERACTIVE USER-FEEDBACK

Once an initial separation is performed, we allow an end-user to interactively refine the separated output estimates by annotating a piano roll display of the results. To do so, we 1) instruct an end-user to draw on regions of each pitch track that are incorrectly separated, 2) incorporate the annotations to update the separation estimates, 3) present the updated results back to the user, and 4) repeat until satisfied. This form of interaction is done as a result of the observation that it is much easier for people to iteratively correct for errors after an initial result is presented, rather then pre-annotate time regions of one source or another. This is similar to the observations discussed in [3], where user-feedback is used to improve a clustering algorithm.

The specific type of drawing interaction can be done in several ways, such as 1) annotating a type of amplitude envelope for each pitch track, where height is used as a measure of confidence/strength of the error annotation or 2) allowing a user to paint over the errors with a colored brush, where opacity is used as a measure of confidence/strength.

The drawing annotations are then collected into a single matrix $\mathbf{\Lambda} \in \mathrm{R}^{P \times T}$, where each row corresponds to the penalties for a given pitch. This matrix is then used to penalize the activations of the incorrectly activated pitches from the initial separation estimates. As a result, we can remove errors caused by incorrectly activated notes and reallocate the incorrectly assigned energy to the remaining pitch tracks in an optimal way. This is in contrast to directly using the annotations to down-weight the appropriate elements of $\mathbf{H}$ without recomputing the factorization, which would not reassign the incorrectly allocated energy.

An example penalty matrix $\mathbf{\Lambda}$ is shown in Fig. 3, which

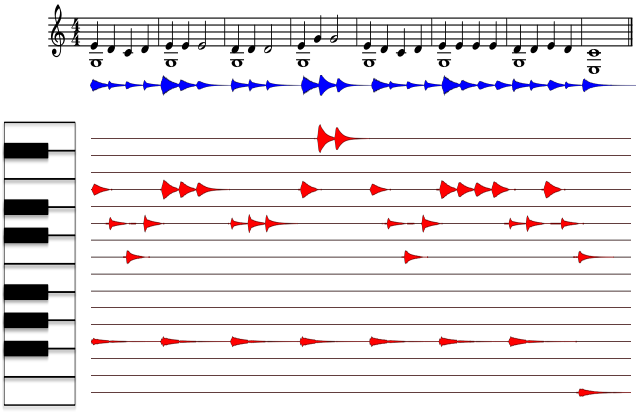


Figure 4: Refined polyphonic separation of *Mary Had A Little Lamb* with user-guided interactive feedback. Notice incorrectly activated notes are eliminated and the note attacks are more sharply outlined.

embodies the user annotations, shown as yellow overlays, in Fig. 1. Note that while Fig. 3 appears binary in nature, $\mathbf{\Lambda}$ is real-valued (the benefit of real-valued annotations is discussed below).

The user-annotation matrix $\mathbf{\Lambda}$ is then incorporated into our NMF model via

$$\begin{aligned} \underset{\mathbf{H}}{\text{minimize}} \quad & D(\mathbf{V} \,|\, \mathbf{W}\,\mathbf{H}) + \lambda\, \Omega(\mathbf{H}; \mathbf{\Lambda}) \\ \text{subject to} \quad & \mathbf{H} \geq 0 \end{aligned} \tag{4}$$

$\Omega(\mathbf{H}; \mathbf{\Lambda})$ is an appropriately chosen penalty that is a function of $\mathbf{H}$ and parameterized by $\mathbf{\Lambda}$ and $\lambda \in \mathrm{R}_{+}$ is a scalar used to decrease or increase the overall weight of the user annotation penalty. The penalty function $\Omega(\mathbf{H}; \mathbf{\Lambda})$ discourages the activations of specific pitches, dependent on the user-annotations. For our case, we use a simple linear group penalty

$$\Omega(\mathbf{H}; \mathbf{\Lambda}) = \sum_{t=1}^{T} \sum_{p=1}^{P} \sum_{k=(p-1)K_p+1}^{pK_p} \mathbf{\Lambda}_{(p,t)}\, \mathbf{H}_{(k,t)} \tag{5}$$

$$= \mathbf{1}_K{}^{\mathrm{T}} (\mathbf{\Gamma}\, \mathbf{\Lambda} \odot \mathbf{H})\, \mathbf{1}_T \tag{6}$$

where the matrix subscripts are used to index the rows or columns of the given matrices, $\mathbf{\Gamma} \in \mathrm{R}^{K \times P}$ is

$$\mathbf{\Gamma} = \begin{bmatrix} \mathbf{1}_{Kp} & 0 & \dots & 0 \\ 0 & \mathbf{1}_{Kp} & \dots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & \mathbf{1}_{Kp} \end{bmatrix}, \tag{7}$$

$K = \sum_{1}^{P} K_p$, and $\mathbf{1}_N \in \mathrm{R}^{N \times 1}$ is a column vector of N ones. While alternative penalties are possible, (6) is relatively straightforward, adds minimal computation complexity, and results in a compact multiplicative update algorithm for solving (4) as discussed below in Section 5. Also note, due to linearity, we can absorb λ into the user annotation matrix $\mathbf{\Lambda}$ and only use $\mathbf{\Lambda}$ for user-tuning.

We can see the immediate benefit of the interactive user-feedback in two demonstrative examples. First, Fig. 4 illustrates the output result of including the user-annotations

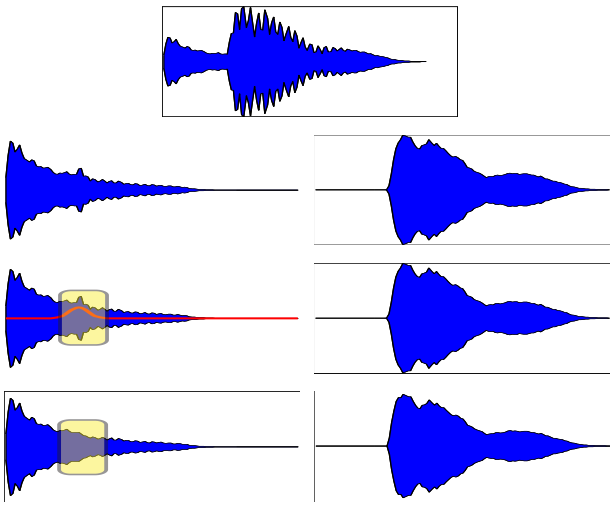


Figure 5: (First Row) Mixture spectrogram of E and D piano pitches. (Second Row) Initially separated results (E-left, D-right) using supervised NMF. (Third Row) Initially separated results (E-left, D-right) using supervised NMF with overlaid annotations (red-line) depicting incorrectly separated note transient (yellow box). (Third Row) The refined separation estimates with the transient error removed.

displayed in Fig. 1, which are used to clean up incorrectly separated regions of each pitch track. We can notice that both the incorrect note activations are removed and the note transients of the remaining (correct) notes are sharpened as a result of reallocating the incorrectly assigned note energy to the remaining pitch tracks.

Second, we can view waveform visualizations in Fig. 5, which show the benefit of having the ability to annotate a level of confidence or strength (real-valued annotations) when correcting for errors. In this example, we separate two overlapping piano pitches (E and D). Using standard, supervised NMF-based separation, the energy from the transient attack of the second note gets incorrectly assigned to the first note, causing a ghost-like effect. After annotation (red line), the transient error is reduced and the separation quality is improved.

5. ALGORITHM

Using a suitably defined divergence function $D(\mathbf{V} \,|\, \mathbf{W}\,\mathbf{H})$, our choice of $\Omega(\mathbf{H})$, and an appropriate pitch model $\mathbf{W}$, we need to derive an efficient algorithm to actually compute the unknown activations $\mathbf{H}$ for each pitch. As before, we define D to be the Kullback-Leibler divergence and follow the mathematical justification of Lee [6] to derive a Majorization-Minimization optimization algorithm to solve (4), resulting in a multiplicative update algorithm that incorporates our user-guided constraints.

Given the modified multiplicative NMF update equations, we outline the complete interactive separation algorithm in Algorithm 1. We define the forward and inverse short-time Fourier transform as $(\mathbf{V},\ \angle\,\mathbf{V}) \leftarrow \text{STFT}(\mathbf{x})$ and $\mathbf{x} \leftarrow \text{ISTFT}(\mathbf{V},\ \angle\,\mathbf{V})$, $\mathbf{1}$ to be an appropriately sized matrix of ones, $\odot$ is element-wise multiplication, the division

Algorithm 1 Interactive Polyphonic Separation

Procedure INTERACTIVE-POLY-SEPARATION (
$\mathbf{x}$, // time-domain mixture signal
$\mathbf{W}$, // pitch basis vectors (model)
K_p, // basis vectors per pitch
)
initialize: $\mathbf{\Lambda} = \mathbf{0}$
precompute:
$(\mathbf{V}, \angle\,\mathbf{V}) \leftarrow \text{STFT}(\mathbf{x})$
repeat
  input: user-annotated penalties $\mathbf{\Lambda} \in \mathbf{R}^{P \times T}$
  initialize: feasible $\mathbf{H} \in \mathbf{R}_{+}^{K \times T}$
  repeat

$$\mathbf{H} \quad \leftarrow \quad \mathbf{H} \odot \frac{\mathbf{W}^{\mathrm{T}}(\frac{\mathbf{V}}{\mathbf{W}\,\mathbf{H}})}{\mathbf{W}^{\mathrm{T}}\,\mathbf{1} + \mathbf{\Gamma}\,\mathbf{\Lambda}} \tag{8}$$

  until convergence
  for all $p \in 1, \ldots, P$ **do**

$$\hat{\mathbf{V}}_p \leftarrow \mathbf{V} \odot \frac{\mathbf{W}_{(p)}\,\mathbf{H}_{(p)}}{\mathbf{W}\,\mathbf{H}} \tag{9}$$

$$\mathbf{x}_p \leftarrow \text{ISTFT}(\hat{\mathbf{V}}_p, \angle\,\mathbf{V}) \tag{10}$$

  end for
until satisfied
return: time-domain signals $\mathbf{x}_p,\ \forall\ p \in \{1, ..., P\}$

is element-wise, and use the subscript notation (p) to pick off the elements of $\mathbf{W}$ and/or $\mathbf{H}$ that correspond to pitch p. At each point within the feedback-loop, the entire NMF-based separation is re-run from scratch, displayed to the user, and used as a starting point for further iterations.

6. RELATED WORK

There are several related works that leverage some form of user-guidance to aid the source separation process. One of the most similar works to our proposed approach is discussed in Ozerov et al. [8]. In this work, segmental information indicating the time activations of particular sources is used in a multichannel nonnegative tensor factorization model to improve separation quality. While similar to our proposed work, this work only allows for binary annotations that are used to zero-initialize elements of $\mathbf{H}$ and does not allow a user to specify a confidence or strength level. As a result, there is no mechanism to guide the separation process within regions where two or more sources overlap, such as our example of Fig. 5.

Other user-guided approaches include the work of Durrieu et al. [4], Lefèvre et al. [7], and Bryan and Mysore [1, 2], which each use some form of time-frequency display to elicit user-annotations. In all such cases, however, the interaction process is limited to separating two sound sources at a time (as oppose to P pitches). In addition, these works require end-users to annotate time-frequency displays of sound, which can be difficult to interpret even for expert users, motivating the proposed approach.

Table 1: Piano results (in dB) averaged across songs, timbre, and active notes.

$K_p = 1$	T	T+	UP	UP+	U	U+
SDR	12.5	13.2	9.4	11.5	9.8	11.6
SIR	19.7	22.9	15.6	21.0	15.9	21.1
SAR	16.2	16.0	15.5	14.5	15.6	14.5
$K_p = 5$	T	T+	UP	UP+	U	U+
SDR	13.1	13.3	10.5	12.8	10.1	12.6
SIR	19.0	23.0	14.4	22.3	14.2	22.3
SAR	17.0	16.0	17.2	15.5	17.0	15.4

Table 2: Guitar results (in dB) averaged across songs, timbre, and active notes.

$K_p = 1$	T	T+	UG	UG+	U	U+
SDR	12.7	12.5	8.2	10.4	7.8	10.2
SIR	19.2	22.2	15.0	20.3	14.5	20.1
SAR	16.2	15.3	15.0	13.7	15.4	13.5
$K_p = 5$	T	T+	UG	UG+	U	U+
SDR	12.8	12.6	9.6	11.5	8.7	11.2
SIR	18.2	22.4	13.5	21.1	12.3	20.4
SAR	17.1	15.3	16.6	14.6	16.6	14.3

Table 3: Piano results (in dB) averaged across songs, timbre, and inactive notes.

$K_p = 1$	T	T+	UP	UP+	U	U+
SDR	-6.5	133.9	-12.3	134.5	-2.3	135.2
SIR	-79.2	61.5	-84.6	61.7	-74.2	62.1
SAR	4.9	90.0	1.8	90.3	7.3	90.7
$K_p = 5$	T	T+	UP	UP+	U	U+
SDR	-5.7	135.1	-11.7	135.0	-10.9	134.7
SIR	-79.6	62.3	-86.3	62.0	-86.2	61.9
SAR	4.8	90.7	0.8	90.6	0.8	90.4

Table 4: Guitar results (in dB) averaged across songs, timbre, and inactive notes.

$K_p = 1$	T	T+	UG	UG+	U	U+
SDR	1.2	133.9	-7.4	135.5	-11.1	135.3
SIR	-84.0	59.4	-91.1	60.6	-94.6	59.9
SAR	7.7	91.4	4.5	92.3	2.5	92.1
$K_p = 5$	T	T+	UG	UG+	U	U+
SDR	2.7	135.1	-13.0	134.7	-13.4	134.7
SIR	-83.9	60.1	-99.2	59.8	-99.4	59.6
SAR	7.7	92.1	0.0	91.8	0.0	91.8

7. EVALUATION

We built a C++ prototype user-interface similar to Fig. 1. We then used the interface to test our proposed method on several polyphonic piano and guitar recordings with various pitch models. We generated the test material from five short MIDI files, including "Mary Had A Little Lamb" by Sarah Josepha Hale/Lowell Mason, "The Blue Danube" by Johann Strauss, "Super Mario Bros" by Nintendo Games, "Yesterday" by The Beatles, and "Maple Leaf Rag" by Scott Joplin, using the 24 different synthesizers discussed in Section 2, resulting in 120 different ground truth recordings. For each of the five unique songs, the user-interface was used to initially separate each song into pitch tracks and then interactively refine the outputs over of the course of 30 minutes. The five user-annotations were then saved and used to test the method across the different ground-truth recordings and pitch models.

The BSS-EVAL metrics were then used to compute the Signal-to-Distortion ratio (SDR), Signal-to-Interference ratio (SIR), and Signal-to-Artifact ratio (SAR) to measure the separation quality [13]. The SIR measures the level of suppression of the unwanted pitch sources, the SAR measures the level of artifacts introduced by the separation process, and the SDR gives an average measure of separation quality that considers both the suppression of the unwanted sources and level of artifacts compared to ground truth.

The results were computed for each instrument before and after user-interaction, averaged across song and timbre, for various pitch models and values of K_p. When evaluating the simultaneous separation of 88 different sound sources, however, the standard approach of comparing all combinations of the estimated sources and known sources becomes computationally prohibitive. As a result, we take the approach of reducing the problem into 88 two-source evaluations that compare the separation quality of each individual pitch p vs. the remaining 87 pitches. In addition, we partition the results for active and inactive pitches (completely zero signals), allowing for a more detailed and careful analysis of the results shown below. Without this partitioning, the results are extremely skewed in favor of the proposed method due to averaging the inactive signal results, limiting interpretability.

The results for active pitches are shown in Table 1 for piano and Table 2 for guitar. The results for inactive pitches are shown in Table 3 for piano and Table 4 for guitar. The different pitch models include: universal (U), universal guitar (UG), universal piano (UP), and each of the 24 instrument/timbre-specific models (T). Items denoted with a plus (+) indicate user-interaction was used. Note, because of the partitioning of active vs. inactive pitches, comparison of SDR vs. SAR vs. SIR should only be done within like rows.

From these results, we have two initial observations. First, we compare the results with and without interaction. For active pitches, the SDR and SIR improve by several decibels for almost all pitch models and values of K_p. When using the timbre-based pitch model (T) for a given set of recordings, however, there are cases where user-interaction (T+) slightly decreases the SDR for active pitches. For inactive pitches, the SDR, SAR, and SIR all improve by an extremely large amount (often more than 100 dB) caused by the zero signal pitch tracks in both the estimated and true recordings, motivating our decision to separate the results for inactive and active pitch sets. This shows us that 1) a vast majority of the SDR increase due to user-interaction is caused by simply annotating inactive pitch tracks, 2) while the user-annotations can occasionally decrease the SDR for active pitches, in most cases, user-interaction increases the SDR results for both active and inactive pitches. This is interesting in that it demonstrates the idea that annotating incorrectly activated notes can improve the separation quality of all other pitches on average.

Secondly, we can compare the results between the uni-

versal pitch model (U), the instrument-specific pitch models (UG, UP), and the instrument/timbre-specific pitch models (T). We can see that the instrument/timbre-specific pitch models (T) perform the best, followed by the instrument-specific pitch models (UG, UP), and then the universal pitch model (U) as expected. When we vary the value of K_p and incorporate user-interaction (+), however, this effect is significantly reduced or eliminated. In the case of piano, the universal model with interaction outperformed the timbre-based pitch model without interaction. This is significant in that it gives hope to the use of "universal" pitch models, which eliminates the need for specific training data for particular instruments and timbres.

Finally, because it is difficult to evaluate the sound quality of the proposed method via numerical comparison, audio and video examples of our prototype can be found at `ccrma.stanford.edu/~njb/research/pitch`.

8. CONCLUSIONS

In an attempt to overcome common, frustrating, and limiting problems in supervised non-negative matrix factorization approaches to polyphonic single-channel source separation, we propose an extension that allows a user to correct for errors (with a confidence value) in the separation results by annotating a piano roll visualization of sound. The user-annotations are mapped to linear grouping regularization parameters within a modified NMF-based algorithm, and used to refine the separation estimates and improve results. In addition, a database of piano and guitar recordings was used to learn a generalized pitch model, instrument-specific pitch models, and instrument/timbre-specific models. A prototype user-interface was built and used to separate several polyphonic guitar and piano recordings and initial results show that 1) user-interaction can significantly increase separation quality and 2) make the use of generalized universal pitch models more viable.

9. ACKNOWLEDGEMENTS

This work was generously supported by Adobe Research.

10. REFERENCES

[1] N. J. Bryan and G. J. Mysore. An efficient posterior regularized latent variable model for interactive sound source separation. In *International Conference on Machine Learning*, June 2013.

[2] N. J. Bryan and G. J. Mysore. Interactive refinement of supervised and semi-supervised sound source separation estimates. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 2013.

[3] D. Cohn, R. Caruana, and A. Mccallum. Semi-supervised clustering with user feedback. In Sugato Basu, Ian Davidson, and Kiri Wagstaff, editors, *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Chapman & Hall/CRC, 2008.

[4] J.-L. Durrieu and J.-P. Thiran. Musical audio source separation based on user-selected f0 track. In *The 10th International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, pages 438–445, 2012.

[5] C. Févotte and J. Idier. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural Computation*, 23(9):2421–2456, 2011.

[6] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 556–562. MIT Press, 2001.

[7] A. Lefèvre, F. Bach, and C. Févotte. Semi-supervised nmf with time-frequency annotations for single-channel source separation. In *In the Proceedings of The International Society for Music Information Retrieval (ISMIR) Conference*, 2012.

[8] A. Ozerov, C. Févotte, R. Blouet, and J.-L. Durrieu. Multichannel nonnegative tensor factorization with structured constraints for user-guided audio source separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 257 –260, May 2011.

[9] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted gaussian mixture models. In *Digital Signal Processing*, page 2000, 2000.

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

[11] P. Smaragdis, B. Raj, and M. Shashanka. Supervised and semi-supervised separation of sounds from single-channel mixtures. In *International Conference on Independent Component Analysis and Signal Separation*, pages 414–421, Berlin, Heidelberg, 2007. Springer-Verlag.

[12] D. L. Sun and G. J. Mysore. Universal speech models for speaker independent single channel source separation. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

[13] E. Vincent, R. Gribonval, and C. Fevotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462 –1469, July 2006.

[14] T. Virtanen. Monaural Sound Source Separation by Nonnegative Matrix Factorization With Temporal Continuity and Sparseness Criteria. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 15(3):1066–1074, March 2007.

OPTICAL MEASURE RECOGNITION IN COMMON MUSIC NOTATION

Gabriel Vigliensoni, Gregory Burlet, and Ichiro Fujinaga
Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
McGill University, Montréal, Québec, Canada
{gabriel,ich}@music.mcgill.ca, gregory.burlet@mail.mcgill.ca

ABSTRACT

This paper presents work on the automatic recognition of measures in common Western music notation scores using optical music recognition techniques. It is important to extract the bounding boxes of measures within a music score to facilitate some methods of multimodal navigation of music catalogues. We present an image processing algorithm that extracts the position of barlines on an input music score in order to deduce the number and position of measures on the page. An open-source implementation of this algorithm is made publicly available. In addition, we have created a ground-truth dataset of 100 images of music scores with manually annotated measures. We conducted several experiments using different combinations of values for two critical parameters to evaluate our measure recognition algorithm. Our algorithm obtained an f-score of 91 percent with the optimal set of parameters. Although our implementation obtained results similar to previous approaches, the scope and size of the evaluation dataset is significantly larger.

1. INTRODUCTION

Optical music recognition (OMR) is the process of converting scanned images of pages of music into computer readable and manipulable symbols using a variety of image processing techniques. Thus, OMR is seen as a valuable tool that helps accelerate the creation of large collections of searchable music books.

However, the automatic recognition of printed music presents several substantial challenges, including: A large variability in the quality of analog or digital sources; the common superimposition of shapes within a score on staves, making it difficult for computers to isolate musical elements and extract musical features that represent the content; and finally, a large number of music symbols that can create a large pattern space [5]. Also, as noted in [7], a common source of OMR errors originate from the misinterpretation of note stems or other vertical structures in the score as barlines, or vice-versa, which leads to measure annotations with false positives or false negatives.

In this project we aim to create an optical measure recognition algorithm capable of recognizing the physical location of barlines in a wide range of scores of common Western music notation (CWMN), deriving the bounding boxes for the measures, and storing these elements in a symbolic music file format. This allows us to relate the physical location on the page to the structure of the music itself. Applications that employ optical measure recognition can enhance interactions with music scores, and include multimodal music presentation and navigation, such as synchronizing digitized scores with audio playback [8], or content-based retrieval systems, which allow users to query the score by its measures [9]. In these applications, the correct extraction of barlines is essential for a proper alignment of the different music representations.

The structure of this paper is as follows: Section 2 presents the structure and implementation details of the optical measure recognition algorithm we have developed. Section 3 describes the annotation methodology and design of the ground-truth dataset that is used to evaluate the optical measure recognition algorithm. The evaluation procedure is presented in Section 4. We conclude in Section 5 with a discussion of our results and future work.

2. MEASURE RECOGNITION ALGORITHM

Our technique for locating the bounding boxes of measures within a music score relies on several image processing functions and follows the task model proposed by Bainbridge and Bell [1], which decomposes the problem of OMR into several key stages: image preprocessing and normalization, staffline identification and removal, musical object location, and musical reasoning.

After preprocessing the input image, stafflines are removed from the score and thin, long, vertical lines that have the same horizontal position within a system of music are located, even if they are connected. The connected height of these elements should approximately equal the height of the system to which they belong. Our approach assumes that barlines are usually taller than the stem of notes.

We experimented with several image processing algorithms in our optical measure recognition system. The Gamera software framework [10] provides a flexible and extensible environment for testing these methods and implementing new ones, if desired. Fig. 1 displays intermediary output of our measure recognition system at different

stages of processing, described in the following sections, on a portion of an image selected from our dataset.

2.1 Preprocessing

Before analyzing the input music score, the image must undergo two preprocessing steps: binarization and rotation correction. The binarization step coerces each pixel in the image to be either black or white according to a specified threshold parameter and is accomplished by using the Otsu binarization algorithm [12]. The rotation correction step automatically rotates skewed images and is accomplished by using the `correct_rotation` method that is part of the document-preprocessing bundle toolkit for Gamera [13]. [1]

2.2 Staff grouping hint

Our measure recognition algorithm requires prerequisite information, supplied by humans, that describes the structure of the staves on the music score being processed. This information, hereinafter referred to as the *staff grouping hint*, indicates how many staves are on the page, how many systems are on the page, how staves are linked together into systems, and whether barlines span the space between groups of staves.

The staff grouping hint is a string that encodes the structure of staves. For example, consider a page of piano music consisting of two staves that are broken into five systems, where barlines span the space between the two staves in each system, as in Fig. 2. The appropriate staff grouping hint for this page is `(2|)x5`, where parentheses indicate a group of staves, the pipe character `|` denotes that barlines span the space between staves in the group, and the `x` character indicates the number of systems the staff group is broken into. [2]

Although the staff group hint can be seen as a bottleneck because it requires human intervention, it is an important component of our system because it is used to properly encode the output symbolic music file and for fault detection of the staff detection algorithm, described in the next section. Also, most multi-page scores do not change their system structure across pages, and so a hint created for one page can often be used for the whole score.

2.3 Staff detection and removal

After preprocessing the input image, a staff detection algorithm searches for staves on the music score and returns the bounding box information for each staff. A staffline removal algorithm then discards the located stafflines from the music score. However, staff detection and removal algorithms yield variable results depending on the notation style of the music score, image scan quality, and the amount of noise (artifacts) present in the image. As a result of the high variability of images in our dataset, we could not rely on only one approach for detecting stafflines.

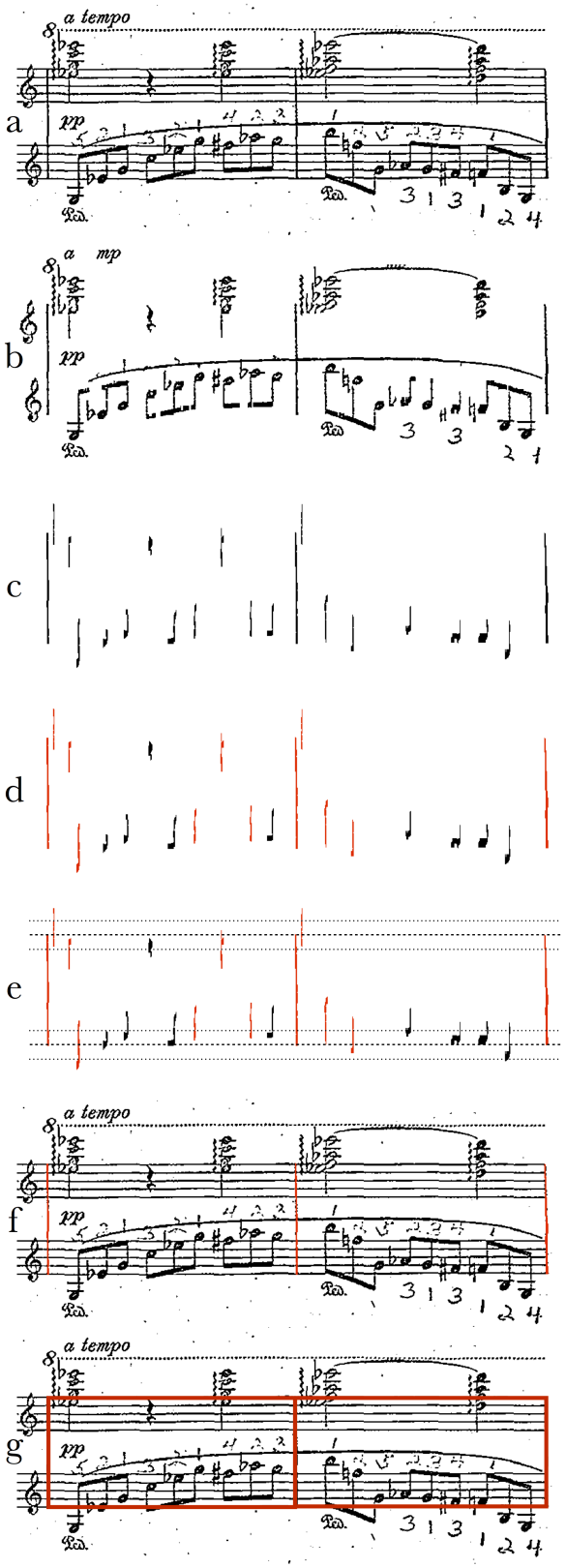


Figure 1. Process of extracting bar candidates. a) Original image, b) stafflines removed on preprocessed image, c) most frequent vertical black runs removed, d) barline candidates filtered by aspect ratio, e) filtering of bar candidates by vertical tolerance thresholding, f) final barlines, and g) final retrieved measures superimposed on the original image.

[1] `https://github.com/DDMAL/document-preprocessing-toolkit`

[2] More staff grouping hint examples can be accessed at `http://ddmal.music.mcgill.ca/optical_measure_recognition_staffgroup_hint_examples`

Low-quality scans and images with an excessive amount of artifacts frequently cause staff detection algorithms to fail, and so we implemented an error handling approach that tries one algorithm first, and if that fails, an alternative is used instead. We consider a staff detection algorithm to have failed when the number of detected staves does not equal the value derived from the provided staff grouping hint.

Following previous research [15], we used the *MusicStaves* Gamera Toolkit because it offers a number of different algorithms for detecting the position of stafflines in an image, and also for removing them. [3] We used the *Miyao* and *Dalitz* algorithms to perform staff detection. The *Miyao* algorithm provides a precise method for determining horizontal staff slope changes in inclined, broken, or non-linear staves by breaking the staffline into equidistant segments and capturing the vertical position of each segment [11]. The *Dalitz* staff detection algorithm [3] is also capable of tracking non-straight lines by using long quasi-black run extraction and skelenotization (i.e., the representation of each staffline as a one point thick continuous path), but it does not break the stafflines into equidistant segments. Our approach for finding measures depends heavily on detecting the position of the staff, and so we implemented two approaches in case one of them fails. We tested both configurations (i.e., *Dalitz-Miyao* and *Miyao-Dalitz*), and concluded that the former arrangement yields superior performance with respect to our ground-truth dataset. After successful recognition of the position of staves, the bounding box of each system can be calculated using the provided staff grouping hint.

Following the staff detection stage, staffline removal must be performed to eliminate each staffline within the music score. This process is important because it isolates superimposed music symbols on staves, facilitating their recognition. However, a comparative study established that there is no single superior algorithm for performing staffline removal [4]. Using a number of different metrics on images with deformations, we observed that the performance of many algorithms for staffline removal is similar, with no one technique being obviously better in general. Based on our previous work [15], we chose the Roach & Tatem staffline removal algorithm [14] in our optical measure recognition system. Fig. 1(b) shows the output of the staffline removal algorithm on a portion of a preprocessed image from our dataset.

2.4 Locating barline candidates

Once the position of each staff and system is calculated and all stafflines have been removed, we filter short vertical runs of black pixels in order to remove ligatures, beams, and other elements on the page that are unlikely to be candidates for a barline. The most frequent run-length is calculated and is used in subsequent processing steps. Since removing stafflines and short vertical runs frequently leaves unwanted artifacts on the page, we finally despeckle the image to remove all connected components smaller than a threshold value, which is dependent on the most frequent run-length value.

Once we have removed the horizontal lines from the image, we perform a connected components analysis to segment all of the residual glyphs on the page. The resulting set of connected components is filtered to only include thin, vertical elements, which are referred to as *bar candidates*. The discriminating feature for the selection of bar candidates is the *aspect ratio*: the relation between the width and the height of a component. Fig. 1(c) shows the result of filtering short vertical runs and despeckling the image. Fig. 1(d) highlights bar candidates that have an acceptable aspect ratio.

A bar candidate may be broken into several unconnected lines, depending on the quality of the original image, the effects of any of the intermediary processing steps, or simply from barlines that intentionally do not span an entire system. If bar candidates within a system have roughly the same horizontal position, they are connected into a single bar candidate. The height of each connected bar candidate is calculated and compared to the height of the system to which it belongs; these heights should approximately be the same. Moreover, the upper and lower vertical position of the bar candidate should lie sufficiently close to the upper and lower vertical position of its system, respectively. If the bar candidate fails to meet this criterion, the glyph is discarded. The sensitivity of this filtering step is controlled by the *vertical tolerance* parameter. Fig. 1(e) shows a visual representation of the vertical tolerance filtering process.

An additional filtering step addresses two common cases whereby certain bar candidates are included as false positives: The first situation occurs when accidentals preface the musical content on a staff. As most horizontal lines are removed in previous processing steps, the vertical lines that remain in the accidentals are usually horizontally aligned across the staves. Therefore, they are linked together into a single bar candidate, which results in a false positive. The second situation occurs when a double barline is considered as two bar candidates. Considering that the largest key signature has seven accidentals spanning twice the vertical size of the staff, we resolve both of the aforementioned issues by filtering bar candidates that are less than twice the height of the staff apart in the horizontal direction.

2.5 Encoding the position of measures

The result of the filtering processes is a set of recognized barline candidates from an input page of music, as seen in Fig. 1(f). These barlines candidates are sorted according to their system number and their horizontal position within the system. The bounding box for each measure is calculated by considering the location and dimensions of sequential barlines in each system, as seen in Fig. 1(g). The resulting set of measure bounding boxes is encoded in the Music Encoding Initiative (MEI) file format. [4] The MEI file format is used because of its ability to record the

[3] http://lionel.kr.hs-niederrhein.de/~dalitz/data/projekte/staffline

[4] http://music-encoding.org

structure of music entities as well as the physical position of all elements on the page [6]. Also encoded in this file is the overall structure of the score that indicates which measures belong to which system and which staves belong to which system.

3. GROUND-TRUTH DATASET

To the best of our knowledge, there are no standard OMR datasets that are complete with annotated measure bounding boxes. Therefore, we created our own to test and evaluate the performance of our optical measure recognition system. Our dataset consists of 100 pages extracted from the International Music Score Library Project (IMSLP). [5] We chose to extract images from IMSLP because of the quantity and diversity of the CWMN materials in its database.

To create this dataset we selected a random musical work from IMSLP, downloaded a random score or part from this work, and finally, selected a page at random from the score or part. As the purpose of our study is to locate the position of measures on pages of music, images of blank pages, pages with mostly text, and pages with no measures were manually discarded and replaced. In the initial draw of the dataset, images with these characteristics accounted for roughly 15 percent of the dataset. The set of downloaded images were in portable document format (PDF), which were processed using the libraries *pyPDF* [6] and *pdfrw*, [7] and converted to the tagged image file format (TIFF) using the *Adobe Acrobat Professional* application.

3.1 Measure annotations

Once the images in the dataset were transformed into the desired TIFF format, we created a ground-truth dataset of manually annotated bounding boxes for all measures on each page of music. We developed a Python application to perform the annotations. [8] The graphical user interface of the application displays an image to be annotated by a user, who indicates the presence of a measure on the page by clicking on the top-left position of a measure and dragging the mouse to the bottom-right corner of the measure. In order to ensure that all stafflines are straight, the image displayed to the annotators was automatically rotated using the same algorithm as in the preprocessing step of the presented optical measure recognition algorithm. The application encodes and saves the annotations as an MEI file, using a similar structure as the output of the optical measure recognition algorithm.

Two annotators with musical training of at least 10 years were hired to annotate the bounding box of each measure occurring in the entire dataset, as well as to provide a text file containing the staff grouping hints for each image. The annotators were instructed to track the time and number of pages they annotated per session, and to start annotating at opposite ends of the dataset to reduce the chances of error

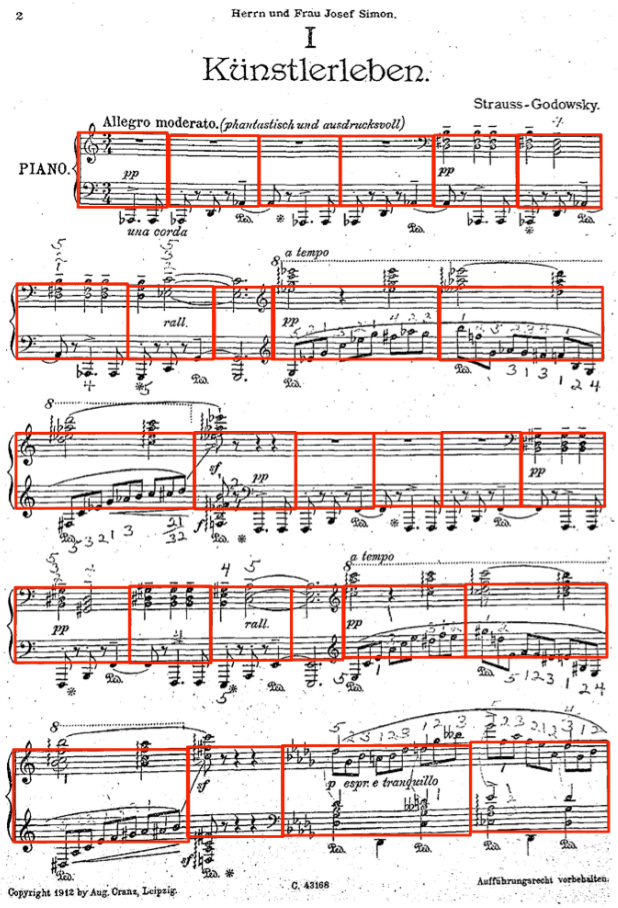


Figure 2. Manual measure annotations created using our standalone Python application for the image *IMSLP08436*, extracted from the International Music Score Library Project.

in the initial pages of the dataset. On average the annotators required 10 minutes to annotate the measures and create the staff group hint for each page. There were few discrepancies between the two annotators; the most common inconsistency was the staff group hint for complex pages of music. The dataset consists of 2,320 annotated measures, with a mean of μ = 23.43 measures per page, and a standard deviation of σ = 21.34. Fig. 2 displays a page of music from our dataset with the measure annotations superimposed.

Even with trained annotators, we encountered several challenges in the creation of this ground-truth dataset. Several recurrent issues arose during the annotation process, including how to interpret measures that are interrupted by a system break, how to annotate anacruses ("pick-up" notes), how to annotate repetition measures, and how to annotate measures that indicate changes in key signature but contain no notes. As our approach for finding measures on the score relies only on visual cues, all of the aforementioned cases are interpreted as separate measures. As such, we considered a measure interrupted by a system break, as well as an anacrusis, as two different measures. In addition, repeated measures were considered a single measure. Similar projects that recognize regions of a music score

[5] http://imslp.org
[6] http://pybrary.net/pyPdf
[7] http://code.google.com/p/pdfrw
[8] https://github.com/DDMAL/barlineGroundTruth

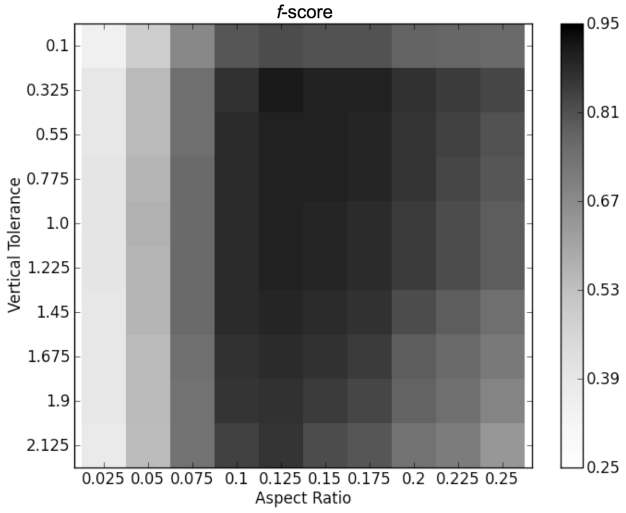


Figure 3. f-score results of the optical measure recognition algorithm. The x-axis displays different values of the *aspect ratio threshold* parameter. The y-axis displays different values of the *vertical tolerance threshold* parameter.

and synchronize these to audio playback have taken a similar approach, not yet considering repetitions of particular parts of a musical work [9]. Finally, areas of systems that contained the clef and accidentals, but no notes, were also considered to be a measure.

4. ALGORITHM EVALUATION

The performance of our optical measure recognition algorithm was evaluated by computing precision, recall, and f-score statistics for the automatically recognized measures on each page of music in the ground-truth dataset. Since we are not only concerned with the number of retrieved measures but also their size and physical position on the page, a measure is considered correctly recognized if its bounding box coordinates are within a quarter of an inch of its corresponding bounding box in the ground-truth measure annotations. [9]

Experiments were conducted to investigate the impact of different values of the two critical parameters of our optical measure recognition algorithm, namely the *aspect ratio* and the *vertical tolerance* parameters, described in Section 2. We iterated over a set of 100 combinations of these parameters and reported the resulting precision, recall, and f-score of our algorithm in each case.

Fig. 3 presents the results of the experiments across the entire dataset. It can be seen that the f-score value was highly influenced by the aspect ratio parameter. When this parameter was < 0.1 the f-score of our algorithm significantly decreased. This finding is intuitive because the appearance of elements on a music score are often variable, especially if it is handwritten. Consequently, it is unlikely to encounter barlines that are perfectly vertical with a small, constant width; they are typically skewed due to deformations in the image scan or contain artifacts resulting from intermediary processing steps of the optical music recognition algorithm.

The vertical tolerance threshold parameter, on the other hand, was found to not significantly affect the performance of the algorithm, especially when the aspect ratio threshold parameter was set to an optimal value. Only with extremely low values of vertical tolerance (i.e., when the tolerance was so small that the height of a bar candidate was expected to be almost the same as the system's height) did this parameter decrease the performance of the system. High values of this parameter also decreased the performance, but to a lesser degree.

Overall, the aspect ratio parameter had the most impact on the performance of the algorithm, though, both parameters exhibited a range of optimal values: aspect ratio threshold $\in [0.125, 0.150]$ and vertical tolerance threshold $\in [0.325, 1.00]$, which yielded an average f-score of 0.91 across the entire dataset. Similar barline recognition results have been obtained by commercial OMR systems [2]; however, in that study the evaluation dataset consisted of only five images and the algorithms being evaluated were undisclosed. Furthermore, Fotinea et al. [5] reported similar results on a dataset containing two pages of music.

Finally, certain pages in the dataset failed with all combinations of parameter values. The quality of these pages were generally quite poor and had discontinuous stafflines, which caused the staff detection algorithms to fail. Nevertheless, these pages were still included in the algorithm evaluation and resulted in an f-score of zero. We believe this accurately reflects how our system would perform in a "real-world" scenario.

5. CONCLUSION

We have presented work on developing a system that performs optical measure recognition on CWMN scores. Our approach follows an OMR workflow that includes image preprocessing, staff removal, and musical glyph recognition. Once all stafflines and short vertical runs of black pixels are removed, the algorithm finds thin, vertical elements on the page to form a set of barline candidates. Several heuristics were employed to filter this set of barline candidates into a final set of barlines, which were then used to calculate the bounding boxes of measures on the page. Our algorithm solely identifies measures on images of music scores, and thus, does not recognize other musical symbols such as the repeat sign, which instructs the performer to repeat a measure of music. This is problematic for applications that intend to synchronize digitized scores with audio playback and is an issue to address in future versions of our measure recognition system.

In order to test and evaluate our system, we manually annotated measure positions in 100 random pages of music from IMSLP and compared the bounding boxes produced by our optical measure recognition algorithm to the manual annotations using several descriptive statistics. We conducted several experiments to test different combinations

[9] Measurements in inches are converted to pixels using the *pixels per inch* parameter from the metadata of each image in the dataset.

of two critical parameters of our algorithm and discovered that the aspect ratio of a glyph is the most important discriminating feature for barlines. With optimal parameters, our algorithm obtained 91 percent f-score across the entire dataset.

Although our approach obtained similar results as previous systems, the scope and size of our evaluation dataset is much larger than those in the literature. We hope that the open-source, command line-based implementation of our system [10] can be easily integrated into existing OMR systems, and will stimulate future work in this area and help other researchers discover new ways to extract meaningful information from images of music scores.

6. ACKNOWLEDGEMENTS

The authors would like to thank our great development team for their hard work: Nick Esterer, Wei Gao, and Xia Song. Special thanks also to Alastair Porter for his invaluable insights at the beginning of the project. This project has been funded with the generous financial support of the *Deutsche Forschungsgemeinschaft* (DFG) as part of the Edirom project, and the *Social Sciences and Humanities Research Council* (SSHRC) of Canada.

7. REFERENCES

[1] Bainbridge, D., and T. Bell. 2001. The challenge of optical music recognition. *Computers and the Humanities* 35 (2): 95–121.

[2] Bellini, P., I. Bruno, and P. Nesi. 2007. Assessing optical music recognition tools. *Computer Music Journal* 31 (1): 68–93.

[3] Dalitz, C., T. Karsten, and F. Pose. 2005. Staff Line Removal Toolkit for Gamera. `http://music-staves.sourceforge.net`

[4] Dalitz, C., M. Droettboom, B. Pranzas, and I. Fujinaga. 2008. A comparitive study of staff removal algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30 (5): 753–66.

[5] Fotinea, S., G. Giakoupis, A. Liveris, S. Bakamidis, and G. Carayannis. 2000. An optical notation recognition system for printed music based on template matching and high level reasoning. In *Proceedings of the International Computer-assisted Information Retrieval Conference*, Paris, France, 1006–14.

[6] Hankinson, A., L. Pugin, and I. Fujinaga. 2010. An interchange format for optical music recognition applications. In *Proceedings of the International Society for Music Information Retrieval Conference*, Utrecht, Netherlands, 51–6.

[7] Knopke, I., and D. Byrd. 2007. Towards Musicdiff: A foundation for improved optical music recognition using multiple recognizers. In *Proceedings of the International Society for Music Information Retrieval Conference*, Vienna, Austria, 123–6.

[8] Kurth, F., M. Müller, C. Fremerey, Y. Chang, and M. Clausen. 2007. Automated synchronization of scanned sheet music with audio recordings. In *Proceedings of the International Society for Music Information Retrieval Conference*, Vienna, Austria, 261–6.

[9] Kurth, F., D. Damm, C. Fremerey, M. Müller, and M. Clausen. 2008. A framework for managing multimodal digitized music collections. In *Research and Advanced Technology for Digital Libraries. Lecture Notes in Computer Science* 5173: 334–45. Springer Berlin Heidelberg.

[10] MacMillan, K., M. Droettboom, and I. Fujinaga. 2002. Gamera: Optical music recognition in a new shell. In *Proceedings of the International Computer Music Conference*, La Habana, Cuba, 482–5.

[11] Miyao, H., and M. Okamoto. 2004. Stave extraction for printed music scores using DP matching. *Journal of Advanced Computational Intelligence and Intelligent Informatics* 8 (2): 208–15.

[12] Otsu, N. 1979. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics* 9 (1): 62–6.

[13] Ouyang, Y., J. Burgoyne, L. Pugin, and I. Fujinaga. 2009. A robust border dectection algorithm with applications to medieval music manuscripts. In *Proceedings of the International Computer Music Conference*, Montréal, Canada, 101–4.

[14] Roach, J., and J. Tatem. 1988. Using domain knowledge in low-level visual processing to interpret handwritten music: An experiment. *Pattern Recognition* 21 (1): 33–44.

[15] Vigliensoni, G., J. A. Burgoyne, A. Hankinson, and I. Fujinaga. 2011. Automatic pitch detection in printed square notation. In *Proceedings of the International Society for Music Information Retrieval Conference*, Miami, FL, 423–8.

[10] `https://github.com/DDMAL/barlineFinder`

MUSICBRAINZ FOR THE WORLD: THE CHILEAN EXPERIENCE

Gabriel Vigliensoni[1], John Ashley Burgoyne[2], and Ichiro Fujinaga[1]

[1] CIRMMT
McGill University
Canada
[gabriel,ich]@music.mcgill.ca

[2] ILLC
University of Amsterdam
The Netherlands
j.a.burgoyne@uva.nl

ABSTRACT

In this paper we present our research in gathering data from several semi-structured collections of cultural heritage—Chilean music-related websites—and uploading the data into an open-source music database, where the data can be easily searched, discovered, and interlinked. This paper also reviews the characteristics of four user-contributed, music metadatabases (MusicBrainz, Discogs, MusicMoz, and FreeDB), and explains why we chose MusicBrainz as the repository for our data. We also explain how we collected data from the five most important sources of Chilean music-related data, and we give details about the context, design, and results of an experiment for artist name comparison to verify which of the artists that we have in our database exist in the MusicBrainz database already. Although it represents a single case study, we believe this information will be of great help to other MIR researchers who are trying to design their own studies of world music.

1. INTRODUCTION

Thousands of commercial and non-commercial websites offer information and metadata about different aspects of music and artists, for example, their recordings, biographies, discographies, video clips, and other resources. However, the data they provide is often disorganized and not interlinked, and the websites disappear frequently. Hence, it is likely that the information collected over years can be lost. It seems sensible to gather all music-related data in a centralized database that can be accessed by several websites or systems. The goal of our project is to take data from several semi-organized collections of Chilean-music cultural heritage—websites and databases which combined represent almost all music that has been composed and performed in Chile—and integrate it into an open-source music database, where information is easy to search and will last for a longer time.

1.1 Music Metadata

Metadata is structured information that identifies, describes, locates, relates, and expresses several, different layers of data about an information resource [3, 5, 8]. It can be of three basic types: *descriptive*, for purposes such as identification and discovery; *structural*, for expressing relations among resources; and *administrative*, for managing resources [8]. Descriptive music metadata commonly provides information about recordings, expressing the song title and length, the artist name, and the release name of a musical object, usually stored in a MP3 ID3 tag. Structural music metadata is used to document relationships within and among digital musical objects to allow navigation, such as the song's order in an album or a playlist, linking song names to video clips, artists to their biographies, and so on. Administrative music metadata can include information such as software and hardware used to digitize the musical resource, system requirements to read the files, use restrictions or license agreements that constrain the use of the resource. Thus, music metadata has been called a digital music commodity because it adds value to the musical objects, mediating the experience of listeners with music and artists, helping them to browse, explore, sort, collect, use, and finally enjoy music [4, 7].

For this project, we collected data about Chilean music from several websites and databases of different scope and size. [1] Centralizing and interlinking these resources should create synergies in the data because one single query will give access to all resources, creating relations that were not established in their previous locations, thus contributing to a larger web of data. Hence, finding the best repository capable in storing and expressing these data relationships was considered significant and crucial for the success of our endeavour.

1.2 Music-Related Metadatabases

Although there are many commercial, professionally reviewed, online music libraries, we focused only on user-built, open-data, music metadatabases. By contrast to commercial libraries, open databases provide user-access to enter data and have similar types of licenses for the use of its data as the Chilean websites. We present now a list of the main services available for free, public use in terms of their

[1] Data from all databases available at
http://www.vigliensoni.com/McGill/BDMC/8_DATA

	Scope	Size	Information resources stored	Relations	IDs	Data quality assurance	Language guidelines	API	Other
FreeDB	CD tracklists	∼2M albums	Album, title, year, genre, time offsets	Disc to genre	FreeDB propriety Disc-ID	Unspecified automatic method	No	No	Limited search engine. Small, fixed set of genres. Two artists can have the same ID.
MusicMoz	Music-related factual data and Internet links	>136K items	Artist's biographies, discographies, profile, reviews, and articles. Music-related links and resources	No relations allowed	Artist name-based URI	MusicMoz's authorized editors	No	No	Many links are no longer available. Unusual ontology of musical categories.
Discogs	Physical discographies and music releases	>2M artists >3M albums	Artist, release, master, label, image	Small set of relations	Artist name-based URI	Community-based high-quality data	English only	RESTful, XML-based API	30s previews. Well designed solution for artist name variation. Marketplace available. Could evolve to a commercial site.
MusicBrainz	Any kind of music release	>600K artists >1M releases >11M tracks	*Core entities* (artist, label, recording, release, release-group, work), entities, and their relationships	*Advanced Relationships* can be expressed for all core and external entities	MB IDs are universally unique identifier (UUID)	Community-based high-quality data	Guidelines for 30 languages	RESTful, XML-based API	MB links data from all other metadatabases. MB's *Advanced Relationships* can be mapped into RDF. MB stores acoustic fingerprints.

Table 1. Comparison of free, user-built, open-data, music metadatabases: *FreeDB*, *MusicMoz*, *Discogs*, and *MusicBrainz*.

scope, size, stored information resources, relations among these resources, ID system, data quality and assurance, language guidelines, and API.

FreeDB is a license-free database of CDs track-listings from user-contributed data, originally based on the *Compact Disc Database* (CDDB). [2]

MusicMoz is a user-contributed database that stores music-related, factual data and Internet links. [3]

Discogs is a large, user-populated database of discographies and music releases from physical sources. It provides data of high quality that is ensured by a strict input form mechanism and a large community of users. [4]

MusicBrainz is a large, community-based, user-contributed metadatabase that stores the three aforementioned types of metadata for any kind of music release. Its high-quality database is managed by an open community of non-professionals that negotiate periodically and consistently, with strict standards and routines, about the orientations, developments, style guidelines, and mostly everything on MusicBrainz [4]. [5]

Table 1 shows a comparison of the four metadatabases. It can be seen that, among all these databases, MusicBrainz is the database with the broadest scope, not being restricted to only physical copies, as in the case of Discogs, or CDs, as in FreeDB. Also, MusicBrainz is the only music database capable of storing not only descriptive, but also structural and administrative metadata. This fact is critical for expressing the many relationships among musical resources, as well as for providing efficient ways of managing and interlinking them. For example, the MusicBrainz universal unique identifier-based IDs (MBIDs) practically ensure unique identifiers for all its *core entities* (i.e., artist, label, recording, release, release-group, and work), and they have been already widely used for linking music data in the semantic-web community. Furthermore, the MusicBrainz community decided to develop a set *Advanced Relationships* to describe relations between its core entities, and to publish all the MusicBrainz database, resources and their relationships, as Linked Data. The recently-finished *LinkedBrainz* [6] subproject was intended to map all these relationships into RDF [6]. MusicBrainz also stores acoustic fingerprints (PUIDs) that can be used to search for a resource even if its descriptive metadata is not available. Users, as well as performance right organizations, could benefit from this feature for music rights identification purposes. Moreover, MusicBrainz has also a strict, language-specific, style guidelines for 30 different languages, which explain how data should be formatted in their database. This fact is very important to this project—and also to other similar projects coming from countries and regions where English is not the primary language—because it allows, and forces, the non-English-speaking users to follow the correct standard for their language. Finally, MusicBrainz is the most "open" of the four reviewed metadatabases because it provides methods to link data from other websites and databases to MusicBrainz (e.g., by extracting CD track lists from FreeDB, linking artists' images from Discogs, data from MusicMoz, *Allmusic*, *BBC Music* and *Wikipedia*; or album covers and videos from *Amazon* and *YouTube*, respectively).

[2] http://www.freedb.org/
[3] http://www.musicmoz.org/
[4] http://www.discogs.com/
[5] http://www.musicbrainz.org/
[6] http://wiki.musicbrainz.org/LinkedBrainz

By analyzing all the aforementioned characteristics, we decided to work with the MusicBrainz metadatabase in order to store and make available the corpus of music that we want to work with: Chilean music.

2. COLLECTING DATA ABOUT CHILEAN MUSIC

Over the last ten years, several endeavors for creating websites devoted to Chilean music have been developed. These projects have collected data about artists' discographies, biographies, video clips, and album and concert reviews. Currently, however, there is no way of creating a common query to retrieve all available data for a specific artist, album, or song because these resources are neither centralized nor interlinked, do not have URIs, and the websites depend on scarce sources of funding, and so sometimes they have to shut down.

2.1 Websites and Databases of Chilean Music

During the past few years, Chilean music-related metadata has been accessible through the following five major websites:

Base de datos de la Música Chilena (BDCH) (the Chilean Music Database) is a restricted digital music database developed by the Sociedad Chilena del Derecho de Autor (SCD, the Chilean Society of Authors), the only performance rights organization in Chile. The BDCH provides all associate radio stations across the country with a secure, fast, and easy-to-use website for accessing and exploring the largest repository of Chilean music. Radio stations can legally download and airplay music from the BDCH. The scope of its collection is wide, ranging from rock, pop, and ballad, to classical, experimental, and jazz. Metadata for each song has been manually generated by the artists or producers themselves and cleaned afterwards by expert annotators. Data includes composer, author, interpreter, album, label, genre, and label. [7]

Musicapopular (MP) is a website developed and maintained by music journalists, advised by a musicologist, and funded by several short-term governmental grants. Its database is updated periodically and their authors have a commitment for data accuracy and quality, which makes MP a good source when looking for information about Chilean music artists. It provides artists' biographies and discographies, the evolution of band members over time, birth and death dates for individuals, and start and end dates for bands. All its data is mostly interlinked, but only within the website. Although MP has a genre taxonomy tailored to Chilean music (e.g., *nueva canción chilena*, *música chilota*, or *proyección folclórica*), the mainstream genres, e.g., rock and pop, comprise the bulk of the database. [8]

Mus (MUS) is a website devoted to new album and concert reviews. It was funded by the SCD and short-term governmental funds and was maintained periodically by music journalists. Although the website was shut down on January 2012, its data can still be accessed with the exact URLs of the resources. The site does not provide any search methods.

Portaldisc (PD) is a web portal where Chilean music of many genres is sold. PD shares efforts with the SCD as well as MP, selling most of the catalog belonging to and reviewed in those collections. Its website provides access to audio previews of all songs as well as short album reviews. PD can be searched only by artist name. [9]

Videoclipchileno (VCCL) is a website with a nearly comprehensive collection of Chilean music video clips. It provides not only links to the video clips and their directors, but also contextual information about them. It is maintained periodically by the same group of music journalists that run MP, and they are waiting for new governmental funding to improve the scope and functionality of the site. VCCL's search engine accepts a keyword that is searched across all fields in their database. [10]

Most data retrieved from the aforementioned websites can be represented explicitly with the MusicBrainz metadata schema. Also, by means of creating advanced relationships between resources from different sources, these can be linked and accessed from the MusicBrainz web page or API.

2.2 Data Harvesting and Parsing

Because most of the surveyed websites depend on external sources of funding, they can be short-lived and their data—and people's work behind it—can be lost. For that reason, we decided to harvest all data from the available websites.

Although the nature of the data in all databases was different, it can be combined or its overlap can be used to extract more accurate data. For example, the album reviews can provide the track list, and so they can be used to obtain the actual list of songs for an specific album, or to help to disambiguate any difference in the track lists. Table 2 shows the data types and approximate number of entries we extracted by scraping the websites. [11]

There were some problems when scraping, parsing, and storing the data from the websites.The first problem was handling all non-ASCII characters. These characters typically arise because most of the words are written in Spanish, but also sometimes because of special characters that artists use for their names or for the names of their songs.

[7] http://bdch.musica.cl/web_bdch/
[8] http://www.musicapopular.cl/
[8] http://www.mus.cl/
[9] http://www.portaldisc.cl/
[10] http://www.vccl.tv/
[11] Code and scripts available at http://www.github.com/vigliensoni/bbdd_much/

Database	Data retrieved
BDCH	40,000 songs
	33,000 different songs
	3,300 artists
	3,000 albums
	400 record labels
	80 genres
MP	1,500 bands
	1,800 individuals
	1,800 biographies
	40 genres
MUS	500 albums
	300 interviews
	600 concerts
PD	3,600 album reviews
VCCL	1,600 video clips

Table 2. Approximate collection sizes and data types within the five major Chilean music databases.

We ended up using Unicode for representing all data internally. Another problem we faced was the many variations that a resource name can have across repositories, releases, or even within the same website. For example, different people can use alternative forms of an artist name (e.g., "Dj Bitman", "DJ Bitman", "Dj Bit Man", and so on). Also, artists themselves can use variations of their names across several releases (e.g., "Bitman" and "DJ Bitman"). Finally, many slight variations of the same resource name can exist across different repositories due to human error when entering the data. Resolving these inconsistencies can be a tricky problem, for example, as in recent work from Angeles et al. [1] where they combined a metadata manager software with fingerprinting-based querying and still obtained a low rate of consistency between resource names among different databases. Libraries' practice of using authority files for identification and disambiguation of catalog names is able to deal with these inconsistencies, however, most Chilean artist names are still not cataloged in institutional databases. This project tries to collect data from several sources, disambiguate name variations, and store the data into a single searchable database. In Section 3.1 we will detail a string matching-based experiment that helped us take a step toward solving this problem.

3. DATA MATCHING WITH MUSICBRAINZ

After we had consolidated all data, we wanted to know how many entries were (and were not) already in the MusicBrainz database. Using the MusicBrainz web services, we proceeded to compare our data with MusicBrainz and obtained 27, 23, and 21 percent of matches for artists, albums, and songs, respectively. However, among the artists, we retrieved a large number that were false positives (i.e., wrongly recognized as the queried artist when they did not correspond with the actual Chilean artist). In order to reduce these inconsistencies, we decided to add constraints on the resulting resources. The country name alone seemed a good predictor for fixing this problem, but we were discouraged because currently only 22 percent of the total number of artists in MusicBrainz have a country name assigned. Even worse, among them there are only 200 artists with a *CL* country-code value, the ISO 3166-1 Alpha-2 code for Chile. A second problem we found was that sometimes the actual, true positive result retrieved by the Lucene-based MusicBrainz search server was not the one in the first position or the one with the highest score. We realized that we would need to iterate over all retrieved results and compare the strings in order to see which one was the proper match.

To improve the number of true positives for our query, we designed and ran an experiment considering the advanced search method that MusicBrainz provides and the two aforementioned issues. The objective of the experiment was to determine an ideal threshold that allows us to have the largest precision and recall for the artist names. That led us to three questions:

- How many artists have an exact match (i.e., they are already in the MusicBrainz database)?
- How many artists do not match (i.e., they are not in the MusicBrainz database)?
- How many artists match partially? Among these we need to see what is the best threshold to obtain the largest precision and recall.

3.1 Approaches to Artist Name Matching

Our approach for the experiment was two-fold. On the one hand, we created a query that consisted of the artist name plus the country code; we also looked for any comment with the word *Chile* in the annotation field for artist disambiguation (i.e., MusicBrainz's own method for disambiguating similar artist names in its database). On the other hand, we hypothesized that for selecting the proper string from all the retrieved results, we could rely on measuring the string difference between the query string and each one of the retrieved results: the one with the smallest difference would be the true positive. Thus, to handle all nuances or variations of the strings due to special Spanish characters or typographical errors when the artist name was entered, we implemented two string metrics with three variations each:

Levenshtein distance (L) permitted us to calculate the cost of the best sequence of edits to convert one string into the other. We used it in its ratio form to consider the number of letters of the query, so we obtained a normalized value, where 1 represents an exact match.

Jaro metric (J) also allowed us to calculate a normalized value that represents the number of edits needed to convert one string into the other, but it further weights positively or negatively if source characters are or are not present in the target string [2].

Normal string (N) was a direct comparison of the original strings.

ASCII-fied string (A) allowed us to compare ASCII-fied versions of the strings where all tilde and accents were removed from the strings.

ASCII-fied, lower-cased, no-space string (P) allowed us to compare strings where all characters were ASCII-fied and lower-cased, and from which the spaces were also removed.

Hence, we ended up with the *LN*, *LA*, *LP*, *JN*, *JA*, and *JP* variations of the similarity of the artist name query string against each one of the retrieved artist names by the MusicBrainz web service. It should be noted that in all cases the Lucene special characters were escaped before doing the query, and so if an artist name had any of these characters, it was not considered.

3.2 Experimental Procedure

For the experiment, we designed a testing subset with artist names randomly chosen from the ones in our dataset. This subset was created among those artists with string distances between the range of 0.75 and 1.00. The size of the dataset was 800 entries, which represents roughly a quarter of the total number of artists in our dataset. We manually searched to see if the artists in the dataset already existed in the MusicBrainz database. This process was long and complex because there were many false positives among those artists with common and short names, such as *Rachel*, *Quorum*, *Criminal*, *Twilight*, and many others. The only way to determine if the query returned a true or false positive was searching in the artist's country, releases, relationships, or works, and seeing if anything there matched some data in our database. We then ran the query for each entry, chose the items with the largest metric values, and identified true and false positives, and false negatives. We then calculated precision and recall for the whole subset at different thresholds, and calculated the error for each metric and threshold using the bootstrapping technique. Figure 1 shows the precision and recall for each threshold, metric, and variant. Error curves at α = 0.05 generated from a bootstrap sample of 1,000 replications of the original sample are also shown. These plots have to be understood as a series of two complementary runs. The plots in the left refer to precision (upper left) and recall (lower left) for the comparison of strings only. In this case, all retrieved artists with the same name were considered as a true positive, even if they are not in fact the same artist. This approach was taken because none of the six metrics can know in advance whether the retrieved name denotes the specific person that we are looking for. We can observe that if we were to be too strict with the string distance threshold, the precision would be high but the recall would be low, as expected.

In terms of precision, it can be seen that Levenshtein string distance (L) in its variants for the normal string (N), its ASCII-fied version (A), and the ASCII-fied, lower-cased, and no-spaced string version (P) performs better than the Jaro (J) distance for thresholds between the range of 0.80 and 0.90. The three different variants (N, A, P) do not make statistically significant differences in the results for any of the methods. However, in terms of recall, it can be seen that the P versions of both L and J are the ones with the best recall, especially when the thresholds become higher. There is no statistically significant difference between the performance of P and A, but at high thresholds, N statistically significantly underperforms both P and A. We can conclude that for the best results, the queried string should at least be ASCII-fied and the Levenshtein distance should be used. The threshold point to obtain the best precision while still having a good recall can be found around 0.90.

The two plots in the right of Figure 1 refer to the same experiment and dataset but evaluated with respect to their real-life context. In other words, in this second case, if the query returned an artist with the same name but it referred to another artist in real life, the returned artist was considered as a false instead of a true positive. Under these conditions, we reached a ceiling of precision at about 60 percent, which establishes that some kind of verification process is essential when using name matching in a real-world application.

4. CONCLUSIONS AND FUTURE WORK

We have done a review of user-contributed, music metadata libraries, and we have shown why we have chosen MusicBrainz as the metadatabase that we will use. We have also shown the five websites that we used for collecting Chilean music-related metadata and given details about the data that each website provides.

When we tried to combine our database with MusicBrainz, we realized that there was a large amount of false-positive noise in the results for each query. We tried several types of queries and determined that an advanced search method, including several fields at the same time, was required, and also that it was necessary to iterate over all results and perform a string comparison between the query and the retrieved query to look for the true result. Hence, we developed an experiment whereby we created a random ground-truth subset of artist names from our database. We then queried these same entries with MusicBrainz, compared the retrieved results with the ground truth, and then tried to define the optimum variation threshold that should be accepted between the queried and the retrieved string in order to obtain the best precision and recall. We compared two different metrics with three variations each. For precision, the Levenshtein ratio offered the best performance, stabilizing its curve close to a normalized value of 0.90, where 1.00 means that the two strings are identical. The three variations did not matter. For recall, however, we established that the strings should, at least, be ASCII-fied to obtain a better recall. Overall, the best string comparison threshold on a normalized scale is close to 0.90, and the method used should be Levenshtein distance with an ASCII-fied, lower-cased, no-spaced version of the strings.

Short-term future work for this project will be to apply these experimentally obtained values for the string comparison across albums and songs. However, it is expected that the results will be much better because we will know

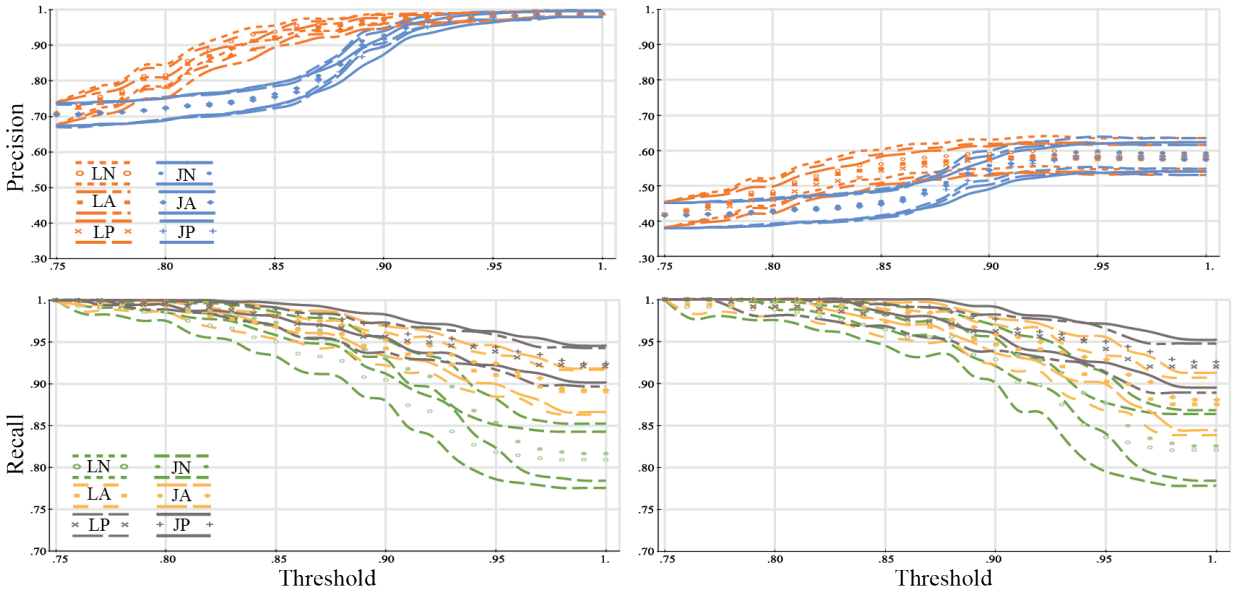


Figure 1. Precision and recall for artist names string comparison between the consolidated data from Chilean music databases and MusicBrainz. While upper and lower left plots show the results for the raw string comparison using Levenshtein (L) and Jaro (J) string distances and three variants (N, A, P), the ones in the right show the results in real-life context, where false positives were discarded. Error curves show upper and lower limits for 1,000 populations replicated from the original sample using bootstrap at $\alpha = 0.05$.

beforehand if the album's artist or song's artist is already in the MusicBrainz database. For future work in the mid-term, we will enter all data we collect into the MusicBrainz database. The MusicBrainz API does not allow one to automate this kind of process, but we experimented using a POST method to fill the forms automatically, and a user would simply need to review and submit the data to the database. By these means, all the data we collect will be available from MusicBrainz. As a second mid-term project, and by asking permission to the SCD, it would also be possible to use the audio, thereby allowing audio analysis over the whole corpus of music. This kind of analysis would be beneficial for everyone in the music-industry chain. For example, running structural music analysis to know where the choruses are in a given song, or knowing the overall tempo of a song, would be of benefit for the radio stations that use the BDCH, or the general public interested in creating a remix of a given song. In the long-term, the experience gained and the techniques developed will be applicable to other countries and cultures, to enable them to merge their metadata to global central databases.

5. ACKNOWLEDGEMENTS

This research was supported by the Social Sciences and Humanities Research Council, and by BecasChile Bicentenario, CONICYT (Comisión Nacional de Ciencia y Tecnología), Gobierno de Chile. The authors would like to thank Alastair Porter for sharing valuable knowledge about the MusicBrainz web services and API, and to the *Sociedad Chilena del Derecho de Autor* (SCD) for granting us access to their database.

6. REFERENCES

[1] Angeles, B., C. McKay, and I. Fujinaga. 2010. Discovering metadata inconsistencies. *Proceedings of the International Society for Music Information Retrieval Conference*. 195–200.

[2] Bilenko, M., R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. 2003. Adaptive name matching in information integration. *Intelligent Systems, IEEE* 18 (5): 16–23.

[3] Dublin Core Metadata Initiative. Retrieved April 6 2013, from http://dublincore.org/metadata-basics/.

[4] J. Hemerly. 2011. Making metadata: The case of MusicBrainz. *SSRN eLibrary* 1982823.

[5] International Federation of Library Associations and Institutions. *Digital Libraries: Metadata Resources*. Retrieved April 6 2013, from http://archive.ifla.org/II/metadata.htm.

[6] Jacobson, K., S. Dixon, and M. Sandler. 2010. LinkedBrainz: providing the MusicBrainz Next Generation Schema as Linked Data. *Late-breaking demo session at the 11th International Society for Music Information Retrieval Conference*.

[7] J. W. Morris. 2012. Making music behave: Metadata and the digital music commodity. *New Media & Society* 14 (5):850–66

[8] National Information Standards Organization. 2004. *Understanding metadata*. NISO Press, Bethesda, MD, USA.

INFLUENCES OF ISMIR AND MIREX RESEARCH ON TECHNOLOGY PATENTS

Sally Jo Cunningham
Department of Computer Science
University of Waikato
sallyjo@waikato.ac.nz

Jin Ha Lee
Information School
University of Washington
jinhalee@uw.edu

ABSTRACT

Much of the current Music Information Retrieval (MIR) research aims to contribute to the field by creating practical music applications or algorithms that can be used as part of such applications. Understanding how academic research results influence and translate to commercial products can be useful for MIR researchers, especially when we try to measure the impact of our research. This study aims to improve our understanding of the commercial influence of academic MIR research by analyzing the patents citing publications from ISMIR (International Society for Music Information Retrieval) Conference proceedings and its associated MIREX (Music Information Retrieval Evaluation eXchange) MIR algorithm trials. In this paper, we provide our preliminary analyses of the relevant patents as well as the ISMIR publications that are referenced in those patents.

1. INTRODUCTION

The ISMIR (International Society for Music Information Retrieval) conference started as a small-scale symposium in 2000 and continued to grow over the past decade as the field of Music Information Retrieval (MIR) matured. ISMIR has been one of the most important MIR conferences since the early establishment of the field, and serves as a key venue for dissemination of MIR research. The associated MIREX (Music Information Retrieval Evaluation eXchange) event, first run in 2005, has similarly grown to be a focus for MIR system and algorithm evaluation. One of the key objectives of MIR research is to make practical contributions toward the development of commercial music applications and services to improve users' interaction and experience with music. There is anecdotal evidence that our research results inform the development of new music applications and services, especially since ISMIR conferences and MIREX trials have continued to attract participants from the commercial sector. However, to date no research has been conducted to systematically investigate the extent of the practical impact of academic MIR research published through the ISMIR conferences and MIREX trials.

To this end, we first identify patents that reference publications from ISMIR and MIREX (Section 3). We then perform an informetric analysis over these patents and the referenced publications drawn from ISMIR and MIREX, to discover patterns of influence of ISMIR and MIREX on patented MIR technology (Section 4).

2. PREVIOUS WORK

Previous investigations of the characteristics of MIR research have focused exclusively on the field as viewed through academic publication. The research methods used were primarily bibliometric—that is, quantitative measures such as citation analysis, based on data drawn from the metadata and text of the ISMIR and MIREX proceedings. These techniques have been used to paint rich pictures of the state of ISMIR academic research at various stages in the history of ISMIR and MIREX [1], [2], [3], with the emphasis on scholarly publishing.

This present paper applies these bibliometric techniques to a set of patent filings rather than academic papers. 'Patent bibliometrics,' the natural extension of bibliometric techniques to collections of patent metadata and texts, has seen widespread use in technology-related fields since its introduction in 1994 [8]. The introduction of online, free-to-search patent databases has further encouraged patent bibliometric investigations [6]; the most comprehensive and widely used databases are provided by the Unites States Patent and Trademark Office (USPTO) and the European Patent Organisation (EPO).

One natural topic of interest has been the relationship between academic publications and patents in a given field. Data regarding this relationship should be straightforward to draw from a patent database, as each patent filing includes the equivalent of the bibliographic citations in the form of references to prior art, and the prior art can include both earlier patents and relevant conference and journal papers.

Unfortunately, patent databases index only the 'front page' prior art citations, and these are almost exclusively limited to patents. References to prior art in the form of relevant academic publications are typically found in the body of the patent—which is not indexed by the databases—and, though they may appear on the patent's front page as "Other Publications," they are not indexed in the USPTO and EPO patent search engines [7].

Given these hurdles to identifying linkages between patents and academic research directly from the patent databases, researchers have had to draw in evidence from additional sources, or to use crude proxy measures for the linkages. For example, a study of academics in Norway compared the publishing behavior of matched sets of academic patent inventors and non-inventors—but since the databases often did not include inventor affiliations, that had to be determined through expensive, error-prone, and time-consuming surveys of the institutions themselves [5]. Meyer [6] estimates the interactions between a country's academic and patented outputs through the coarse mechanism of comparing patent and publication rates for that country in a small number of narrowly focused fields.

The introduction of search over the full text of a patent by Google Patent Search (GPS) [1] supports patent bibliometric investigations to a far greater degree than has been possible previously, in particular supporting citation analysis of references to published scientific literature in patents. To our knowledge, this present paper is the first to use this facility to directly explore the influence of academic research on the patents.

3. DATA COLLECTION

We used Google Patent Search in order to find all the patents containing references to ISMIR and MIREX. Google Patent Search is a specialized Google search engine that indexes patents and patent applications drawn from public domain patent databases of the USPTO and the EPO. While both the USPTO and EPO offer search facilities for their individual collections, the Google Patent Search presents a unified interface to both databases. More significantly for this present investigation, Google Patent Search indexes the entire patent for search, where the USPTO and EPO support search only over the basic patent metadata (title, classification code, publication date, inventor, etc.). Through GPS, we can link back to the scientific literature supporting a patent, via the references cited in the textual description of the background to the invention and of the invention itself.

Searches for the initial patent datasets were conducted over Google Patent Search in April 2013, using the terms 'ISMIR music' and 'MIREX music'. Where multiple patent filings under the same inventor name and title were found, we retained the earliest filing, and the granted patent record over the application. As a consequence of preferring the record with the earlier filing date, in most cases the US patent record was retained rather than European (a common pattern in technology patents is to file first in the US). As a result, we found a total of 141 patents citing ISMIR papers, and 13 patents citing MIREX.

For each of these patents, we identified: the year of filing; whether this patent instance was an application or had been granted; if granted, the year; the inventor(s) and assignee(s); and the number of references to ISMIR and MIREX in each patent. For each reference to ISMIR or MIREX, we collected basic bibliographic data (title, authors, and year of publication).

4. DATA AND DISCUSSION

This section presents an analysis of both sets of patents—those referencing ISMIR publications, and those referencing MIREX. Given the disparity in the size of these two datasets (141 and 13, respectively), we present the results separately.

4.1 Analysis of the Patents

In this section, we summarize the broad characteristics of the 141 patent documents that reference ISMIR publications: distribution of patents by year of application, unique inventors and assignees, ISMIR references included in the patents, and patent topics.

4.1.1 Year of Application

Of the 141 patents identified as referencing ISMIR publications, 102 have been issued as of the date of our dataset gathering (April 2013), and 39 exist as applications. Figure 1 shows the number of patents by application year.

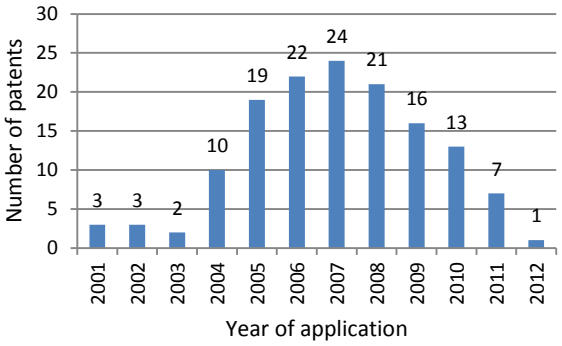


Figure 1. Number of patents by year of application

Examining the dates of application, we see a peak of patent applications referencing ISMIR publications in 2007, with a sharp drop-off in 2011 and 2012. The presence of only a single ISMIR-referencing patent filing in 2012 may be partly explained by a time delay in updates to the Google Patent Search and the underlying USPTO and EPO databases; according to the USPTO website, most patent applications filed on or after November 29, 2000, are published 18 months after the filing date of the application[2]. Also the EPO website states that the patent application is published 18 months after the date of filing

[1] https://www.google.com/patents

[2] http://www.uspto.gov/faq/patents.jsp

or the priority date[1]. Still, the steady decline remains; this may be due to the fact that technologies in some areas are saturated and new developments are appearing in a slower pace than they used to in the earlier stage of the MIR field. We consider possible explanations for this trend as we examine the characteristics of the ISMIR references themselves (Section 4.2).

4.1.2 Analysis of Inventors and Patent Distribution

There are 241 unique inventors in the ISMIR dataset, with an average number of 1.39 patents per inventor. All inventors are natural persons (US patents were preferred in selecting between multiple filings, and under US law corporate entities cannot be registered as inventors). The distribution of patents over inventors (Figure 2) indicates a strong skew towards single-filing inventors; here, approximately 80% of inventors are associated with one filing. We found this a bit surprising as one would assume that typically a team of researchers is involved in developing new algorithms/technologies (as evidenced by a growing trend toward co-authorship in ISMIR [1]).

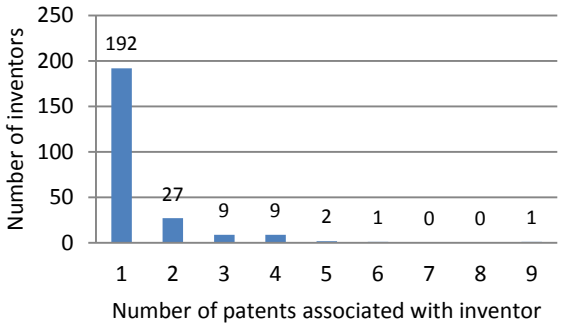


Figure 2. Number of patents sorted by number of inventors

The four inventors associated with the largest number of patents are presented in Table 1. Of the four, Masataka Goto and Brian A. Whitman have published in ISMIR proceedings.

No. of patents	Inventors
9	Louis B. Rosenberg
6	Thomas Kemp
5	Masataka Goto Brian A. Whitman

Table 1. Inventors with the largest numbers of patents referencing ISMIR

4.1.3 Analysis of assignees and Patent Distribution

Looking at assignees, there are 120 unique assignees, with an average of 1.18 patents per assignee. Of the 120 assignees, 21 are individuals, and 99 are corporate entities (primarily commercial organizations and universities).

Figure 3 shows the number of patents per assignee, which ranged from 1 to 12; there were 120 unique assignees in total. Most of the assignees were associated with a single patent (58.3%). Looking at the top 10 patent-holding assignees (Table 2), we see a mix of large IT corporations (Google, Apple, Microsoft, Yahoo!), two electronics corporations (Philips, Sony), a music and video metadata specialist company (Gracenote), and two organizations specializing in patent acquisitions (Colwood Technology, Outland Research).

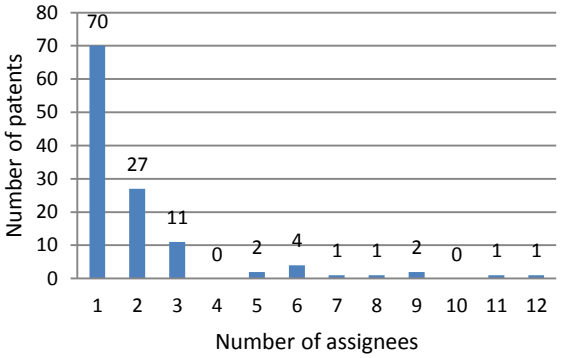


Figure 3. Number of assignees per patent referencing ISMIR

No. of patents	Assignees
12	Google Inc.
11	Apple Inc.
9	Colwood Technology, Llc
9	Outland Research, Llc
8	Strands, Inc.
7	Microsoft Corp.
6	Gracenote, Inc.
6	Koniklijke Philips Electronics N.V.
6	Sony Corp.
6	Yahoo! Inc.

Table 2. Top 10 assignees by number of patents

4.1.4 Number of ISMIR References per Patent

There were a total of 213 references to ISMIR publications in the patents we analyzed. The average number of ISMIR references per patent was 1.5. Unfortunately we cannot estimate the proportion of ISMIR references to all references included in the patents; the lack of standardization in patent descriptions and background formats, compounded by errors in the Google process for identifying patent metadata (including references and patent citations) preclude this type of analysis.

[1] http://www.epo.org/applying/basics.html

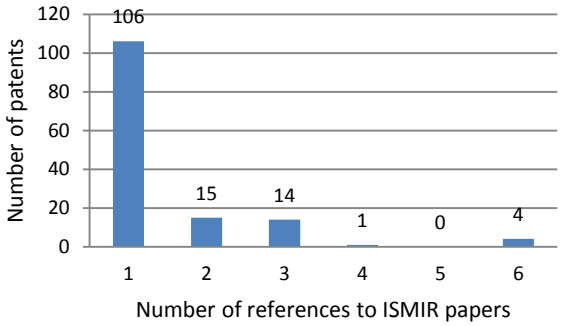


Figure 4. Number of ISMIR reference per patent

4.1.5 Topics of the Patents

In order to get an overview of the patent topics, we manually identified and categorized the main topics represented by each patent. Table 3 shows the top 10 most common topics represented in the patents analyzed.

Topics	Number of patents
Audio fingerprinting	14
Identification of similar songs	12
Music recommendation	9
Automatic playlist generation	9
Audio/Music analysis	6
Music player/interface	6
Music search and display	5
Music visualization	5
Music classification	5
Music retrieval methods	4

Table 3. Top 10 patent topics

The most common topic was audio fingerprinting; 14 patents dealt with various technologies related to audio fingerprinting or thumbprinting. This is followed by 12 patents about using audio similarity algorithms to identify songs similar to a sample song from a music collection. Music recommendation and automatic playlist generation were also popular topics. This is probably related to the increase in the popularity of streaming services and emergence of new types of services such as music identification as all of these technologies are commonly used in popular music services [1], for instance, music streaming services such as Pandora and Spotify, and music identification services such as Shazam or Soundhound. Topics such as music analysis, search, display, classification, and retrieval methods that are important components of music digital libraries/applications also appeared multiple times in the patents. Different techniques for music display and visualization were also found multiple times. Some examples of other topics that appeared two times include: music metadata, composition, audio encoding/decoding, associating music and geographic information, and social ratings.

4.2 Analysis of ISMIR Publications Cited in Patents

In this section, we examine the ISMIR publications cited in the patents: specifically, the distribution of citations from the ISMIR conference series, the ISMIR publications most frequently referenced in the patent dataset, and the overlap between ISMIR authors and inventors.

4.2.1 Year of ISMIR Publications

Figure 5 shows the year of publication of ISMIR papers referenced in the patent dataset. The most striking aspect of this figure is the peak at 2002; just over a (34.9%) of the ISMIR papers referenced were drawn from the 2002 ISMIR conference proceedings. This may be due to the fact that topics related to content-based retrieval (e.g., audio music similarity, automatic generation of playlists) started to become quite popular around that time.

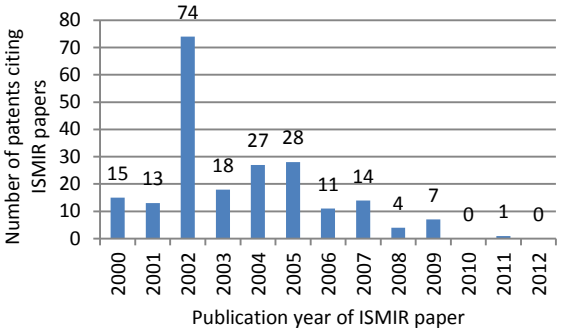


Figure 5. Number of patents citing ISMIR publications, by publication year of ISMIR paper

4.2.2 Most Highly Cited ISMIR Publications in Patents

Table 4 lists the top 10 most highly cited ISMIR publications in the patents we analyzed, sorted by the number of times cited. These publications date primarily from the early years of the ISMIR conference series; this skew is to be expected, given that the older publications have more time to accumulate citations and given the distribution of ISMIR-referencing patent filings (Figure 1).

Authors (pub. year)	Title	Freq
Haitsma J, Kalker T (2003)	A highly robust audio fingerprinting system	21
Cano P, Kaltenbrunner M, Gouyon F, Batlle E (2002)	On the use of FastMap for audio retrieval and browsing	10
Logan B (2002)	Content-based playlist generation: exploratory experiments	9

McKinney MF, Moeland D (2004)	Extracting the perceptual tempo from music	9
Pauws S, Eggen B (2002)	PATS: realization and user evaluation of an automatic playlist generator	9
Aucouturier J-J, Pachet F (2002)	Music similarity measures: What's the use?	7
Pampalk E, Flexer A, Widmer G (2005)	Improvements of audio-based music similarity and genre classification	5
Logan B (2000)	Mel frequency cepstral coefficients for music modeling	5
Berenzweig A, Logan B, Ellis DPW, Whitman B (2004)	A large-scale evaluation of acoustic and subjective music similarity measures	4
Liu D, Lu L (2003)	Automatic mood detection from acoustic music data	4
Tzanetakis G, Essl G, Cook P (2002)	Automatic musical genre classification of audio signals	4
West K, Cox S (2005)	Finding an optimal segmentation for audio genre classification	4
Paulus J, Klapuri A (2002)	Measuring the similarity of rhythmic patterns	4
Oliver N, Kreger-Stickles L (2006)	PAPA: Physiology And Purpose-Aware automatic playlist generation	4

Table 4. Ten most highly cited ISMIR publications by patents

Four of the papers in Table 4 are also listed among the most highly cited ISMIR papers in a 2009 informetric analysis of the ISMIR conference series [1]: specifically, Aucouturier & Pachet (2002), *Music similarity measures: What's the use?*; Logan (2000), *Mel frequency cepstral coefficients for music modeling*; Tzanetakis et al (2002), *Automatic musical genre classification of audio signals*; and Paulus & Klapuri (2002), *Measuring the similarity of rhythmic patterns*.

4.2.3 Inventors Who Publish in ISMIR

A total of 49 inventors have published in ISMIR proceedings; Table 5 presents a list of most prolific ISMIR authors among the the inventors. Masataka Goto (National Institute of Advanced Industrial Science and Technology) emerged as one of the key figures for connecting the academic research and commercial developments, followed by Brain Whitman (The Echo Nest) and Malcolm Slaney (Microsoft).

Author	No. of ISMIR publications	No. of patents invented
Masataka Goto	29	5
Daniel P. W. Ellis	28	1
Elias Pampalk	15	1
Francois Pachet	15	1
Tim Pohle	14	2
Hiroshi Okuno	13	2
Kazuyoshi Yoshii	12	2
Dominik Schnitzer	10	1
Douglas Eck	9	1
Mitsunori Ogihara	9	1
Paul B. Lamere	8	1
Brian A. Whitman	7	5
Malcolm Slaney	7	3
Kristopher C. West	7	2
Josep-Lluis Arcos	6	1

Table 5. Top 15 inventors who are also ISMIR authors (sorted by the number of ISMIR publications)

4.3 Analysis of MIREX References in Patents

Thirteen patents were identified that referenced MIREX: 9 had been granted, and 4 were applications. Table 6 summarizes the filing dates and, for the granted patents, the year in which they were issued. Of these thirteen, six also referenced ISMIR papers. A significant degree of overlap could be expected, given that the research of MIREX participants is also frequently published in the associated ISMIR conference. Indeed, the relationship between the MIREX and ISMIR events may explain the relatively small number of patents including MIREX references: entries to the MIREX trials are frequently accompanied by more detailed submissions regarding the algorithms to the associated ISMIR. Further, MIREX proceedings are informally published and can be difficult to locate and cite [1]—additional reasons why an ISMIR paper might be cited in preference to a similar MIREX publication.

Year	No. of Applications	No. Issued
2006	3	
2007	5	
2008	2	1
2009	1	
2010	2	5
2011		1
2012		2

Table 6. Summary by year of MIREX patents

These thirteen patents included 24 references to MIREX: 21 references to 11 unique papers in the MIREX proceedings, 2 references to an ISMIR paper providing an overview of MIREX results, and 3 more general references to the MIREX trials as a whole. These latter point to the significance of algorithms by relevance to their representation and relative performance in the MIREX trials (e.g., "…systems using this technique regularly rank in

the very top places in the yearly MIREX Automatic Music Recommendation evaluations…" [1]).

The thirteen patents had 22 unique inventors (one inventor was named in two patents). Of these 22 inventors, 11 had published in at least one ISMIR conference (See Figure 6, MIREX inventors publishing in ISMIR by year)—yielding an average of 2.91 inventors who cited MIREX in attendance at each ISMIR, 2002 – 2012.

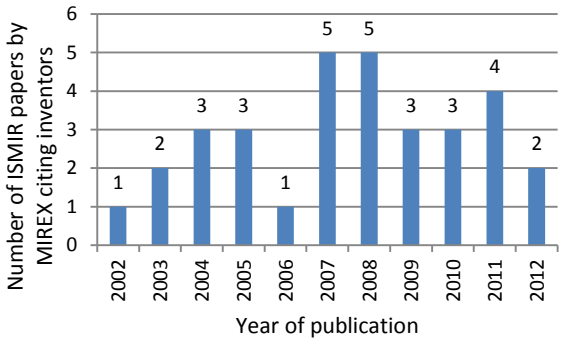


Figure 6. MIREX inventors publishing in ISMIR by year

5. CONCLUSION AND FUTURE WORK

In this paper, we have examined the influence of academic MIR research from the ISMIR conference series and MIREX on patents, as viewed through citation links from patents to the academic publications. We identified over hundred references to ISMIR and MIREX research in the patents, most prominently in early 2000s, and concerning various content-based retrieval technologies such as audio fingerprinting, recommendations, automatic playlist generation, and so on. The investigation indicates the presence of strong and ongoing personal links between academic and commercial MIR research, as evidenced by the number of individuals who produce both academic publications and patents in MIR (Section 4.2.3). It is also encouraging to see significant proportions of the ISMIR/MIREX references in patents not filed by inventors with direct connections to ISMIR (Sections 4.2.1, 4.2.3).

Ironically, it would not be straightforward to investigate the opposite flow of influence—ISMIR/MIREX publications citing patents—because the ISMIR and MIREX events are not associated with the primary computer science and engineering professional societies and consequently are not included in the societies' digital library portals (i.e., the ACM Digital Library and IEEE Xplore). Further, the MIREX papers are incompletely represented in Google Scholar [1], and both MIREX and ISMIR are inconsistently indexed [1], [3]—to the extent that it is not possible to be assured of complete coverage of ISMIR and MIREX publications through Google Scholar searches. Two digital libraries have been developed and maintained by individuals within the MIR community to provide improved access to ISMIR and MIREX publications: Michael Fingerhut's ISMIR net (http://www.ismir.net/) for ISMIR publications, and David Bainbridge's library (http://music-ir.org/mirex-dl/library) for MIREX. However, it lacks full-text search capabilities, and the latter does provide complete, standardized metadata—and so their utility for informetric investigations is reduced. More seriously, the visibility of the ISMIR and MIREX series as a whole is diminished.

In our future work, we plan to conduct a topic analysis of ISMIR publications, to identify shifts in focus over the conference series and for comparison to topics identified in the patents (Section 4.1.3). Also further investigation is required to tease out the factors contributing to the decline in the number of patents referencing ISMIR. One possibility is that the research interests of the academic and commercial communities have indeed diverged (though the overlap between academic publishers and inventors argues against this). Ultimately, we hope to expand our search and identify the patents citing any publications related to MIR in multiple publication venues, not limited to ISMIR conference proceedings. This may help reveal a direction for new research that can make strong impact in the everyday life of music users.

6. REFERENCES

[1] S. J. Cunningham, D. Bainbridge, and J. S. Downie: "The impact of MIREX on scholarly research," *Proc. of the ISMIR*, pp. 259-264, 2012.

[2] J. Futrelle, J. S. Downie: "Interdisciplinary Communities and Research Issues in Music Information Retrieval," *Proc. of the ISMIR*, pp. 215-221, 2002.

[3] J. H. Lee, M. C. Jones, J. S. Downie: "An analysis of ISMIR proceedings: Patterns of authorship, topic, and citation," *Proc. of the ISMIR*, pp. 57-62, 2009.

[4] J. H. Lee and N. M. Waterman: "Understanding user requirements for music information services," *Proc. of the ISMIR*, pp. 253–258, 2012.

[5] A. Klitkou and M. Gulbrandsen: "The relationship between academic patenting and scientific publishing in Norway", *Scientometrics*, Vol. 82, pp. 93-108, 2010.

[6] M. Meyer: "Patent citations in a novel field of technology—what can they tell about interactions between emerging communities of science and technology?" *Scientometrics,* Vol. 48, No. 2, pp. 151-178, 2000.

[7] M. Meyer: "What is special about patent citations? Differences between scientific and patent citations," *Scientometrics*, Vol. 49, No. 1, pp. 93-123, 2000.

[8] F. Narin: "Patent bibliometrics," *Scientometrics*, Vol. 30, pp. 147–155, 1994.

[9] D. Schnitzer: "A method and a system for identifying similar audio tracks." European Patent No. EP 2273384. 12 Jan. 2011.

TOWARD UNDERSTANDING EXPRESSIVE PERCUSSION THROUGH CONTENT BASED ANALYSIS

Matthew Prockup, Erik M. Schmidt, Jeffrey Scott, and Youngmoo E. Kim
Music and Entertainment Technology Laboratory (MET-lab)
Electrical and Computer Engineering, Drexel University
{mprockup,eschmidt,jjscott,ykim}@drexel.edu

ABSTRACT

Musical expression is the creative nuance through which a musician conveys emotion and connects with a listener. In un-pitched percussion instruments, these nuances are a very important component of performance. In this work, we present a system that seeks to classify different expressive articulation techniques independent of percussion instrument. One use of this system is to enhance the organization of large percussion sample libraries, which can be cumbersome and daunting to navigate. This work is also a necessary first step towards understanding musical expression as it relates to percussion performance. The ability to classify expressive techniques can lead to the development of models that learn the the functionality of articulations in patterns, as well as how certain performers use them to communicate their ideas and define their musical style. Additionally, in working towards understanding expressive percussion, we introduce a publicly available dataset of articulations recorded from a standard four piece drum kit that captures the instrument's expressive range.

1. INTRODUCTION

In music, it is the human component of expression that imparts emotion and feeling within a listener. Expression relates to the nuances in technique that a human performer imparts on a piece of music. Musicians creatively vary timing, dynamics, and timbre of the musical performance, independent from the score, in order to communicate something of deeper meaning to the listener [1]. For example, a musician can alter tempo or change dynamics slightly to impart tension or comfort. Similarly, they can alter the timbre of their instrument to create different tonal colors. All of these parameters add an additional level of intrigue to the written pitches, rhythms, and dynamics being performed.

In studying percussion, one of the fundamental ways of communicating a musical idea is through expressive *articulation*. Differences in articulation are created by the creative combination of dynamics and excitation timbre. This simple relationship is outlined in Figure 1. There are an almost infinite number of ways that a percussionist can strike a drum. While the strike itself is restricted to being a single discrete event, there exists a vast range of articulations that make each of those seemingly discrete actions sit in a continuous and highly dimensional space.

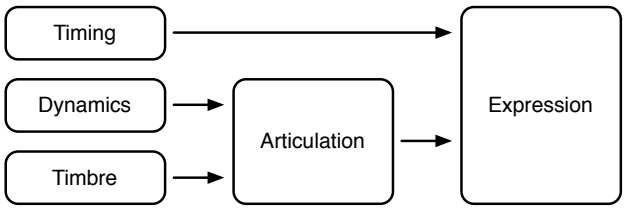


Figure 1. Expression: Creative alterations in timing, dynamics, and instrument timbre can define a musician's expressive style.

In percussion, there are four main techniques of excitation: strikes, rim shots, cross sticks, and buzz strokes. An explanation of these techniques is outlined in Table 1. This simple set of excitation techniques become the building blocks of the standard *rudiments* that define most aspects of percussion music [2]. Each expressive articulation has meaning in the context of a rudiment, and many individual performers have unique ways of expressing and combining them. This defines their style and identity as a musician. In this initial work, we seek to quantify and understand differences in excitation techniques. It is important in the context of percussion that a bottom up approach be taken to expressive performance analysis. Percussion performance is built on the rudimentary combination of unique articulations, so this is a logical place to start. In the music information retrieval community, it has been a large aspect of percussion performance and expression that has been ignored.

In working towards this understanding of expressive percussion, we have compiled a comprehensive new public dataset of expressive samples recorded from a standard four piece drum kit. The dataset includes samples varied by intensity of stroke (staccato vs legato), height of stroke, and strike position over a variety of excitation techniques for each instrument of the drum kit. Using this dataset we train a simple four class support vector machine (SVM) to distinguish these expressive articulations both depen-

Articulation	Description
Strike	The drumhead is struck with the tip of the stick.
Rim Shot	Both the drumhead and rim are struck with the tip and shaft of the stick simultaneously.
Cross Stick	The butt of the stick strikes the rim while the tip rests on the head.
Buzz Stroke	The stick is pressed into the drum to create multiple, rapid strokes.

Table 1. Excitation Techniques: There are four basic drum excitation techniques.

dent on and independent of percussion instrument type. In the context of this paper, we will investigate the three drums commonly struck with sticks (snare drum, rack tom, and floor tom) and the four excitations that become the building blocks of rudiments. Excitation classification is only a small aspect of percussion expression, but the ability to recognize these differences in articulation is a necessary first step in understanding percussion performance as a whole.

2. BACKGROUND

There are a few areas of research tangentially related to expressive percussion performance. The first and most widely studied is the task of instrument identification. Earlier studies in instrument recognition have focused mainly on the ability to classify a wide range of traditional instrument tones, but more recently, a greater effort has been made to classify instruments specific to the realm of percussion. In [3], a set of systems using a wide range of feature selection and classification techniques performed well at discriminating percussion instruments. However, this study only took into account a standard drum strike and purposely did not include alternative articulations, such as rim shots or buzz strokes.

Some studies take the instrument identification approach a step further and attempt to transcribe drum patterns. One such transcription study presented in [4] used non-negative spectrogram factorization and onset detection techniques in order to separate drum sounds and classify them as either a snare drum, bass drum, or hi-hat. This shows promise in the ability to retrieve drum sounds directly from patterns. In [5], Battenberg and Wessel used deep learning approaches in order to learn beat sequence timings of the snare drum, bass drum, and hi-hat in different drum patterns. Understanding a drum's context within a performance can lead to models that can inform musical style. This was a step in the right direction for the analysis of percussion expression.

There has also been an evolving volume of work studying musical performance analysis and expression specifically. Mion and Poli in [1] stated that musical expression is best represented with score independent descriptors that model intricacies in timing, dynamics, and timbre. They showed that a simple set of features can be used to capture and classify the expressive intent of a performer in both affective and sensorial domains. Other work in music expression focuses on the intricacies of specific instruments. In [6], an analysis-by-synthesis experiment was performed to model, synthesize, and evaluate the expressive characteristics of a clarinet performance. The authors identified feature dynamics that relate to expressive performance. They then forced the dynamic features to be static, creating a less expressive re-synthesis. A listening test was then performed which asked if subjects preferred the original or altered recordings. Results from the test showed that listeners preferred the original musically expressive performance. It also showed that expression is captured in the evolution of features over time, and removing this aspect effectively removes musical expression. This demonstrated that the dynamic nature of instrument timbre is an important aspect of music expression. In order to capture feature dynamics, simple polynomial expressions can be fit to the time varying process. This provides a compact representation of sequential data in both the time and frequency domains [7].

A vast majority of prior work in musical expression analysis has revolved around understanding the timbral characteristics of pitched instruments. A detailed analysis of expressive percussion is also necessary, yet it is largely ignored. However, some sparse examples of these studies do exist. The work in [8] focuses on snare drum expression and attempts to distinguish playing position on the head as well as excitation techniques, such as using brushes or playing a rim shot. These experiments, however, were very limited in scope, with models being only applicable to one drum. Additionally, all training and testing examples were performed at a single volume and intensity level.

In this paper, we perform the task of percussion articulation classification similar to the work found in [8]. In our study however, it is important for the models to generalize over multiple pieces of the drum kit. Secondly, our models incorporate additional excitation techniques (buzz strokes and cross stick strokes) as well as a dataset containing many different ways of performing these articulations. Using compact representations of timbral characteristics over time, we train classifiers to distinguish excitation techniques independent of drum, stick height, intensity of stroke, and head strike position.

3. DATASET OF EXPRESSIVE PERCUSSION

In domains outside of percussion, there exist large datasets that can be used for expressive performance analysis. A comprehensive, well-labeled set of expressive percussion samples is less common. The presented work makes use of a newly recorded dataset that encompasses a vast array of percussion performance expressions on a standard four piece drum kit. In the context of this paper, only the snare drum, rack tom, and floor tom samples are used. Each drum used has samples that span the following range:

- stick heights: 8cm, 16cm, 24cm, and 32cm
- stroke intensities: light, medium, heavy

Feature Names	Feature Abbreviation	Feature Description	Source
RMS energy	RMS	root-mean-squared energy	n/a
roughness	R	energy of beating frequencies	[9]
brightness	B	description of spectral brightness	[9]
2 bin ratio (bottom half)	SRA	ratio of spectral energy below 1000Hz to the full spectrum	[1]
3 bin ratio (low)	SRL	ratio of spectral energy below 534Hz to the full spectrum	[1]
3 bin ratio (med)	SRM	ratio of spectral energy between 534Hz and 1805Hz to the full spectrum	[1]
3 bin ratio (high)	SRH	ratio of spectral energy above 1805Hz to the full spectrum	[1]

Table 2. Basic Features: Single dimensional time and frequency domain features are used as the basis for the evolution features.

- strike positions: center, halfway, edge
- articulations: strike, rim shot, buzz stroke, cross stick

This subset includes 1804 individual examples across the four articulations over the three drums. Additionally, there are at least 4 examples of each expressive combination. Recordings include samples with the snare wires both touching (snares on) and not touching (snares off) the bottom head of the snare drum. The division of sample variety is not completely uniform across the entire set, but it was designed to allow for the most complete coverage of each instrument's expressive range. That being said, no one combination of expressive parameters vastly outweighs another and all are adequately represented.

The full dataset also includes a complete array of expressive bass drum, hi-hat, and cymbal samples as well. Each articulation example has monophonic and stereo versions with multiple mixes using direct (attached) and indirect (room) microphone positioning techniques. This is the first publication where this dataset appears and it can be made freely available to others upon request.

4. PREDICTING EXPRESSIVE ARTICULATION

In expressive performance, the evolution of timbre over time is an important component on both a micro and macro level. This work investigates expression at the micro level by attempting to model the evolution of percussion articulations. Using the sequential evolution of features derived from time domain and frequency domain components of the signal, a set of classifiers is trained to predict percussion articulations within subsets containing only individual drums (only snare, only rack tom, etc.) as well as within the superset of all drum samples.

4.1 Feature Design

The aural differences in percussion articulations are defined by the short time evolution of their spectral components. For example, a buzz stroke evolves very differently than a rim shot. These differences are apparent in both their time domain and frequency domain characteristics. In order to capture this evolution, a set of compact features was implemented that model the envelope of single dimensional features over time. This compact representation is derived from the coefficients of a polynomial fit to the time varying feature data similar to [7]. This compact polynomial representation was calculated for the features outlined in Table 2. Descriptions of the new polynomial coefficient features are described in Table 3.

Feature Names	Feature Description
RMS_3 RMS_6	3^{rd} and 6^{th} order coefficients of RMS
R_3 R_6	3^{rd} and 6^{th} coefficients of R
B_3 B_6	3^{rd} and 6^{th} coefficients of B
SR_3 SR_6	3^{rd} and 6^{th} aggregated coefficients of SRA, SRL, SRM, and SRH

Table 3. Evolution Features: New features are derived from the coefficients of polynomials fit to the the single dimensional features in Table 2 over time.

Figure 2 shows the time evolution of selected features and their polynomial representations for a snare drum across each of the articulation examples. It is easy to qualitatively discriminate the differences in shape for each of the articulations. Polynomials fit to the feature data are able to capture this shape in a compact manner. It was found in early experimentation that the third and sixth degree polynomial fits were optimal for representation. In order to evaluate the salience of these newly implemented features, Mel-Frequency Cepstral Coefficients (MFCCs) and their first and second derivatives were also used in the classification tasks for comparison.

4.2 Experiments

The main focus of the work presented is to classify the excitation techniques of expressive drum strike articulations. The articulations observed and their descriptions are shown in Table 1. Using the polynomial coefficient features from Table 3, a four class support vector machine (SVM) using a radial basis function (RBF) kernel was trained to discriminate excitation. In all experiments, five-fold cross validation was performed for both parameter tuning and training/testing. The classification task was run for each drum individually as well as for all drums in combination. This tested the effectiveness of the system to understand expression on individual drums as well as throughout the entire drum kit. For example, in a robust system a rim shot should be classified as such regardless of the instrument on which it was performed. In order to compare the effectiveness of each of the new features, the classification task was also performed using the means of the MFCCs and their first and second derivatives over the duration of the sample.

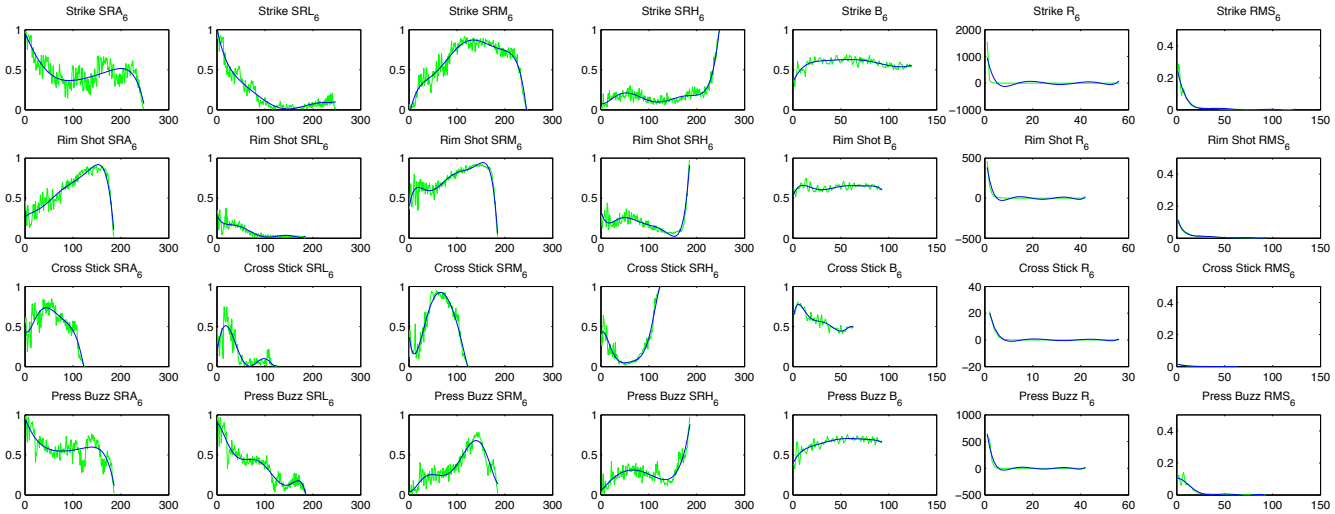


Figure 2. Feature Evolution Example: Sixth order polynomials are fit to the temporal feature data of four snare drum articulations.

The first experiment involved classifying excitation on the each drum individually. Features were used both alone and in aggregation. In order to aggregate the features, each dimension was normalized to have zero mean and unit variance. The testing data was transformed using the mean and variance derived from the training data. This allowed each feature to be simply concatenated for training and testing. The raw features and projections via a principal components analysis (PCA) were also explored, but in practice, the simple normalization transformation yielded the best results. The second experiment classified excitation over the set of all drum samples. Again, the features were used both individually and in aggregation with the simple normalization. In both experiments, the new features and their combinations were also used in conjunction with MFCCs. This MFCC aggregation shows their ability to add time domain information to an already salient, yet static, feature and improve its performance.

4.3 Results

The first experiment classifies excitation for each drum independently using the features individually as well as in selected aggregations. Table 4 shows the accuracies for the features individually. MFCCs averaged over the example are the best single performing feature for both the snare drum and rack tom. The floor tom, however, shows better performance with the 3rd and 6th order polynomial coefficients of the spectral ratios (SR_3 SR_6) than it does with the MFCCs. While standard MFCCs do not take into account any information about time evolution, each articulation does have an inherently different average timbre. Because MFCCs are designed to provide an estimate of the spectral envelope and capture this timbre, they perform reasonably well. However, when the samples have a greater length and therefore a longer timbre evolution, such as that of a floor tom, MFCCs start to degrade in performance while some of the evolution features start to improve.

Individual Feature	Snare	Rack Tom	Floor Tom
MFCC	$\mathbf{0.956 \pm 0.012}$	$\mathbf{0.914 \pm 0.037}$	0.872 ± 0.027
Δ MFCC	0.771 ± 0.015	0.661 ± 0.029	0.834 ± 0.015
Δ^2 MFCC	0.646 ± 0.031	0.544 ± 0.061	0.637 ± 0.020
SR_3	0.897 ± 0.016	0.835 ± 0.017	$\mathbf{0.907 \pm 0.045}$
SR_6	0.776 ± 0.023	0.838 ± 0.033	0.896 ± 0.025
B_3	0.736 ± 0.032	0.846 ± 0.031	0.859 ± 0.036
B_6	0.713 ± 0.013	0.755 ± 0.032	0.845 ± 0.009
R_3	0.407 ± 0.017	0.670 ± 0.017	0.523 ± 0.022
R_6	0.514 ± 0.021	0.822 ± 0.017	0.578 ± 0.050
RMS_3	0.637 ± 0.013	0.696 ± 0.039	0.795 ± 0.039
RMS_6	0.773 ± 0.014	0.893 ± 0.030	0.845 ± 0.018

Table 4. Classification Accuracies: Excitation techniques were classified using each feature on each drum individually.

Table 5 shows the performance of features in combination on the individual drums. The feature combinations with the highest classification accuracies for each drum are displayed along with the best performing individual features for comparison. In all cases, the aggregated feature combinations had a higher classification accuracy than each of the best performing individual features. This shows that combining an estimation of general timbre with certain features that capture that timbre's evolution can improve classification accuracy. In Table 5 only the top five performing feature combination accuracies for each drum are shown. Those that appear in multiple lists show they are better at generalizing over the different drum types. The 6^{th} order brightness feature in combination with MFCCs (B_6 MFCC) was the only aggregation to appear within the top five best performing combinations over all three drum types.

In the second experiment, a single classifier was trained on articulation samples from all three drums. The classifiers were again trained on each feature individually and in combination. The accuracies for the classification of percussion articulations, independent of drum, are shown in Table 6. In the classification of excitation over the superset

Feature Aggregation	Snare Drum	Rack Tom	Floor Tom
SR_3 R_3 B_3 MFCC	**0.987 ± 0.001**	-	0.982 ± 0.010
SR_3 B_3 MFCC	0.982 ± 0.005	-	0.972 ± 0.007
B_3 MFCC	0.982 ± 0.007	0.956 ± 0.013	-
B_6 MFCC	0.978 ± 0.005	0.963 ± 0.015	0.974 ± 0.019
SR_3 R_3 MFCC	0.977 ± 0.012	-	-
SR_6 R_6 B_6 MFCC	-	**0.9712 ± 0.009**	0.982 ± 0.014
SR_6MFCC	-	0.955 ± 0.020	-
R_6 MFCC	-	0.955 ± 0.012	-
SR_6 B_6 MFCC	-	-	**0.984 ±0.005**
Best Individual	0.956 ± 0.012 (MFCC)	0.914 ± 0.037 (MFCC)	0.907 ± 0.045 (SR_3)

Table 5. Classification Accuracies: Excitation techniques were classified using selected feature aggregations on each drum individually. Results are shown for the top five performing features on each drum. Feature combinations that are outside the top five best performing aggregations for a single drum type are marked with '-'.

of all drums, MFCCs were shown to be the best performing feature. However, when the polynomial envelope features were used in combination with MFCCs, accuracy was again improved. The 6^{th} order brightness feature in combination with MFCCs (B_6 MFCC) was the best performing feature for over the superset of all drums. This is likely due to the fact that this combination was also the only one contained within the top performing combinations of all individual experiments from Table 5.

Feature	All Drums
MFCC	**0.930 ± 0.011**
Δ MFCC	0.745 ± 0.021
Δ^2 MFCC	0.534 ± 0.016
SR_3	0.847 ± 0.010
SR_6	0.744 ± 0.020
B_3	0.734 ± 0.024
B_6	0.719 ± 0.018
R_3	0.498 ± 0.020
R_6	0.514 ± 0.006
RMS_3	0.731 ± 0.017
RMS_6	0.590 ± 0.008
B_6 MFCC	**0.972 ± 0.004**
SR_3 R_3 B_3 MFCC	0.969 ± 0.011
SR_6 B_6 MFCC	0.967 ± 0.008
SR_6 R_6 B_6 MFCC	0.965 ± 0.006
SR_3 B_3 MFCC	0.963 ± 0.004

Table 6. Classification Accuracies: Excitation techniques were classified using features individually and in aggregation over the superset of all drum types.

In all cases, for each individual drum and the superset of all drums, MFCCs performed rather well on their own. However, they do not take into account any information regarding the temporal evolution of the signal. The derivatives of MFCCs were also used, but they provide only a static picture of the amount of change present when averaged over the example. They still lack information as to how those changes evolve. Additionally, in the presented experiments, MFCCs were shown to be better at modeling articulations than were their derivatives. However, by using the polynomial coefficients of simple time varying features along with standard MFCCs, the system was able to gain temporal context, leading to better performance.

5. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, it was shown that the coefficients of polynomials fit to model feature evolution can provide a compact representation with the ability to quantify percussive articulation. These features in conjunction with popular features, such as MFCCs, can improve performance by adding temporal context. In this paper, we also introduced a new comprehensive dataset of expressive percussion articulations. This presented work only scratches the surface of this dataset's applicability to problems involving expression in musical performance. Classifying articulation is a small, yet very necessary step in the understanding of percussion performance and expression in general. Moving forward, more work must be done towards understanding the micro and macro evolution of expression.

On the micro level, this work can be expanded upon by using more sophisticated systems to improve the modeling of feature evolution. It was shown in [10] that linear dynamical systems (LDS) are a compact way of representing and synthesizing pitched percussive instrument tones. This introduces the possibility of training an LDS for each articulation example and training a classifier that uses system parameters as features. Secondly, an LDS is a generative model, so it may also be possible generate or alter learned sets of percussive articulation. Understanding this micro evolution can greatly assist in the navigation and organization of large humanly expressive sample libraries, which are usually cumbersome for percussion instruments.

In future work, we look to model not only the micro evolution, but the macro evolution of expression as well. If we are able to classify percussion articulations, we can look further into its meaning by developing models that learn the functionality of articulation in patterns and performance. The articulation classification along with statistics of their usage, dynamics, and time onsets can lead to models that contain information about human playing style. This performance style can be used to model individual percussionists or larger populations of similar percussionists. With these performance models in conjunction with the ability to classify articulation, we can investigate the possibility of expressive performance generation using unlabeled sets of any custom sample library that a producer or composer wishes to use. This may seem like a lofty goal in relation to this work's present state, and in most respects, it is. However, expressive articulation is one of the most important parameters of a percussionist's performance. The ability to classify expressive excitation, independent of percussion instrument, is the necessary first step towards understanding the unique intricacies and nuances of percussion performance and its relation to human expression in general.

6. ACKNOWLEDGMENTS

The authors would like to thank the Music Industry Department of Drexel University's College of Media Arts and Design for their support and assistance in the recording, mixing and organization of the expressive percussion sam-

ple library. It was with their help that we were able to create a comprehensive, high quality, labeled audio dataset of expressive percussion.

7. REFERENCES

[1] L. Mion and G. D. Poli, "Score-independent audio features for description of music expression," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, no. 2, pp. 458–466, 2008.

[2] M. Goldenberg, *Modern school for snare drum*. Hal Leonard, 1955.

[3] P. Herrera, A. Yeterian, and F. Gouyon, "Automatic classification of drum sounds: A comparison of feature selection methods and classification techniques," in *Music and Artificial Intelligence*, vol. 2445 of *Lecture Notes in Computer Science*, pp. 69–80, Springer Berlin Heidelberg, 2002.

[4] J. Paulus and T. Virtanen, "Drum transcription with non-negative spectrogram factorisation," in *Proceedings of the 13th European Signal Processing Conference*, p. 4, 2005.

[5] E. Battenberg and D. Wessel, "Analyzing drum patterns using conditional deep belief networks," in *Proceedings of the International Conference on Music Information Retrieval*, 2012.

[6] M. Barthet, P. Depalle, R. Kronland-Martinet, and S. Ystad, "Analysis-by-synthesis of timbre, timing, and dynamics in expressive clarinet performance," *Music Perception*, vol. 28, no. 3, pp. 265–278, 2011.

[7] M. Lagrange, M. Raspaud, R. Badeau, and G. Richard, "Explicit modeling of temporal dynamics within musical signals for acoustical unit similarity," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1498–1506, 2010.

[8] A. Tindale, A. Kapur, G. Tzanetakis, and I. Fujinaga, "Retrieval of percussion gestures using timbre classification techniques," in *Proceedings of the International Conference on Music Information Retrieval*, pp. 541–544, 2004.

[9] O. Lartillot and P. Toiviainen, "A matlab toolbox for musical feature extraction from audio," in *International Conference on Digital Audio Effects*, pp. 237–244, 2007.

[10] E. M. Schmidt, R. V. Migneco, J. J. Scott, and Y. E. Kim, "Modeling musical instrument tones as dynamic textures," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2011 IEEE Workshop on*, pp. 329–332, IEEE, 2011.

DATA DRIVEN AND DISCRIMINATIVE PROJECTIONS FOR LARGE-SCALE COVER SONG IDENTIFICATION

Eric J. Humphrey, Oriol Nieto, Juan P. Bello
Music and Audio Research Laboratory
New York University
{ejhumphrey, oriol, jpbello}@nyu.edu

ABSTRACT

The predominant approach to computing document similarity in web scale applications proceeds by encoding task-specific invariance in a vectorized representation, such that the relationship between items can be computed efficiently by a simple scoring function, e.g. Euclidean distance. Here, we improve upon previous work in large-scale cover song identification by using data-driven projections at different time-scales to capture local features and embed summary vectors into a semantically organized space. We achieve this by projecting 2D-Fourier Magnitude Coefficients (2D-FMCs) of beat-chroma patches into a sparse, high dimensional representation which, due to the shift invariance properties of the Fourier Transform, is similar in principle to convolutional sparse coding. After aggregating these local beat-chroma projections, we apply supervised dimensionality reduction to recover an embedding where distance is useful for cover song retrieval. Evaluating on the Million Song Dataset, we find our method outperforms the current state of the art overall, but significantly so for top-k metrics, which indicate improved usability.

1. INTRODUCTION

Cover song identification is a well-established task in the MIR community, motivated by both theoretical and practical interest. On one hand, a "cover" is an abstract form of musical variation and presents a challenging computer audition problem. Alternatively, music collections continue to expand to unprecedented volumes, particularly in terms of amateur and user-generated content. As evidenced by even a brief review of websites like YouTube [1], Vimeo [2], or Soundcloud [3], a considerable portion of online musical content now consists of covers.

In light of this, previous research in cover song identification explores a variety of approaches, including the cross-correlation of beat-synchronous chroma features [4], dynamic time warping on binary chroma similarities [10], cross-recurrence quantification [11], etc. For a comprehensive review, the reader is referred to [9]. Over time, it has been shown that these methods can achieve robustness to specific kinds of musical variation, e.g., tempo changes, differences in structure, or key transpositions. In practice however, making use of these non-trivial operations yields complex systems that are computationally prohibitive to evaluate, let alone deploy, on large music databases.

Recognizing this limitation, recent work in cover song retrieval explores a slightly different approach to the task [1]. Rather than attempting to resolve irrelevant musical variation in the process of comparing two tracks, this particular system tries to encode this invariance directly with a multi-stage, feed-forward architecture. Local beat-chroma patterns are efficiently transformed into shift-invariant features via the 2D-Fourier Transform, median-pooled over time into a summary representation, and projected into a PCA subspace. Having transformed a collection of tracks into a much lower dimensional space, pairwise comparisons can be efficiently computed by Euclidean distance. As a result, this approach scales well to large collections like the Million Song Dataset [4] (MSD), and offers a promising research direction for pursuing general web-scale music similarity.

Here, we seek to advance this initial work by improving the feed-forward architecture to yield better representations for cover song retrieval. After fine-tuning the previously developed system, we propose two major modifications: sparse, high-dimensional data driven component estimation to improve separability, and supervised dimensionality reduction to recover a cover-similarity space. Our initial analysis on a training subset shows how the combination of sparse projections and supervised embeddings can lead to better organized spaces and improve cover song retrieval. Interestingly, evaluating on the MSD results in two notable findings: one, that our approach significantly improves performance at top-k metrics; and two, though our supervised embedding can be prone to over-fitting, PCA subspaces help alleviate this issue.

The remainder of this paper is organized as follows. Section 2 formally motivates and introduces the approach in [2] upon which our work is based, while Section 3 de-

[1] http://youtube.com
[2] http://vimeo.com
[3] http://soundcloud.com

[4] http://labrosa.ee.columbia.edu/millionsong/

tails our proposed modifications. In Section 4 we present results on the development set and give detailed analyses on the impact of each proposed modification. Section 5 discusses results on the MSD and tests strategies for minimizing the generalization error. Finally, Section 6 draws conclusions and advances a number of ideas for future work.

2. SCALABLE COVER SONG RETRIEVAL

2.1 Problem Formulation

Expressed symbolically, cover song retrieval proceeds by determining the relationship $S_{i,j}$ between a query A_i and reference track B_j via the composite of feature extraction f and a pairwise comparison [5] function g:

$$S_{i,j} = g(f(A_i), f(B_j)) \tag{1}$$

Note then that computing the full comparison matrix S between a set of Q queries against a collection of R reference tracks requires a double-for loop, and the total computational cost $\mathcal{C}_S$ is expressed as $QR\bar{\mathcal{C}}_{S_{i,j}}$, where $\bar{\mathcal{C}}_{S_{i,j}}$ is the expected cost of computing a single pairwise relationship. However, when f and g are independent, feature extraction can be performed separately, and the total computational load can be re-written as follows:

$$\mathcal{C}_S = QR\bar{\mathcal{C}}_g + (Q+R)\bar{\mathcal{C}}_f \tag{2}$$

Importantly, though the average comparison cost $\bar{\mathcal{C}}_g$ scales quadratically, the start-up cost of feature extraction $\bar{\mathcal{C}}_f$ is linear. The intuition for this trick is a common optimization in software engineering —minimize the amount of computation inside for-loops— and pinpoints the fundamental deficiency of many cover song retrieval systems: comparison functions often rely on expensive operations like cross-correlation or dynamic time warping, which *must* remain inside a nested for-loop. Thus, scalable cover song retrieval necessitates choosing an efficient comparison function g; the challenge then becomes one of designing the feature extraction stage f so as to maximize the accuracy of the rankings according to S.

2.2 Relating to Previous Work

To these ends, the authors of [1] propose an astute solution to this challenge. Intuitively, cover song retrieval algorithms are designed to be invariant to time and key transpositions. One cleverly efficient way of achieving this behavior is by computing the 2-dimensional Discrete Fourier Transform (2D-DFT) of local patches of beat-synchronous chroma features and keeping only the magnitude coefficients. Whereas the phase component of the 2D-DFT encodes circular rotations in time and pitch class, 2D-Fourier Magnitude Coefficients (2D-FMC) capture these patterns regardless of absolute position. As shown by [8] in the context of rhythm analysis, the DFT is sensitive to the order of events in a sequence, where the addition of different sinusoids results in patterns of cancellations that affect the magnitude coefficients.

Describing holistically, the system presented in [1] defines f as a feed-forward embedding function that operates at multiple time scales. First, 2D-FMC are computed on a moving window of 75 beat-synchronous chroma vectors, with a 1 beat hop size. A track is then pooled over time by taking the coefficient-wise median across all 2D-FMC vectors and L_2-normalized. Having sampled a collection of summary 2D-FMC vectors, PCA is performed and used to embed tracks in a low dimensional subspace; the authors experimentally found that preserving anywhere between 50 and 200 principal components returns better results. Importantly, once tracks are embedded in this feature space via f, they define g as Euclidean distance to efficiently compute pairwise comparisons.

3. IMPROVING FEATURE EXTRACTION

Starting from the work presented in [1], we now propose a series of modifications to make the feature extraction process more robust and improve cover song retrieval, outlined in its entirety in Figure 1. First, we discuss various data pre-processing strategies, including non-linear scaling and normalization. Next, we describe our approach to data driven component estimation, addressing its motivation and conceptual parallels to recent developments in information processing strategies. Lastly, a supervised learning stage is introduced to realize an embedding where summary representations of covers are significantly closer.

3.1 Data Pre-processing

As an initial step, here we apply three operations to the 2D-FMC representation preparing it for further processing: logarithmic compression, vector normalization, and dimensionality reduction via PCA. Expressed formally, the first two are achieved by

$$\hat{X} = \log\left(\frac{CX}{\|X\|_2} + 1\right) \tag{3}$$

where C is a constant hyperparameter, X is a 2D-FMC vector, and $\|\cdot\|_2$ is the L_2-norm. We empirically observed that L_2-normalization followed by log-scaling with $C = 5$ yields slightly better results than the inverse order with $C = 100$. Intuitively speaking, while log-compression scales all coefficients independently, unit normalization adjusts the dynamic range of each vector *relative* to the other dimensions, and can be viewed as a form of adaptive gain control. This contrast adjustment turns out to be quite necessary, as certain coefficients, e.g., the DC component, are prone to dominating the overall representation and unfavorably biasing distance calculations downstream.

Finally, PCA is applied for two reasons. First, as we will see, it is important to center the representation such that each coefficient has zero mean. Additionally, we discard the redundant components of the Fourier transform, reducing the dimensionality from 900 to 450 coefficients.

[5] The choice of similarity or distance is only a matter of preference.

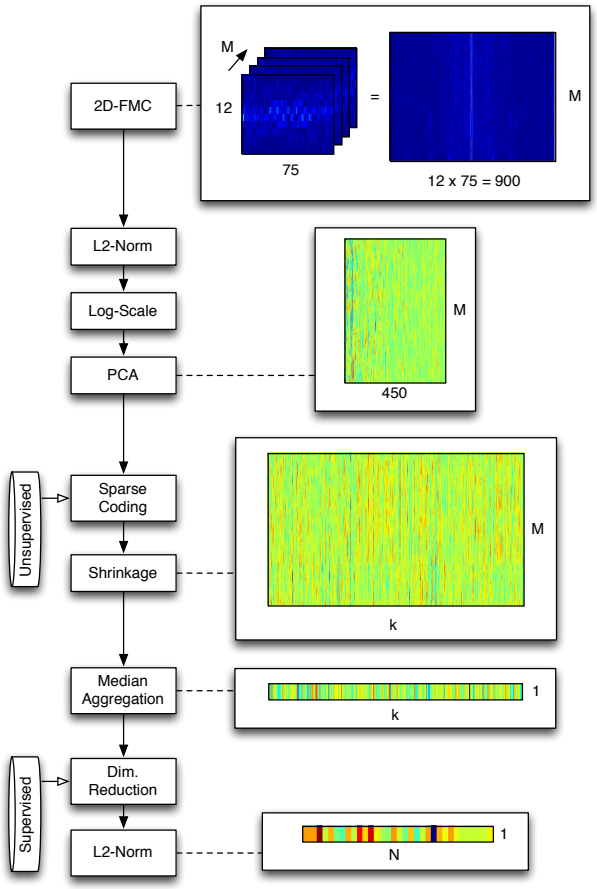


Figure 1. Diagram of the proposed method.

3.2 Sparse Component Estimation

Previous work used 2D-FMCs as a clever way to represent meaningful, rotation-invariant beat-chroma patterns. While this is, strictly speaking, an accurate insight, the Fourier bases themselves do not necessarily make for good feature extraction. Generally speaking, when the bases of a projection are unlike the data to which it is applied, it is unable to compactly represent this information. As a result, the noise floor of the resulting representation is higher and most coefficients tend to be active; furthermore, this behavior becomes especially problematic when pooling features. This observation is quite relevant to this particular instance, as the data being projected —non-negative chroma— is not sinusoidal.

Alternatively, data driven transformations learn a set of bases [6], or a *dictionary*, from a sampling of data, and are often able to encode meaningful behavior with a small number of active components. This is typically realized by the dot-product of an input X with a dictionary W, followed by an activation function $h(\cdot)$, expressed as follows:

$$Z = h(W \cdot X) \tag{4}$$

Interestingly, this formulation draws strong parallels to previous work both neural networks and sparse coding. Recent research in these areas has emphasized the importance of $h(\cdot)$ being defined as the shrinkage operator, $h_\theta(x) = sgn(x) * max(\|x\| - \theta, 0)$, where θ controls the knee or threshold of the function [5]. This non-linearity exhibits the desirable behavior of suppressing low-level activations while passing sufficiently large ones. Such a process inherently leads to sparser outputs, with the rationale being that only the most representative attributes are encoded.

Additionally, convolutional variants [7] apply this dictionary, also referred to as *kernels*, at all translations over an input to achieve shift-invariant feature extraction, given by the following:

$$Z = h(W \circledast X) \tag{5}$$

Reusing these kernels at all positions, known as *weight sharing*, results in fewer parameters to learn, reduced overfitting, and thus better generalization. Despite these advantages, convolution is a computationally expensive operation. Therefore, to reap the benefits of data driven transformations while still learning shift-invariant features, we leverage the convolution-multiplication duality of the DFT and operate on the Fourier magnitude representation:

$$Z = h(W \circledast X) = h(W \cdot \|\mathcal{F}(X)\|) \tag{6}$$

Note that it is unnecessary to also take the Fourier transform of W, as Eq (6) is now equivalent to Eq (4) and the dictionary can be learned directly on the pre-processed 2D-FMC representation. Additionally, by first centering the data, a bias term is unnecessary and both W and Z will also be approximately zero-mean.

3.3 Semantically Organizing the Space

Having designed a transform to project 2D-FMCs into a sparse representation, we subsequently pool features over a track by taking the median over each coefficient. However, while discriminative power can be achieved by projecting into higher dimensional spaces, it is often necessary to recover a lower dimensional embedding where distance encodes the desired semantic relationship between vectors, i.e. covers are near-neighbors. There are at least two conceptual justifications motivating an embedding transform. First, high dimensional representations are known to suffer from the curse of dimensionality, i.e. distance is not well behaved. Second, and more specific to this approach, the dictionary used in the previous stage is learned as an unsupervised process. As a result, there are no guarantees that the representation it produces provides the latent organization necessary for this task.

Therefore, using known relationships between songs in a training set, we can treat covers as distinct classes in a large, multi-class problem, and apply supervised learning to recover an embedding that tries to preserve these relationships. The resulting projection can then be used to transform unseen data into a cover-similarity space for computing distances between tracks.

[6] These are not strictly bases in the orthogonal, linear algebra sense, but it is a commonly used term in the literature.

4. EXPERIMENTAL DESIGN

4.1 Methodology

Having introduced our main contributions, we now turn our attention to a discussion of implementation details and explore various hyperparameters. To quantitatively navigate this space, we use the training split of the Second Hand Song (SHS) dataset [7] for development and save the test split for our final evaluation. The SHS is a collection of 18,196 tracks from 5,854 "cliques", or distinct classes, with 12,960 from 4,128, respectively, set aside for training; the remainder constitutes the test set. Importantly, the SHS is also a subset of the MSD, which allows for large scale evaluation by using the entire MSD as background noise in a cover song retrieval task.

In line with previous work, the primary metrics of interest here are mean average precision (MAP) and average rank (AR). MAP is computed as the mean of the average precision over a set of queries, and reflects not only accuracy but also the order of correct documents in a ranked list. As an additional statistic, AR is computed as the average position of relevant documents, and measures where relevant documents fall in a ranked list. For evaluating performance in the training condition, each track in the training set is treated as a query and ranked relative to the remaining items in the training set, i.e. 1-vs-12,959; alternatively, in the test condition, each track in the test set is treated as a query and ranked relative to all other tracks in the MSD, i.e. 1-vs-999,999.

4.2 Impact of Sparse Projections

Here, we propose using the k-means algorithm to learn various dictionaries, inspired by recent work in [3]. While we acknowledge that there are alternative methods that could be applied to learn the bases of this transform, k-means is particularly attractive being unsupervised and relatively simple, having a single hyperparameter k. Noting that k-means is a batch, as opposed to on-line, learning algorithm, we first draw 50,000 2D-FMC vectors randomly from the SHS training set. This subset is used for both fitting PCA in the pre-processing set as well as learning dictionaries for various values of k; at this stage, we consider $k \in [128, 512, 1024, 2048]$. It is worth mentioning that due to the nuances of the algorithm —we use the Scipy implementation [8] — only 2045 elements were returned for $k = 2048$, as three of the centroids did not change. Additionally, after inspecting the data to determine a reasonable knee for the shrinkage function, we set $\theta = 0.2$ for our experiments.

Shown in Table 1, we find that applying learned k-means dictionaries as sparse projections, followed by median pooling and L_2-normalization, leads to slightly worse performance than the baseline system. This negative result illustrates that a sparse, higher dimensional feature space does not necessarily exhibit the organization necessary for distance to be meaningful. However, the goal of a sparse projection is only to make the information more separable, and this behavior must be explored further to determine its true impact on system performance.

[7] http://labrosa.ee.columbia.edu/millionsong/secondhand

[8] http://docs.scipy.org/doc/scipy/reference/cluster.vq.html

k	128	512	1024	2045	Baseline
MAP	3.44%	4.54%	4.92%	5.51%	**8.91%**
AR	3,248	3,154	3,112	**3,026**	3,097

Table 1. Exploring values of k on the Training set.

4.3 Semantically Organizing the Space

In light of this, we now seek to better encode semantic relationships with distance measures. Linear Discriminant Analysis (LDA) is a natural choice for learning a supervised embedding that jointly minimizes intra-class variance and inter-class discrimination. This approach also has a single hyperparameter N, the dimensionality of the projection, and we explore $N \in [50, 100, 200]$.

As shown in Table 2, the combination of sparse projections *and* supervised dimensionality reduction leads to considerably better performance on the training set. While this result says nothing about generalization, it more than demonstrates that the representation produced by projecting onto a learned dictionary is indeed significantly more separable. It is interesting to note how performance degrades sharply as a function of decreasing k, and less so with decreasing N. The interpretation of this is two-fold: one, because the dictionary learning is unsupervised, it requires an over-complete set of bases to adequately capture the "right" information for LDA to recover; and two, model complexity can be constrained by limiting N, and therefore serve as a type of regularization.

Before proceeding, it is necessary to ensure that this increase in performance is in fact due to the sparse projection and not just the supervised embedding. To test this hypothesis, we apply LDA to the baseline system, with the 2D-FMC pre-processing pipeline discussed in Section 3.1. Table 3 clearly shows that, though there is some improvement to be had via LDA alone, projecting into a higher dimensional space first is indeed significant, almost doubling MAP as a linear function of N.

Mean Average Precision			
$k \setminus N$	200	100	50
128	5.34%	4.82%	4.19%
512	9.30%	7.38%	4.95%
1024	13.99%	9.63%	5.63%
2045	**28.51%**	17.35%	9.05%
Average Rank			
$k \setminus N$	200	100	50
128	2,915	3,116	3,345
512	2,719	3,153	3,688
1024	2,420	2,980	3,665
2045	**1,844**	2,539	3,249

Table 2. Exploring impact of both k-means *and* LDA on the Training set.

Method	MAP	AR
Baseline + LDA(50)	5.35%	3,666
Baseline + LDA(100)	9.85%	3,034
Baseline + LDA(200)	14.31%	2,434
k-means(2045) + LDA(50)	9.05%	3,249
k-means(2045) + LDA(100)	17.35%	2,539
k-means(2045) + LDA(200)	**28.51%**	**1,844**

Table 3. Results for the SHS Training set applying LDA to the baseline, versus the best performing sparse projection.

Method	MAP	AR
Random	$\sim 0.001\%$	500,000
2DFTM + PCA(50) [1]	1.99%	**173,117**
2DFTM + PCA(200) [1]	2.95%	180,304
k-LDA(50)	**13.41%**	343,522
k-LDA(200)	0.83%	398,005
k-PCA(200) + LDA(200)	12.76%	338,882

Table 4. Results for the SHS Test set over the full MSD. Note that we contract k-means(2045) here simply as "k-".

5. LARGE-SCALE EVALUATION

So far, we have focused exclusively on the SHS training set, both as a computational simplification and an approach to system development. We now turn our evaluation to the test split of the SHS dataset to investigate how our approach generalizes to unseen data. Based on the results of the previous section, we reduce the parameter space by fixing $k = 2045$ but continue to observe performance as a function of N.

First, evidenced by the results given in Table 4, the combined k-means and LDA projections —which we contract here on as k-LDA for brevity— observe radically different behavior based on the dimensionality of the embedding. In fact, k-LDA(200), the best performing system on the training set, seemingly fails to generalize at all; MAP and AR are over two-times worse than the baseline system, and these results clearly indicate extreme over-fitting. Setting this observation aside for a moment though, something even more curious occurs with k-LDA(50). While the AR is also much worse than baseline, the MAP improves by a factor of 6. This behavior begs an obvious question: what is occurring under the surface such that these metrics move in drastically different directions?

On closer inspection, a rather surprising observation precipitates: despite a significantly worse AR, the k-LDA(50) projection actually produces a remarkable number of correct *nearest* neighbors, i.e. the top-ranked item in the list is an accurate match. This intuitively explains the discrepancy between these metrics, as MAP weights precision as a function of rank position, e.g. being correct at the top matters more than being correct lower in the list. Furthermore, despite pulling relevant tracks to the top of the list, the k-LDA(50) system also pushes some to the very bottom. As a result, the distribution of relevant items in the ranked list is bimodal, and AR is at a loss to characterize this behavior.

To get a better sense of this behavior, we investigate precision-@-k, defined simply as the precision over the top-k items in a ranked list. Figure 2 clearly illustrates how our proposed method not only yields better performance overall, but offers improved usability as well. For this particular test set, the system gets the top result correct nearly 25% of the time, out of a space of one million possible items. Considering the top 10 results, or approximately the first page of a web search, about 5% of the documents are correct; in other words, there is a 50% chance that a true cover will appear on the first page of a search.

Turning back to the k-LDA(200) projection, the question now becomes how to reduce such substantial over-fitting. Fortunately, projecting into a PCA subspace before fitting LDA has been shown to reduce over-fitting in the image processing and pattern recognition communities, notably for face recognition [6]. This is because PCA dimensionality reduction avoids singularities or near singularities in any of the scatter matrices used in LDA; this problem is exacerbated for small datasets or high dimensional feature spaces, of which this application is both. Furthermore, the cascade of PCA and LDA has been shown to be a general case of other LDA variations like uncorrelated LDA (ULDA), which are also used to avoid the singularity issue. Most importantly, how much PCA alleviates LDA over-fitting depends on the dimensionality of the intermediate PCA subspace. Therefore, selecting the right number of principal components is both crucial for good results, and non-trivial.

In lieu of a more extensive exploration, we perform an initial inquiry into the potential of PCA to address this particular problem. Here, we fit a 200 dimensional PCA subspace by transforming the SHS training set into its mid-level, 2045-dimensional representation, just before the application of LDA. In an effort to help minimize potential singularities and other such problems, we take two additional steps when fitting LDA to encourage better generalization; however, the true impact of such decisions are admittedly uncertain. First, we subsample the training set, only using cliques with 9 or more tracks each. Then, we include an arbitrarily large number of tracks that do not belong to any clique. This resulting embedding, dubbed k-PCA(200)+LDA(200), is then evaluated on the MSD, and we again recover performance roughly on par with k-LDA(50), e.g. relatively low AR, but a significantly higher MAP than baseline.

Finally, in terms of computation time, our method takes three more times to compute than baseline. In a machine with plenty of RAM, 16 cores, and splitting the process into 10 different threads, the baseline takes 8.7 hours to compute the features of 50, 100 and 200 PCA components. However, as our method produces features with the same output dimensionality as the baseline, our distance calculations —the prohibitive computation— requires the same amount of time. More specifically, it takes 0.4, 0.9, and 1.5 hours using 50, 100, and 200 components respectively.

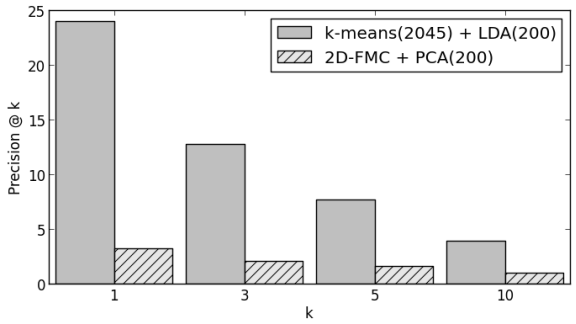


Figure 2. Comparison of Precision@k on the Test set for k-LDA(50), versus the best baseline result.

6. CONCLUSIONS

In this work we have presented an improved system for large-scale cover song retrieval, demonstrating how sparse, high-dimensional projections can be combined with low-dimensional embeddings to achieve greater performance than either piece alone. This semantically organized space is recovered by efficiently capturing shift-invariant features by effectively performing convolutional sparse coding in the Fourier magnitude domain, and learning a supervised cover-similarity space where distance is meaningful. Our system not only achieves state-of-the-art performance with respect to previously used evaluation metrics (MAP), but greatly improves precision-at-k for k less than 10, indicative of a more useful system. This encourages the additional observation that top-k, as opposed to full-list, metrics may be more informative for characterizing the usability of large scale information retrieval systems.

Looking toward future work, we identify several areas with the potential for improvement. As mentioned, there are a variety of ways the sparse dictionary could be learned; and, depending on the temporal pooling strategy defined, it would be possible to fine-tune the overall architecture like a deep network via backpropagation. Additionally, there are other pooling strategies that could be employed, leveraging structural knowledge to summarize the information over a full track in more musically meaningful ways. Lastly, the challenge of realizing a semantically organized space for computing distances between tracks is hardly a solved problem. Over-fitting seems to be a problem in higher dimensions, but the PCA-subspace trick discussed offers encouraging results, complementing those obtained directly from low-dimensional LDA.

Finally, to facilitate reproduction of results and encourage future work, we provide an open source implementation of our method in a public repository [9] .

7. REFERENCES

[1] Thierry Bertin-Mahieux and Daniel P. W. Ellis. Large-Scale Cover Song Recognition Using The 2D Fourier Transform Magnitude. In *Proc. of the 13th International Society for Music Information Retrieval Conference*, pages 241–246, 2012.

[2] 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*, Miami, FL, USA, 2011.

[3] Adam Coates and Andrew Y Ng. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*, pages 561–580. Springer, 2012.

[4] Daniel PW Ellis and Graham E Poliner. Identifying cover songs' with chroma features and dynamic programming beat tracking. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–1429. IEEE, 2007.

[5] M. Henaff, K. Jarrett, K. Kavukcuoglu, and Y. LeCun. Unsupervised learning of sparse features for scalable audio classification. In *Proc. 12th Int. Conf. on Music Information Retrieval (ISMIR)*, 2011.

[6] Shuiwang Ji and Jieping Ye. Generalized linear discriminant analysis: a unified framework and efficient model selection. *Neural Networks, IEEE Transactions on*, 19(10):1768–1782, 2008.

[7] Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu, and Yann LeCun. Learning convolutional feature hierachies for visual recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[8] Geoffroy Peeters. Spectral and temporal periodicity representations of rhythm for the automatic classification of music audio signal. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(5):1242–1252, 2011.

[9] Joan Serrà, 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.

[10] Joan Serrà, Emilia Gómez, Perfecto Herrera, and Xavier Serra. Chroma binary similarity and local alignment applied to cover song identification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(6):1138–1151, 2008.

[11] Joan Serrà, Xavier Serra, and Ralph G Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11(9):093017, 2009.

[9] https://github.com/urinieto/LargeScaleCoverSongId

SIMULTANEOUS UNSUPERVISED LEARNING OF FLAMENCO METRICAL STRUCTURE, HYPERMETRICAL STRUCTURE, AND MULTIPART STRUCTURAL RELATIONS

Dekai Wu
HKUST, Human Language Technology Center, Department of CSE, Hong Kong
dekai@cs.ust.hk

ABSTRACT

We show how a new unsupervised approach to learning musical relationships can exploit Bayesian MAP induction of stochastic transduction grammars to overcome the challenges of learning complex relationships between multiple rhythmic parts that previously lay outside the scope of general computational approaches to music structure learning. A good illustrative genre is flamenco, which employs not only regular but also irregular hypermetrical structures that rapidly switch between 3/4 and 6/8 *mediocompas* blocks. Moreover, typical flamenco idioms employ heavy syncopation and sudden, misleading off-beat accents and patterns, while often elliding the downbeat accents that humans as well as existing meter-finding algorithms rely on, thus creating a high degree of listener "surprise" that makes not only the structural relations, but even the metrical structure itself, ellusive to learn. Flamenco musicians rely on both complex regular hypermetrical knowledge as well as irregular real-time clues to recognize when to switch meters and patterns. Our new approach envisions this as an integrated problem of learning a bilingual transduction, i.e., a structural relation between two languages—where there are different musical languages of, say, flamenco percussion versus *zapateado* footwork or *palmas* hand clapping. We apply minimum description length criteria to induce transduction grammars that simultaneously learn (1) the multiple metrical structures, (2) the hypermetrical structure that stochastically governs meter switching, and (3) the probabilistic transduction relationship between patterns of different rhythmic languages that enables musicians to predict when to switch meters and how to select patterns depending on what fellow musicians are generating.

1. INTRODUCTION

Little work has been done on automatic algorithms for learning across multiple parts in rhythmically complex music genres such as flamenco, despite a respectable history of work on automatic meter finding utilizing a wide range of underlying modeling paradigms. Repetition of patterns is a central feature in many automatic meter finding algorithms such as that of Steedman [23]. Early approaches were rule-based (e.g., Longuet-Higgins and Steedman [16]), while others employed neural nets (e.g., Desain and Honing [6]) or preference rules (e.g., Povel and Essens [19], Temperley and Sleator [28]). More recent approaches are based on probabilistic modeling, such as the generative models of Cemgil *et al.* [4]), Raphael [20], and Temperley [27].

Generally, however, these approaches are based on relatively simplistic assumptions about straightforward duple, triple, or 4/4 meters accompanied by the regular occurrence of strong accents on or near the downbeat. These assumptions are widely understood to apply primarily to Western music conventions rather than worldwide music genres that can be rhythmically much more complex. Flamenco conventions, for example, often defy for example what Lerdahl and Jackendoff [14] termed the "Strong Beat Early" rule—omitting the downbeat accent is a typical idiom, and in fact the strong beat is often understood in flamenco to be late. Moreover, flamenco rhythms employ continual meter switching in both regular and irregular ways, with a complex hypermetrical language governing the switching, and make frequent use of polyrhythm in addition.

Even less work has been done to date on computational approaches to analysis of flamenco. Models such as those of Diaz-Banez *et al.* [7], Gomez and Bonada [12], Guastavino *et al.* [11], Mora *et al.* [17], or Thul and Toussaint [29] represent various intriguing attacks on specific aspects of flamenco, but do not attempt to actually induce the musical structures.

To attack more complex rhythmic forms such as these, we propose an approach based on unsupervised induction of stochastic transduction grammars. On one hand, this follows the generative modeling paradigm of GTTM [14] and Steedman [24] or [25] in which various aspects of music can each be modeled as languages that can be generated by formal grammars. On the other hand, to facilitate automatic learning and scaling up of the models, we formulate the task in terms of stochastic grammars that describe probabilistic models of musical structure.

The majority of previous work on stochastic grammatical models for music employs flat Markov models and/or hidden Markov models (HMMs). For example, both the Continuator model of Pachet [18] and the Factor Oracle models of Assayag *et al.* [2] use Markov models to learn

music improvisation conventions-an approach further explored by François *et al.* [8] and [9]. A grammar induction approach for learning jazz grammars under Markovian assumptions is proposed by Gillick *et al.* [10]. Relatively little has been done on musical structure modeling using stochastic context-free grammars [13]. Related work on unsupervised learning of CCMs (a variant of SCFGs) for musical grammars includes that of Swanson *et al.* [26], or the Data Oriented Parsing approach of Bod [3].

Stochastic grammars are excellent for describing individual aspects of music. Much of music, however, is about the loosely coupled relationships between *multiple* strands of different kinds of sequences taking place in parallel.

Our new approach differs from previous stochastic grammatical models of music in that we (1) shift to *bilingual* stochastic transduction grammars instead of conventional monolingual stochastic grammars, allowing us to model learning structural *relations* between different musical languages of separate percussive flamenco parts, and (2) apply a new grammar induction strategy that searches for the Bayesian MAP (maximum *a posteriori*) model encompassing metrical relations, hypermetrical relations, and probabilistic transduction relations in a single integrated process.

2. STOCHASTIC TRANSDUCTION GRAMMARS

In classic formal language theory, a **transduction** is a relation between two languages, which is exactly what we wish to induce. A **transduction grammar** or **translation grammar** (TG) is a bilingual grammar of transductions, and describes structured relations between two languages [1], [15]. (An equivalent term "synchronous grammar" used only in computational linguistics is not as long established or widely understood throughout computer science.)

Thus **stochastic transduction grammars** are probabilistic bilingual grammars of transductions, and describe structured relations between two languages *probabilistically*—which means that stochastic TGs do not suffer from the overly rigid constraints of non-stochastic transduction models, and can be automatically learned [31]. In a stochastic transduction grammar, a probability distribution is imposed over the space of possible derivations. This is typically done by associating a conditional probability with each rule, representing the probability that any nonterminal symbol matching the left-hand-side of the rule generates children matching the right-hand-side of the rule. Techniques have been developed for numerous tasks utilizing stochastic transduction grammars including aligning bilingual corpora, unsupervised segmentation and annotation of bilingual corpora, automatic induction of bilingual correspondences, grammar induction for stochastic TGs, and so on [32].

In this paper we will make use of stochastic **monotonic transduction grammars**, which in terms of generative capacity sit in the hierarchy of transduction grammars between stochastic finite-state transducers and linear inversion transduction grammars, as detailed in [32].

A monotonic transduction grammar or MTG (equivalent to the "simple syntax-directed transduction grammar" or "simple SDTG" of Aho and Ullman) in normal form is a tuple $\langle N, \Sigma, \Delta, S, R \rangle$ where N is a finite nonempty set of nonterminal symbols, Σ is a finite nonempty set of input languge symbols, Δ is a finite nonempty set of output language symbols, $S \in N$ is the designated start symbol, and R is a finite nonempty set of syntax-directed transduction rules on the forms:

$$S \to A, \quad A \to \varphi, \quad A \to e/f$$

where $A \in N$, $\varphi \in NNN^*$, and e and f are **terminal** symbols representing musical event segments as follows.

Strings in both the languages represent sequences of symbolic musical event tokens. A "sentence" is a full musical passage for a single instrumental part, whereas a "bisentence" is a matched musical passage with both instrumental parts. For convenience of musical interpretation, instead of writing musical sequences using linguistic string notation, we shall use conventional music staff notation for musical event segments e and f, as in Figure 1. Just as we consider the monolingual terminal symbols e and f to represent musical event segments, we consider the bilingual e/f notation to denote a **biterminal** symbol representing a *parallel pair* of musical event segments from different musical instruments. Technically, $e/f \in (\Sigma^* \times \Delta^*) - (\epsilon/\epsilon)$, in which we exclude the degenerate case of pairing a zero-length empty segment ϵ with another zero-length empty string ϵ to avoid unnecessary complications arising from infinite recursion.

3. TRANSDUCTION GRAMMAR INDUCTION

In this section we describe our new model for unsupervised induction of stochastic transduction grammars. For concrete examples of the abstract model, see Section 4.

Minimizing description length We begin with the overall Bayesian model whose posterior we wish to maximize. We seek the maximum *a posteriori* (MAP) model given the data; that is, we attempt to optimize the posterior probability following Bayes' rule

$$P(\Phi \mid D) = \frac{P(\Phi)\, P(D \mid \Phi)}{P(D)}$$

where Φ is the model and D is the training data. The prior probability of the data is constant during search, which gives us the following search problem:

$$\underset{\Phi}{\operatorname{argmax}} P(\Phi)\, P(D \mid \Phi)$$

In our case, the probability of the data given the model can be determined through parsing since it is a grammar. The prior of the model is, however, somewhat more complicated because it must incorporate the effect of both the structure and the parameters of the model.

$$P(\Phi) = P(\Phi_G)\, P(\Phi_S \mid \Phi_G) P(\theta_\Phi \mid \Phi_S, \Phi_G)$$

Φ_G is a global prior over possible model formalisms, which we set to be the space of possible monotonic transduction grammars, $P(\Phi_S \mid \Phi_G)$ is a prior on the model

structure given the model formalism, and $P(\theta_\Phi \mid \Phi_S, \Phi_G)$ is a prior over the model parameters given the formalism and the structure. We approximate the prior over the model structure using the description length of the model:

$$-\log_2\left(P(\Phi_S \mid \Phi_G)\right) \propto \mathrm{DL}\left(\Phi_S\right)$$

The description length of a model is calculated by summing the length of all the rules, where each unique (monolingual) terminal segment is efficiently given a unique Huffman encoding. This avoids redundant double-counting of terminal segments that appear in more than one rule. The length of a symbol is proportional to $-\log_2\left(\frac{1}{M}\right)$ where $M = 2 + N + \Sigma + \Delta$ is the total number of symbols (N is the number of nonterminals, Σ is the size of the L_0 vocabulary and Δ is the size of the L_1 vocabulary). Thus, for example, reducing the number of distinct nonterminals in a grammar reduces its description length.

We set the prior over the model parameters to be a uniform Dirichlet distribution over right-hand sides given left-hand sides:

$$P(\theta_\Phi \mid \Phi_S, \Phi_G) = \prod_{i=0}^{N-1} \frac{1}{B(\alpha_0, \alpha_1, ..., \alpha_{R_{n_i}-1})} \prod_{j=0}^{R_{n_i}-1} \theta_\Phi^{n_i}(j)$$

where N is the number of nonterminals, R_{n_i} is the set of rules where n_i is the left-hand side, and $\theta_\Phi^{n_i}$ is a function that gives the rule probabilities for rule where the left-hand side is n_i. Fleshing out the search problem, we have:

$$\underset{\Phi_G, \Phi_S, \theta_\Phi}{\operatorname{argmax}} \; P\left(\Phi_G\right) P(\Phi_S \mid \Phi_G) P(\theta_\Phi \mid \Phi_S, \Phi_G) P(D \mid \Phi_G, \Phi_S, \theta_\Phi)$$

Recall that are restricting Φ_G to monotonic transduction grammars. We further divide the search into two phases: a top-down rule segmentation phase, which focuses on the structural induction to optimize $P(\Phi_S \mid \Phi_G)$ and $P(D \mid \Phi_G, \Phi_S, \theta_\Phi)$, and a parameter tuning phase, which focuses on $P(\theta_\Phi \mid \Phi_S, \Phi_G)$ and $P(D \mid \Phi_G, \Phi_S, \theta_\Phi)$.

Initializing model structure The induction procedure starts with a transduction grammar that memorizes the training data as well as possible, and generalizes from there. The transduction grammar that best fits the training data is the one where the start symbol rewrites to the full sentence pairs that it has to generate. It is also possible to add any number of nonterminal symbols in the layer between the start symbol and the bisentences without altering the probability of the training data. We take initial advantage of this by allowing for one intermediate symbol so that the start symbol conforms to the normal form and always rewrites to precisely one nonterminal symbol.

Our initial model thus consists of the rule $S \rightarrow A$ plus numerous rules of the form $A \rightarrow e_{0..T}/f_{0..V}$ where S is the start symbol, A is the nonterminal, T is the length of the output sentence, and V is the length of the input sentence.

Generalizing model structure In order to generalize the initial monotonic transduction grammar we need to identify parts of the existing biterminals that could be validly used in isolation, and allow them to combine with other segments. This is the very feature that allows a finite transduction grammar to generate an infinite set of sentence pairs; when we do this, we move some of the probability mass which was concentrated in the training data out to other data that are still unseen—i.e. we generalize from the training data. The general strategy is to propose a number of sets of biterminal rules and a place to segment them, estimate the posterior given the sets and commit to the best set. That is: we do a greedy search over the power set of possible segmentations of the rule set. This intractable problem can be reasonably efficiently approximated.

The key component in the approach is the ability to evaluate the change in *a posteriori* probability if a specific segmentation was made in the grammar. This can then be extended to a set of segmentations, which only leaves the problem of generating suitable sets of segmentations. Any segment that can be reused maximizes the model prior. The more rules we can find with shared biaffixes, the more likely we are to find a good set of segmentations.

Our algorithm takes advantage of the above observation by focusing on both the monolingual and bilingual affixes (i.e., prefixes or suffixes) found in the training data. Each **affix** or **biaffix** defines a set of lexical rules paired up with a possible segmentation. We evaluate the (bi)affixes by estimating the change in posterior probability associated with committing to all the segmentations defined by a (bi)affix. This allows us to find the best set of segmentations, and commit to as many of them as possible. Moreover, as we generate new nonterminal categories during this process, we also use affixes and biaffixes to suggest possible merges of the nonterminal categories. This minimizes the parsing efforts, which are more expensive. A priority queue based agenda keeps track of possible candidates for rule segmentation and nonterminal category actions, and always greedily commits at each step to the action that best improves overall posterior probability:

```
G = the transduction grammar
biaffixes_to_rules = index of G's transduction rules by their (bi)affixes
lhs_to_rules = index of G's transduction rules by their LHS nonterms
agenda = []        // Priority queue of actions by their DL impact on G
for each affix or biaffix x in G :
   delta = eval_seg_post(x, biaffixes_to_rules[x], G)
   if (delta < 0)
      agenda.add(SEGMENT, x, delta)
while agenda.pop(act, x) < 0 :
   if (act == SEGMENT)
      real_delta = eval_seg_post(x, biaffixes_to_rules[x], G)
      if (real_delta < 0)
        G, modified_rules = segment_rules(x, biaffixes_to_rules[x], G)
         for each pair y of nonterms serving as LHS of modified_rules
               that share a common (monolingual or bilingual) RHS :
            delta = eval_merge_post(x, biaffixes_to_rules[x], G)
            agenda.add(MERGE, y, delta)
   else if (act == MERGE)
      real_delta = eval_merge_post(y, lhs_to_rules[y], G)
      if (real_delta < 0)
         G = merge_nonterms(y, lhs_to_rules[y], G)
```

Note that *both* affixes and biaffixes are handled in `biaffixes_to_rules` (since an affix can be regarded as a special case of a biaffix where one of the two affixes is the empty string ϵ). We have written `eval_seg_post` and `segment_rules` as shorthand for the above-discussed evaluation of the impact of a rule segmentation action upon the

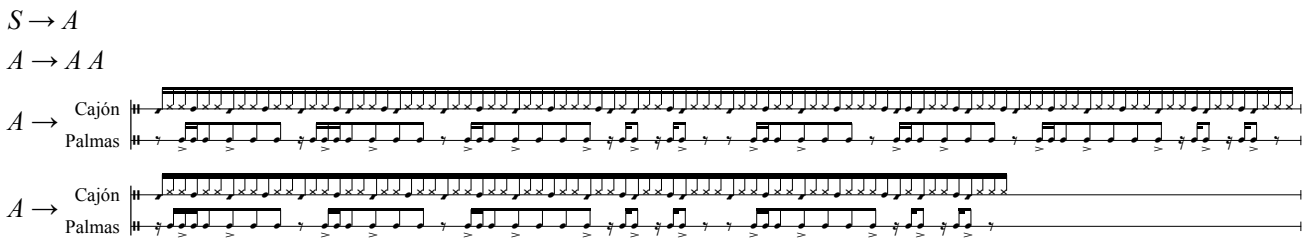


Figure 1. Initial transduction grammar with two training examples (see text). Lexical transduction rules are shown with their biterminals in *cajón* and *palmas* staves using standard music notation for sequences instead of character strings.

posterior. This consists of removing the rule being segmented, inserting the three new rules (one structural and two terminal) along with two new nonterminals, and uniformly distributing the segmented rule's probability mass over the three new rules; exact mathematical details are in [21]. Similarly, the shorthand `eval_merge_post` and `merge_nonterms` denote consolidating the probability mass of two nonterminal categories.

The change in the model prior is easy to estimate, as it is proportional to the change in grammar length when the old rule is removed and the new rules are inserted (keeping in mind that a rule can only be added once, so if it already exists inserting it will not change the description length).

The change in the probability of the data given the model is expensive to get through biparsing the data, so instead we accumulate enough statistics during biparsing to be able to make an educated guess. We follow the analogous procedure after merging two nonterminals.

Optimizing model parameters Although the iterative segmentation of the rules result in reasonable parameters, there is still room for improvement. In this phase we consider the model structure to be fixed, and optimize the model parameters to give the highest possible posterior probability, i.e., we fix Φ_S (to be what we arrived at using the algorithm described in the previous section), as well as Φ_G (which remains fixed as MTGs). The two remaining free factors in the MAP are thus: $P(\theta_\Phi \mid \Phi_S, \Phi_G)$ and $P(D \mid \Phi_G, \Phi_S, \theta_\Phi)$—the prior over the parameters and the conditional probability of the data given the complete model.

The prior is, as described earlier, a uniform Dirichlet distribution over all the rules, which can be described using a concentration parameter. To get the conditional, we have to biparse the training data, and to maximize it, we perform expectation maximization [5], as a special case of EM as specified for inversion transduction grammars by [30]. This requires biparsing, which we do with the cubic time biparsing algorithm described in [22].

4. RESULTS

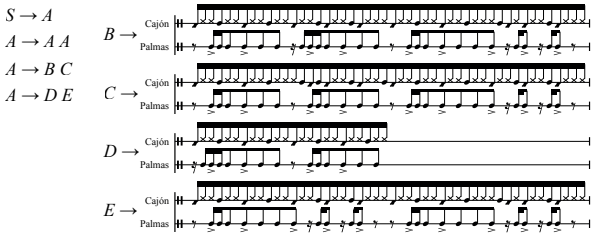
For our experiments we chose the *buleriás* form of flamenco because of its metrical and hypermetrical complexity. Various passages from multi-track recordings were collected, from which were taken approximately 5 minutes of aligned *cajón* (box drum) and *palmas* (clapping percussion) tracks. Symbolic "tick" notes were extracted from the *cajón* and *palmas* tracks via heuristics primarily relying on a combination of volume and frequency range filters. For the *cajón* the notes were separated into "bass", "tone", and "tip" categories. For the *palmas* the notes were separated into ordinary or accented notes. All notes were quantized into 1/16th note intervals.

As no meaningful gold standard for mechanically evaluating the quality of the learned model exists, only subjective evaluations are possible. Moreover, variance in flamenco expectations is extremely large, and our subjective evaluations were so close to 100% accuracy so as to be swamped by statistical variance in human judgments. Reasons for the high accuracy of the model can best be seen by tracing specifically how it learns, looking at a small concrete subset of the training set.

Two short training passages are shown in Figure 1. Note that in our induction method's rule segmenting strategy, the initial transduction grammar starts out containing one rule for each training example, each belonging to the same generic category A as shown by the left-hand-side nonterminal. In addition, the grammar contains the start rule and a low-probability "glue rule" that allows arbitrary concatenation of any valid sequences when all else fails.

Learning metrical structure In the early iterations, the MDL-driven induction algorithm primarily works toward learning transduction patterns that are a single full *compas* in length, i.e., a twelve beat cycle. This accurately mirrors the primary (mixed meter) structure that is most common and most fundamental in flamenco.

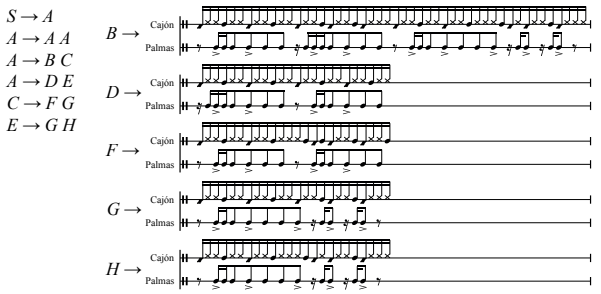
The two lexical transduction rules in the initial transduction grammar of Figure 1 share no biaffixes, but the first half of the first lexical rule's *cajón* part is repeated at the end of the second lexical rule's *cajón* part. Thus, the first induction decision is to segment both lexical transduction rules so as to gain the bits from efficiently encoding their common *cajón* sequence, instead of redundantly enumerating the same string twice. This yields a revised grammar



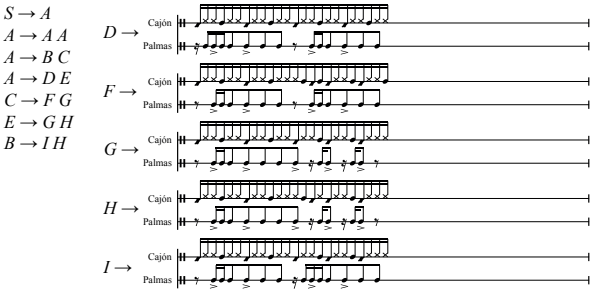
with shorter description length, that introduces new nonter-

minals so as to generate the same transduction as before.

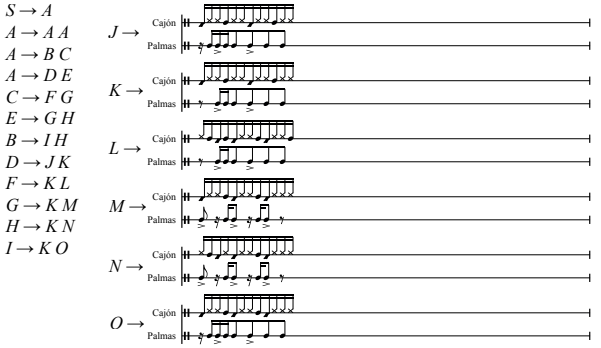
In the new grammar, a shared biaffix is found, between the second half of the C rule and the first half of the E rule. Segmenting both rules, again introducing new nonterminals as needed, yields:



This has created another shared biaffix—the new H rule is a bisuffix of the B rule. Segmenting the B rule yields:



Note that the algorithm has learned the full *compas* length patterns. Typically, in subsequent iterations, the induction process moves on to gradually learn *mediocompas* patterns that are only six beats in length. Here, the next segmentation, based on the shared bisuffix of the D and F rules and biprefix of the G, H, and I rules, has this effect:

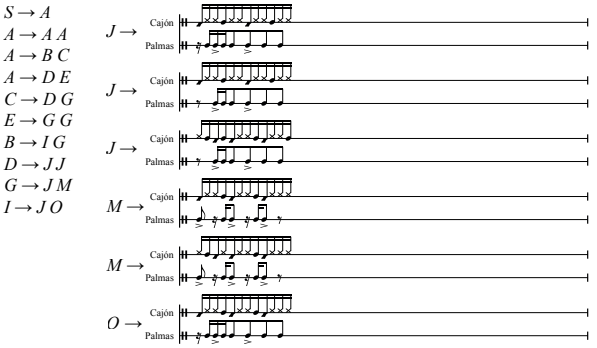


At this point, there are no more rules that can be segmented to lower the description length. However, we notice that it is still possible to improve the posterior probability of the model, because some of the newly created nonterminal categories might be merged in such a way as to improve the model structure prior $P(\Phi_S \mid \Phi_G)$by reducing the model's description length in terms of the number of nonterminal categories, without introducing so much error that it excessively decreases the likelihood of the data $P(D \mid \Phi_G, \Phi_S, \theta_\Phi)$.

The nonterminals that have just been created generate four candidates for merging, since the algorithm considers each pair of nonterms serving as the LHS of modified rules that share a common RHS: $J \approx K$, $J \approx O$, $K \approx L$, and $M \approx N$. Of these, $K \approx L$ gives by far the best improvement in posterior probability, so we replace all instances of L in the grammar with K instead.

The next best merge is $J \approx K$, so we replace in turn all instances of K (including those formerly L) with J. In so doing, the updated rules for D and F become identical, both now with $J\ J$ on the right-hand-side. Thus we propagate the merging to replace all instances of F with D, at no additional cost. Similarly, merging $M \approx N$ slightly improves the posterior, and in so doing, the updated rules for G and H become identical so we also merge H into G.

Merging $J \approx K$, $K \approx L$, and $M \approx N$ inherently introduces generalizations that are able to (bi)parse new examples outside the training set. The iteration terminates without merging $J \approx O$ which would introduce too much error to improve the posterior. The final transduction grammar induced is thus:



Accurately reflecting flamenco norms, induction has categorized the mediocompas patterns into three distinct nonterminal types: J comprises patterns in 6/8 meter, while M comprises patterns in 3/4 meter, and O is a polyrhythmic pattern that crosses both. The fact that a distinction between the two meters can be learned—even though our current approach does not incorporate any explicit a priori model of accented pulses at constant repeated intervals—arises from the transduction grammar induction's natural integration of metrical structure learning together with hypermetrical structure learning, as discussed below.

Between the 6/8 and 3/4 transduction patterns, certain common *palmas* sequences appear in both. This correctly reflects conventional flamenco usage of palmas (playing a similar function to clave patterns in Afro-Latin genres). However, the 3/4 transduction patterns also tend more frequently to relate certain *palmas* sequences that do not generally appear with 6/8 patterns. Such patterns tend to have notes that align more naturally with 6/8 accents. *Palmas* sequences are useful to learning because of their clave-like function, even though they are not as consistent as Afro-Latin clave, and are usually silent on what would be the strong downbeat in most mainstream dance music forms.

Learning hypermetrical structure Among the many flamenco forms, the *buleriás* style is particularly aggressive about using *mediocompas* patterns in irregular ways. The learned rule $G \rightarrow J\ M$ models a regular full alternating meter *compas*, while $I \rightarrow J\ O$ models the same polyrhythmically against a 6/8 *mediocompas* feel. The rule $D \rightarrow J\ J$ models a full compas in 6/8 *mediocompas* feel. The

rules for B, C, and E model longer *compas* pairs, with typical idioms such as staying in 6/8 meter until the final fourth *mediocompas*. This behavior is naturally emergent from the MDL-driven induction (even more so on larger training sets).

Learning probabilistic transduction relationships The induced transduction grammar is potentially useful in a wide variety of applications, which we plan to investigate in detail in our next steps. Currently, the learned model can already be used to predict suitable accompaniment for either instrumental part. Given a previously unseen *cajón* sequence, a Viterbi parse translates the sequence into the most probable *palmas* sequence with nearly perfect accuracy. Similarly, given a new *palmas* sequence, the model can translate it into the most probable *cajón* sequence, which is currently generally acceptable though not necessarily musically optimal. Our future work will focus on further refining the predictive accuracy for accompaniment from a stylistic standpoint.

The new MDL-driven transduction grammar induction method we have introduced is the first to (1) exploit opportunities to compress both monolingual affixes and bilingual affixes, (2) exploit regularities in either single language to help segment rules describing both languages, and (3) exploit both monolingual and bilingual regularities to induce categories for longer hypermetrical patterns. We anticipate numerous further applications beyond the flamenco genre.

5. ACKNOWLEDGMENTS

This material is based upon work supported in part by the Hong Kong Research Grants Council (RGC) research grants GRF620811, FSGRF13EG28, GRF621008, and GRF612806; the Defense Advanced Research Projects Agency (DARPA) under BOLT contract no. HR0011-12-C-0016, and GALE contract nos. HR0011-06-C-0022 and HR0011-06-C-0023; and by the European Union under the FP7 grant agreement no. 287658. Thanks to Markus Saers, Karteek Addanki, Chi-kiu Lo, and Meriem Beloucif for assistance with implementation. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the RGC, EU, or DARPA.

6. REFERENCES

[1] Alfred V. Aho and Jeffrey D. Ullman. *The Theory of Parsing, Translation, and Compiling (Volumes 1 and 2)*. Prentice-Halll, Englewood Cliffs, NJ, 1972.

[2] Gérard Assayag, Georges Bloch, Marc Chemillier, Arshia Cont, and Shlomo Dubnov. OMax Brothers: A dynamic topology of agents for improvization learning. In *First ACM Workshop on Audio and Music Computing Multimedia*, pages 125–132, 2006.

[3] Rens Bod. Stochastic models of melodic analysis: Challenging the gestalt principles. *Journal of New Music Research*, 30(3), 2001.

[4] Ali Taylan Cemgil, Bert Kappen, Peter Desain, and Henkjan Honing. On tempo tracking: Tempogram representation and Kalman filtering. *Journal of New Music Research*, 29(4):259–273, 2000.

[5] Arthur Pentland Dempster, Nan M. Laird, and Donald Bruce Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[6] Peter Desain and Henkjan Honing. *Music, Mind, and Machines: Studies in Computer Music, Music Cognition, and Artificial Intelligence*. Thesis Publications, Amsterdam, 1992.

[7] J. Miguel Díaz-Báñez, Giovanna Farigu, Francisco Gómez, David Rappaport, and Godfried T. Toussaint. El compás flamenco: A phylogenic analysis. In *BRIDGES: Mathematical Connections in Art, Music and Science*, pages 61–70, Southwestern College, Winfield, Kansas, Jul 2004.

[8] Alexandre R.J. François, Elaine Chew, and Dennis Thurmond. Mimi - a musical improvisation system that provides visual feedback to the performer. Technical Report 07-889, USC Computer Science Department, Apr 2007.

[9] Alexandre R.J. François, Isaac Schankler, and Elaine Chew. Mimi4x: An interactive audio-visual installation for high-level structural improvisation. In *IEEE International Conference on Multimedia and Expo (ICME 2010)*, pages 1618–1623, 2010.

[10] Jon Gillick, Kevin Tang, and Robert M. Keller. Machine learning of jazz grammars. *Computer Music Journal*, 34(3):56–66, Fall 2010.

[11] Catherine Guastavino, Francisco Gómez, Godfried Toussaint, Fabrice Marandola, and Emilia Gómez. Measuring similarity between flamenco rhythmic patterns. *Journal of New Music Research*, 38(2):129–138, 2009.

[12] Emilia Gómez and Jordi Bonada. Automatic melodic transcription of flamenco singing. In *Fourth Conference on Interdisciplinary Musicology (CIM08)*, Thessaloniki, Greece, Jul 2008.

[13] Karim Lari and Steve J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

[14] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, 1983.

[15] Philip M. Lewis and Richard E. Stearns. Syntax-directed transduction. *Journal of the Association for Computing Machinery*, 15(3):465–488, 1968.

[16] Hugh Christopher Longuet-Higgins and Mark J. Steedman. On interpreting Bach. *Machine Intelligence*, 6:221–241, 1971.

[17] Joaquín Mora, Francisco Gómez, Emilia Gómez, Francisco Escobar-Borrego, and José Miguel Díaz-Báñez. Characterization and melodic similarity of a cappella flamenco cantes. In *11th International Society for Music Information Retrieval Conference (ISMIR)*, pages 351–356, 2010.

[18] François Pachet. The continuator: Musical interaction with style. *Journal of New Music Research*, 32(3):33–341, 2003.

[19] Dirk-Jan Povel and Peter Essens. Perception of temporal patterns. *Music*, 2(4):411–440, Summer 1985.

[20] Christopher Raphael. A hybrid graphical model for rhythmic parsing. *Artificial Intelligence*, 137(1-2):217–238, May 2002.

[21] Markus Saers, Karteek Addanki, and Dekai Wu. Iterative rule segmentation under minimum description length for unsupervised transduction grammar induction. In Adrian-Horia Dediu, Carlos Martín-Vide, Ruslan Mitkov, and Bianca Truthe, editors, *First International Conference on Statistical Language and Speech Processing (SLSP 2013)*, volume 7978 of *LNAI*, pages 224–235, Tarragona, Spain, Jul 2013. Springer.

[22] Markus Saers, Joakim Nivre, and Dekai Wu. Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithms. In *11th International Conference on Parsing Technologies (IWPT'09)*, pages 29–32, Paris, Oct 2009.

[23] Mark J. Steedman. The perception of musical rhythm and metre. *Perception*, 6(5):555–569, 1977.

[24] Mark J. Steedman. The formal description of musical perception. *Music Perception*, 2:52–77, 1984.

[25] Mark J. Steedman. The blues and the abstract truth: Music and mental models. In A. Garnham and J. Oakhill, editors, *Mental Models in Cognitive Science*, pages 305–318. Erlbaum, 1996.

[26] Reid Swanson, Elaine Chew, and Andrew S. Gordon. Supporting musical creativity with unsupervised syntactic parsing. In *AAAI Spring Symposium on Creative Intelligent Systems*, 2007.

[27] David Temperley. *Music and Probability*. MIT Press, 2007.

[28] David Temperley and Daniel Sleator. Modeling meter and harmony: A preference-rule approach. *Computer Music Journal*, 23(1):10–27, 1999.

[29] Eric Thul and Godfried T. Toussaint. On the relation between rhythm complexity measures and human rhythmic performance. In *Conference on Computer Science & Software Engineering (C3S2E '08)*, 2008.

[30] Dekai Wu. Trainable coarse bilingual grammars for parallel text bracketing. In *Third Annual Workshop on Very Large Corpora (WVLC-3)*, pages 69–81, Cambridge, MA, Jun 1995.

[31] Dekai Wu. Stochastic Inversion Transduction Grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404, Sep 1997.

[32] Dekai Wu. Alignment. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing*, pages 367–408. Chapman and Hall / CRC, second edition, 2010.

Oral Session 2:

Musical Cultures

A CORPUS-BASED STUDY ON RAGTIME SYNCOPATION

Anja Volk
Utrecht University
A.Volk@uu.nl

W. Bas de Haas
Utrecht University
W.B.deHaas@uu.nl

ABSTRACT

This paper presents a corpus-based study on syncopation patterns in ragtime. We discuss open questions on the ragtime genre and the potential of computational tools in addressing these questions, contributing to the fields of Musicology and Music Information Retrieval (MIR), and giving back to the ragtime enthusiasts community. We introduce the RAG-collection of around 11000 ragtime MIDI files collected, organised, and distributed by many ragtime lovers around the world. The collection is accompanied by a compendium, providing useful metadata on ragtime compositions. Using this collection and the compendium, we investigate syncopation patterns in ragtime melodies, for which we tailored a melody extraction algorithm. We test and confirm musicological hypotheses about the occurrence of syncopation patterns that are considered typical for ragtime on the extracted melodies. Thus, the paper presents a first step towards modelling typical characteristics of the ragtime genre, which is an important means for enabling automatic genre classification.

1. INTRODUCTION

This paper presents a corpus-based study on ragtime syncopation using a collection of around 11000 ragtime MIDI files. Corpus-based studies have the potential to quantitatively answer research questions in musicology using a data-rich approach, thus providing a complementary strategy to methods in Musicology. Moreover, they provide examples of how Music Information Retrieval (MIR) methods can successfully contribute to other research areas, which has recently been discussed as an important issue of interdisciplinarity in MIR [1, 15]. In this paper we argue that corpus-based studies also contribute to important research questions within MIR, such as similarity estimation and genre classification.

Mauch and Dixon have carried out a corpus-based study of bar-length drum patterns in a collection of 48000 MIDI files gathered from the Internet, motivated by the observation that "comparatively little work in MIR has quantitatively examined rhythms in symbolic data" [11, p. 164]. The paper argues that the outcome of such a study might be of direct interest for musicians and musicologists. Conclusions from the detected repeated drum patterns in this collection are drawn on a general level, such as about the difference between language and music with regard to repetition. In this paper we argue that the outcome of a corpus-based study is even more meaningful to musicians and musicologists, if the questions addressed are meaningful for the specific corpus investigated. Hence, we demonstrate what questions on ragtimes exist among musicologists and music lovers, and why a corpus-based study on ragtimes is therefore of vital interest to musicologists, music lovers and MIR researchers.

In this paper we introduce a corpus of around 11000 ragtime MIDI files as the dataset for our study on syncopation patterns, the RAG [1] -collection. The MIDI files of this collection have been produced, organised and collected by many ragtime musicians and lovers all around the world and shared via the Internet. The collection has grown over many years, and an accompanying Ragtime Compendium has been created by Michael Mathew and colleagues with metadata about the corresponding ragtime compositions (such as title, year, composer, original publisher, website of the MIDI file) [10]. The RAG-collection is the product of a shared effort of ragtime enthusiasts and hence presents a heterogeneous collection of MIDI files of very different quality due to their different origins (such as original piano rolls that have later been translated into MIDI format, actual performances of ragtimes or encodings of the notated score). In this paper we undertake first steps to use the RAG-collection as a basis for investigating characteristics of the ragtime genre employing MIR methods.

The contribution of the paper is threefold. First, we give an overview over open questions on ragtime as a genre relevant to musicologists, MIR researchers, and ragtime lovers; we argue how MIR can productively contribute to answering these questions with a corpus-based study. Second, we investigate syncopation patterns on the RAG-collection as a means of understanding rhythmic characteristics of this genre. Specifically, we test and confirm musicological hypotheses about the occurrence of so-called *tied* and *untied* syncopation patterns in ragtimes during different time periods of *the ragtime era* (1890-1919). Furthermore we show that these patterns remained important for ragtimes in the period 1920-2012 and have been used even more extensively. Third, we tailor MIR tools for melody finding to the ragtime genre. By investigating syncopation patterns in ragtimes using a data-rich approach, we undertake a first step towards modelling typical characteristics of

[1] Ragtime Admirers Group

the ragtime genre, which is an important means for automatic genre classification based on similarity, contributing to a core topic in MIR.

2. SIGNIFICANCE OF A CORPUS-BASED STUDY ON RAGTIMES FOR MUSICOLOGY AND MIR

In this section we argue that a corpus-based study on ragtimes contributes to important issues in both MIR and Musicology regarding the question on what constitutes the genre of ragtime.

2.1 Open questions on ragtime as a genre

While most people associate *ragtime* with the piano music by Scott Joplin, James Scott and Joseph Lamb, ragtime music in fact comprises a wide range of vocal, instrumental, improvised and composed music with lots of different musical influences (such as difference dance music styles like march, two-step, habaneras, or tangos). Ragtime is considered a "hybrid from folk and written cultures" [5, p. 65]; as a consequence, ragtime is both investigated in the context of "folk", "popular" and "art" music [4, p. 142]. As of today, ragtimes present "thorny problems" to musical scholars [4], one of those crucial problems concerns the very identity of what is ragtime: "A major question that emerged in the course of this musical survey was how to determine what music could reasonably be considered ragtime" [2, p. xviii]. As we will argue later, this question is not only of scholarly interest, but also concerns the vital community of ragtime lovers all over the world.

One of the main characterizations of ragtimes is related to syncopation, as Edward Berlin points out: "At the core of the contemporary understanding of ragtime ... was syncopation. The question 'What is ragtime?' was asked throughout the period, and almost invariably explanations included a statement about syncopation" [2, p. 11]. However, there are ragtimes that do not contain syncopation, and there exist lots of syncopated music that is not considered ragtime. Hence, the role of syncopation for the ragtime genre needs still to be more specified, and additionally other features need to be described that contribute to ragtime as a genre. One example of investigating the specific role of syncopation in ragtime is that musical scholars have determined typical syncopation patterns in ragtime alongside with hypotheses on what patterns are most typical for a certain period of ragtime [2, 7], which we will address in this paper using computational approaches.

Closely related to the question of what constitutes a ragtime is the question about the different origins or musical sources that contributed or are still contributing to ragtime, and what subgroups within this broad genre can be identified. There exist different theories as to what has contributed to shaping the genre of ragtime, such as coon-songs, cakewalk, two-steps, or dance music of the Caribbean or South America (such as danzas, habaneras, and tangos) and music characteristics of African American origins. With regard to piano ragtimes, Berlin distinguishes at least three different subgroups: "piano renditions of ragtime songs; 'ragged' versions of pre-existing unsyncopated music; and original, dance-oriented ragtime compositions" [2, p. 63]. Furthermore he lists patriotic and folk tunes as well as classical music pieces as common musical sources that by *being ragged* have become members of the ragtime repertoire. In the U.S. many local Ragtime societies and festivals exist, and attempts have been made to distinguish stylistically different types of ragtimes dependent on their local origin – however, this has not yet been achieved according to Berlin [3].

Moreover, Berlin discusses that what has been considered ragtime by listeners in the past and present, has also been shaped by socio-cultural aspects: "My original search for a working definition of ragtime was almost dwarfed as I became immersed in the conflicts of its day, concerning ragtime's origins, racial content, relevance to American music, innovative features, potential for 'artistic' development, effects on cultural, moral, and physical well-being and the like" [3, p. xviii].

2.2 The significance of a corpus-based investigation of ragtimes for musicology

An investigation of ragtimes using a large digital corpus comprising ragtime pieces from the 1890s to present is necessary in order to overcome current biases about this genre due to a rather restricted perspective that comes from the musical pieces that have been mainly investigated so far. The years of roughly 1890s–1919 are considered as *the Ragtime era* when the genre was established, and for the general public a resurrection of the genre took place following the film "The Sting" in the 1970s. However, in fact ragtime music has been a lively music tradition also in the years in between. Edward Berlin argued that published anthologies of ragtimes over-represent "the best-known composers" (Joplin, Scott, Lamb) from the Ragtime era, and that therefore in ragtime research the stylistic traits of their music has overweighted stylistic traits of other ragtime compositions [2, p. 72]. Therefore, in order to get a more balanced view of the ragtime genre as a whole, he has carried out by hand a study of a random selection of ragtimes at the Lincoln Center. Studying a large digital ragtime corpus with computational tools is the logical next step of broadening the perspective on ragtime. Moreover, attempts to define possible subgroups in the broad range of existing ragtimes, provide a natural setting for research on genre using a computational data-rich approach.

Investigating rhythmic patterns in ragtime also contributes to research on variation in music. Variation is a fundamental trait in music and is linked to the experience of similarity in music, a central concept in MIR [14]. Most often, variation has been studied in the context of Variation sets in Western classical music, where one musical idea is widely exploited by the composer. However, as a means of establishing similarity in music, it seems more appropriate to study the relation between variation patterns and style, as suggested by Meyer [12, p. 3]: "Style is a replication of patterning". For ragtime there exist musicological hypotheses on what patterns are characteristic for the genre; the role of these patterns can be tested with computational

methods. In this paper we test hypotheses about the occurrence of a typical syncopation pattern in ragtime, which has been dubbed as *Leitmotiv* [2] of ragtime by musicologists [6]. We examine how often and in which variations with respect to the location within the bar, this pattern occurs within the RAG-collection.

2.3 The significance of a corpus-based investigation of ragtimes for MIR and ragtime lovers

The question on how to determine with computational methods the genre of a given digitized musical piece lies at the heart of MIR research. At the same time, it is all too often very unclear what a reasonable argument would be to classify a certain musical piece into a given genre category. The genre of ragtime provides an interesting case study of linking rhythmic and harmonic patterns that are discussed as being constituents of the genre to the occurrence of these patterns within a large digitized corpus of ragtime pieces.

Moreover, the question of what constitutes a ragtime is not only interesting from a scholarly perspective, but is of interest to a rather large international ragtime fan community (with ragtime festivals and societies [3] existing) that shares lots of digital collections on the Internet: "the ragtimers ... have clubs, magazines, and an international following. The subject owes an enormous amount to these individual collectors and dedicated players who have helped to preserve the repertoire and keep it alive. New rags are being composed and played all the time" [5, p. 65].

Piano rolls belong to the first sources that preserved ragtimes before they appeared as printed scores in sheet music. Many of these piano rolls have been later preserved as MIDI files, and along with many other compositions for which MIDI files have been created and shared via the Internet. We have now a collection of around 11000 MIDI files that have been recognized as ragtime by ragtime lovers. It is of vital interest, also to this community of ragtime fans, whether a given digital file might be considered as ragtime or into what subcategory of ragtime it might belong: "A discussion on what constitutes ragtime could fill several web pages" [10]. While socio-cultural aspects certainly contribute to constituting the ragtime genre, content-based MIR-methods can assist in investigating in how far musical characteristics contribute to the the identity of ragtime. Moreover, MIR-methods can be used to identify variants of the same ragtime composition in different MIDI file renditions existing on the Internet. Content-based MIR-methods hence would be of great use for this community, since metadata on ragtimes is rather problematic and not very reliable (i.e. ragtime composers have used different pseudonyms for different genres [10]). Hence, ragtimes provide an interesting application area for content-based methods on similarity in MIR.

[2] by metaphorically extending the original concept of *Leitmotiv* from recurring motifs that represent a certain person, idea etc. to recurring motifs that represent a genre

[3] e.g. http://www.westcoastragtime.com; http://indianarag.org; http://www.bohemragtime.com/en/act.html

Figure 1. (a) shows the syncopation pattern dubbed *121* in [7] and its different shifts within the bar, leading to *untied* syncopations in (b) and (c), and *tied* syncopations in (d) and (e). Behind the pattern scores we display the patterns used for matching: `I` denotes an onset, `O` denotes no onset, and `.` denotes a wild-card that matches everything. Every position represents sixteenth note position within a bar.

3. THE RAG-COLLECTION

The RAG-collection currently contains 11591 MIDI files of ragtimes gathered from the Internet. The Ragtime Compendium [10] is an accompanying database on the identification of individual ragtime compositions (year, title, composer, publisher) and whether or not a MIDI file exists. The Compendium lists around 15000 identified compositions (each entry corresponds to a unique ragtime composition), out of these 5600 pieces have MIDI files. Hence, the 11591 MIDI files contain lots of different MIDI renditions for the same composition. The compendium "has evolved over several years, starting from very humble beginnings and now, thanks to the help of many ragtime enthusiasts, it is probably the most complete listing that exists" [10].

4. A CORPUS-BASED STUDY ON SYNCOPATION

For the characterization of musical pieces as ragtimes, syncopation plays a major role, as described in section 2. To be able to distinguish between pieces that contain syncopations and that belong to the ragtime genre or not, more detailed investigations of syncopation patterns are necessary. Are ragtimes characterized by a higher amount of syncopation patterns than other genres, and/or are there specific syncopation patterns prevalent in ragtimes more than in other music styles? Musicologists [2, 7] discuss a number of syncopation patterns typically occurring in ragtimes of *the ragtime era* (1890s–1919), along with hypotheses on what patterns occur most often in early (until 1901) or late ragtimes (1902–1919) based on an extensive study of ragtime pieces from this era by hand.

In this paper we are testing these hypotheses about typical syncopation patterns with a computational approach, as a first step of exploring the role of syncopation in defining ragtimes. Specifically, we test Berlin's hypothesis that so-called *untied* syncopations are a typical characteristic trait of early ragtimes, while *tied* syncopations have been introduced in the late ragtimes starting roughly from 1902. Hence, many early ragtimes use *untied* syncopations exclusively, while tied syncopations became the predominant type of syncopation later [2, p. 128]. The difference between tied and untied syncopation is discussed with the

syncopation pattern shown in Figure 1(a), which has been called the *Leitmotiv* of ragtime music [6] and called pattern *121* in [7]. Shifting this pattern along the bar leads to either untied syncopation (Figures 1(b) and (c)) or tied syncopation (Figures 1(d) and (e)).

We were unable to find musicological hypotheses about the role of these syncopation patterns in ragtimes that have been composed after 1919 (following the ragtime era). Since the amount of ragtime pieces composed dropped significantly after 1919, one might hypothesize that by then the consolidation of the ragtime style has taken place. Ragtimes following this consolidation might therefore exhibit similar amounts of untied and tied syncopation as in the second half of the ragtime era (1902–1919). However, since the extensive use of syncopation patterns was new to the general musical audience at the beginning of the 20th century, one might also hypothesize that the amount of syncopation patterns increases over time due to a common habituation to syncopations. For instance, an increased use of syncopation in American popular music in general has been concluded from an empirical study in [9], using 437 short random samples from pieces dating back from 1890 to 1939. Yet a third hypothesis arises from Berlin's remarks on the erosion of ragtime's "most distinctive qualities" [2, p. 147] that started in the 1910s. The increased use of dotted rhythms observed in late ragtimes "reduced the need for syncopation in ragtime" [2, p. 148]. From this perspective, a decreased use of syncopation might be hypothesized for the following years. We take advantage of the RAG-collection containing ragtime pieces until 2012 in order to test the role of these syncopation patterns in ragtimes composed after the ragtime era.

4.1 Data preparation

For testing hypotheses on syncopation patterns, we first need to pre-process the MIDI files of the RAG-collection.

4.1.1 Melody finding

Within the ragtime genre most of the interesting rhythmical material is in the melody, hence in the case of a piano piece, this mainly concerns the right hand part of the piece. The accompaniment in the left hand is characterised by stable rhythmical patterns that follow the beat (also dubbed *oompah* bass line). Since the melody is not annotated explicitly in the MIDI files of the RAG-collection, we need to employ automatic melody extraction in order to split the melody from the accompaniment. We use a version of the *skyline* algorithm for melody finding [16]; this algorithm takes the highest sounding note when multiple notes sound simultaneously. To overcome that highest notes from the left hand are classified wrongly as part of the melody at sections where there is no melody, we set a lower limit: all notes below the middle C are classified as accompaniment.

We evaluated our skyline implementation on a small selection of 435 songs of the RAG-collection. These songs were selected automatically based on whether they were quantised, and had exactly two tracks. We ensured that there were no duplicate files or tracks. In most cases the first track was the melody track, but in a few cases the melody and accompaniment track were reversed. Hence, we checked whether both the lowest and the highest notes in the melody were higher than the lowest and highest notes in the accompaniment. If this was not the case, we examined the file by hand. When the two tracks did not clearly reflect a melody and accompaniment, we removed it from the selection. Next, we performed a melody extraction experiment by creating one track containing all notes, performing the melody finding, and comparing the result to the original melody track (our ground-truth melody). In some cases the ground-truth melody track contained chords, since we are mainly interested in rhythm, and did not tailor our skyline implementation for retrieving chords in the melody track, we removed all but the highest chord notes.

	P	R	F
Melody extraction	.930	.989	.958
Melody extraction w. dip-detection	.972	.984	.978

Table 1. The Precision, Recall, and F-measure of the melody extraction with and without *dip-detection*, evaluated on a 435-piece subset of the RAG-collection.

We compared the extracted melody to our ground-truth melody by calculating the precision, recall and F-measure for every piece, and averaged these numbers over the total selection of 435 songs. The results are displayed on the first row of Table 1, and can be considered good. The high recall indicates that almost all notes of the original melody are retrieved. The high precision shows in turn that we do not include a lot of non-melody notes into our automatically extracted melody.

Although the melody extraction could be considered good, manual inspection of some extracted melodies revealed that accompaniment notes were classified as melody notes typically at occasions where syncopation in the right hand occurs. In absolute numbers there were not many of these spurious melody notes. However, since they broke some of the syncopation patterns that we are interested in (see Figure 1), we extended our melody finding algorithm with a *dip-detection*: most of the misclassifications can be recognised by a large interval down followed by a large interval up. Therefore, after performing the skyline algorithm, we removed notes that were characterised by an interval down greater than 9 semitones followed by an interval up greater than 9 semitones.

We evaluate the melody extraction with dip-detection on the same subset of the RAG-collection, the results are displayed in the second row of Table 1. They show that the precision improves, while the recall hardly changes. Overall the skyline algorithm with dip-detection yields a near-perfect result on this subset of the RAG-collection.

4.1.2 Quantisation

To be able to analyse the rhythmical patterns in the ragtime melodies, the MIDI data needs to be quantised appropriately. Although most ragtime performers claim that ragtime should be played with *straight* eighth notes, we found

a large number of MIDI files that contained ragtimes performed with *swing*. Swing refers to a characteristic long-short subdivision of the beat that is generally considered a crucial aspect contributing to the quality of a jazz or pop performance (see [8] for further information). To be able to capture and represent all of these different subdivisions of the beat properly, we shift all onsets onto a grid of 12 equally spaced divisions per quarter note. To estimate how well a file can be aligned to the metrical grid, we store the quantisation deviation for every note (in MIDI ticks), and calculate the average. Because MIDI files can have different time resolutions, we divide the average deviation by the quarter note length as specified in the MIDI file; this *normalised average quantisation deviation* gives us a reasonable estimate of the quantisation error.

4.1.3 Selection of relevant MIDI files

Since we want to test hypotheses about the occurrence of syncopation patterns in ragtime compositions stemming from specific time periods, we need to match the metadata in the compendium to the MIDI files in the RAG-collection. This is done by comparing the file name with the title field in the compendium. To make the file name comparison more robust, we removed all spaces, commas, parentheses etc. before doing a case-insensitive comparison. Moreover, not all MIDI files are suitable for analysis: we require that a MIDI file has only one metre, which should be $\frac{2}{2}$, $\frac{2}{4}$, or $\frac{4}{4}$, and starting exactly at tick 0. Also, we found that pieces after 1920 have on average a higher normalised average quantisation deviation than pieces before 1920. To eliminate this bias, we removed all pieces that have a normalised average quantisation error above 2% from the corpus. For many entries in the compendium (a single entry stands for a unique ragtime composition) there exist multiple MIDI files in the collection. In order to not bias our study by using different MIDI renditions of the same ragtime composition, we select only one MIDI rendition per composition, namely the file with smallest average quantisation deviation.

We divided the data in two periods: *the ragtime era* period (1890–1919, 1806 pieces) and the modern period (1920–2012, 659 pieces). To be able to investigate changes in syncopation patterns in different time periods, we split pieces from the ragtime era into the pieces up to 1901 (253 pieces) and from 1902 onwards (1553 pieces).

4.2 Testing of syncopation hypothesis

For testing Berlin's hypothesis on syncopations in the ragtime era we calculate the proportion of tied and untied *121* syncopation patterns in the period 1894–1901 (253 pieces) and the period 1902–1919 (1553 pieces). All pitch information of the ragtime melodies is ignored, and we only examine the onsets of the melodies. The melodies are segmented in two bar segments that overlap exactly one bar.[4] Furthermore, based on the time signature, the patterns are scaled appropriately to match the predominant pulse. As a consequence, in $\frac{2}{2}$ and $\frac{4}{4}$ metres the patterns in Figure 1 (a)–(e) are matched at the quarter note level. We match all tied and untied patterns for all songs written before and after 1902, and average the results. The experiment was implemented using the functional programming language Haskell, and the code is available to the research community on request.

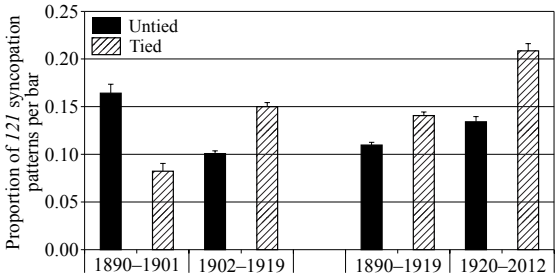


Figure 2. On the left side, the proportion of untied and tied *121* syncopation patters per bar averaged over all songs before and after 1902 is displayed. On the right side we show the distribution of tied and untied syncopation patterns for the ragtime era and for modern ragtimes. The error bars denote the standard error of the mean.

The results displayed on the left side of Figure 2 show that the number of tied patterns has almost doubled in the period 1902–1919, compared to the period 1890–1901. In the same period, the number of untied patterns has decreased with approximately one third. We examined if these differences were statistically significant by performing two Wilcoxon rank-sum tests, and we can confirm that the difference in mean before and after 1902 are statistically significant for the untied pattern, $p < 0.0001$, as well as for the tied pattern, $p < 0.0001$. [5]

Examining the difference in tied and untied patterns in the entire ragtime era (1890–1919, 1806 pieces) reveals that the differences are less pronounced (see the right side of Figure 2). The average *121* proportion per bar is 0.11 for untied and 0.14 for tied syncopation. We confirm that this difference is still statistical significant at the $\alpha = 0.01$ level with a Wilcoxon signed-rank test, $Z = -5.57, p < 0.0001$. If we examine the difference between the proportions of tied and untied patterns after the ragtime era (1920–2012, 659 pieces), we observe that the usage of syncopation patterns increases compared to the ragtime era. Hence, we cannot confirm the hypothesis that a consolidation of the ragtime style within the ragtime era led to the same amount of syncopation patterns after 1919 (except for the fact that tied syncopation occur more often than untied syncopation); neither can we observe a decreased use of these patterns, which would be in accordance with Berlin's hypothesis that the increased use of dotted rhythms in late ragtimes reduced the need for syncopation. Especially the proportion of tied syncopation increases significantly, and the difference between tied and untied syncopation is highly statistical significant, $Z = -8.62, p < 0.0001$. The increased use of these synco-

[4] This is necessary since the tied pattern in Fig. 1(e) crosses a barline.

[5] One might think that the large difference in sample size has affected the significance tests. We reassure the reader that the differences remain statistically significant if we take a random sample of 253 songs after 1902 for tied, $p < 0.0001$, and untied, $p < 0.0001$, patterns.

pation patterns in the modern era indicates that our second hypothesis on the habituation to syncopation by the general musical audience in the 20th century allowing an even stronger use of syncopation in ragtimes, should indeed not be rejected on the basis of the RAG-collection. However, since the RAG-collection is much more complete for the ragtime era than for the modern era, it is difficult to predict whether this result is representative for all existing modern ragtime pieces (including those for which no MIDI file exists yet).

5. CONCLUSION AND FUTURE WORK

In this paper we have provided an overview of open questions on ragtime as a genre, and have argued on how MIR-methods can productively contribute to answering these questions. As a first step for finding characteristic musical features of the ragtime genre, we have evaluated musicological hypotheses on syncopation patterns on the RAG-collection. We introduced the RAG-collection that evolved from the shared effort of the Internet community of ragtime lovers, and introduced a number of preprocessing steps to be able to parse the ragtime files of this collection for a corpus-based study; specifically, we tailored the skyline melody extracting algorithm successfully to ragtimes.

With our computational investigation of the occurrence of the *121*-pattern we were able to confirm Berlin's hypothesis on the increased use of tied syncopations in comparison to untied syncopations in the ragtime era after 1902. Furthermore, by extending our investigation to modern ragtimes, we were able to show that these patterns remain important in ragtime also after 1920. We have found no evidence that the amount and kind of syncopation after the ragtime era stays stable, rather we observe an increased use of these patterns, especially of tied syncopations. The increase of syncopation in ragtime over time concluded from our analysis of 2465 compositions provides an interesting complementary finding to the increase of syncopation in American popular music in 1890-1939 concluded in [9] from short samples extracted from 437 pieces.

In the near future, we will quantitatively test all occurring kinds of syncopation patterns in ragtime and determine the proportion of the *121*-pattern among these patterns. This will contribute to evaluating whether the *121*-pattern can indeed be considered such a typical pattern in ragtime, providing a promising candidate for typical ragtime features for classification tasks in MIR. For modelling syncopation patterns as a characteristic musical feature of ragtime, another next step is to consider the location of these patterns within specific form sections, such as discussed in [7] in order to establish different typologies of ragtimes. Apart from rhythmic features, musicologists argue that there exist also typical harmonic patterns in ragtime, such that the study of chord progressions is a next logical step. The strengthening of the syncopation by placing a chord rather than only a melody note at the right hand part provides yet another candidate of syncopation patterns typically for ragtimes that we are going to evaluate.

For distinguishing syncopation patterns typical for ragtimes and testing their discriminative power for automatically classifying ragtimes, we plan to extend our database with jazz, pop and rock pieces, which constitute related genres for which syncopation has been claimed being relevant [9, 13]. In a study of syncopation in rock, Temperley argues that a "full understanding of rock requires consideration of social and cultural aspects". This certainly applies also to ragtime music, and by employing content-based MIR-methods for genre classification, we plan to contribute to the debate on how far a musical genre is both a sociological construct and is rooted in musical characteristics that can be extracted from the musical content.

Acknowledgments. We acknowledge the efforts of many ragtime enthusiasts that have contributed to the RAG-collection and the accompanying Ragtime Compendium. Specifically, we thank Michael Mathew, who has kindly provided us with the collection and the compendium. We thank F. Wiering and M.E. Rodríguez-López for providing valuable comments on an earlier draft on this text. 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.

6. REFERENCES

[1] J. J. Aucoutourier & E. Bigand: "Mel Cepstrum & Ann Ova: The difficult dialog between MIR and Music cognition." *ISMIR-Proceedings*, pp. 397–402, 2012.

[2] E. A. Berlin: "Ragtime. A Musical and Cultural History." University of California Press, 1980.

[3] E. A. Berlin: Review of "That American Rag: The Story of Ragtime from Coast to Coast", by David A. Jasen and Gene Jones. *American Music*, Vol. 19, No. 4, pp. 474–476, 2001.

[4] S. DeVeaux: Review of "Ragtime: Its History, Composers, and Music" by John Edward Hasse; "Reflections and Research on Ragtime" by Edward A. Berlin. *Ethnomusicology*, Vol. 32, No. 2, pp. 142–145, 1988.

[5] P. Dickinson: "The Achievement of Ragtime: An Introductory Study with Some Implications for British Research in Popular Music". *Proceedings of the Royal Musical Association*, Vol. 105, No. 2, pp. 63–76, 1978-1979.

[6] F. Gillis: "Hot Rhythm in Piano Ragtime". *Music in the Americas*, pp. 91–104, 1967.

[7] I. Harer: *Ragtime. Versuch einer Typologie.* Verlag Hans Schneider, Tutzing, 1989.

[8] H. Honing & W.B. de Haas: "Swing Once More: Relating Timing and Tempo in Expert Jazz Drumming", *Music Perception*, Vol. 25, No. 5, pp. 471–476, 2008

[9] D. Huron & A. Ommen: "An Empirical Study of Syncopation in American Popular Music, 1890-1939", *Music Theory Spectrum*, Vol. 28, No. 2, pp. 211–231, 2006.

[10] M. Mathew: "A Ragtime Compendium" http://ragtimecompendium.tripod.com/

[11] M. Mauch & S. Dixon: "A corpus-based study of rhythm patterns", *ISMIR-Proceedings*, pp. 163–168, 2012.

[12] L.B. Meyer: *Style and Music: Theory, History, and Ideology*, University of Chicago Press, 1989.

[13] D. Temperley: "Syncopation in Rock: A Perceptual Perspective", *Popular Music*, pp. 19–40, 1999.

[14] A. Volk, W.B. de Haas, & P. van Kranenburg: "Towards Modelling Variation in Music as a Foundation for Similarity", *Proceedings of the 12th ICMPC*, pp. 1085-1094, 2012.

[15] A. Volk, & A. Honingh: "Mathematical and computational approaches to music: challenges in an interdisciplinary enterprise" *Journal of Mathematics and Music*, Vol. 6, No. 2, pp. 73–81, 2012.

[16] A.L. Uitdenbogerd & J. Zobel: "Manipulation of music for melody matching." *In Proceedings of the ACM Multimedia Conference*, pp. 235-240, 1998.

A COMPUTATIONAL COMPARISON OF THEORY AND PRACTICE OF SCALE INTONATION IN BYZANTINE CHANT

Maria Panteli[1] **& Hendrik Purwins**[2,3]
[1]Department of Computer Science, University of Cyprus, Cyprus
[2]Neurotechnology Group, EE & CS, Berlin Institute of Technology, Germany
[3]Sound and Music Computing Group, Aalborg University Copenhagen, Denmark
{m.x.panteli, hpurwins}@gmail.com

ABSTRACT

Byzantine Chant performance practice is quantitatively compared to the Chrysanthine theory. The intonation of scale degrees is quantified, based on pitch class profiles. An analysis procedure is introduced that consists of the following steps: 1) Pitch class histograms are calculated via non-parametric kernel smoothing. 2) Histogram peaks are detected. 3) Phrase ending analysis aids the finding of the tonic to align histogram peaks. 4) The theoretical scale degrees are mapped to the practical ones. 5) A schema of statistical tests detects significant deviations of theoretical scale tuning from the estimated ones in performance practice. The analysis of 94 echoi shows a tendency of the singer to level theoretic particularities of the echos that stand out of the general norm in the octoechos: theoretically extremely large scale steps are diminished in performance.

1. THE OCTOECHOS AND THE CHRYSANTHINE THEORY

Byzantine Chant is the Christian liturgical song of the Eastern Roman Empire (*Byzantium*) that gradually emerged from the Roman Empire from the 4th century on. Byzantine Chant has been the dominant liturgy of the Eastern orthodox Christianity. Referring to various theoretic accounts on Byzantine Chant, Zannos in [18] argues that '*none of them can be said to correspond with contemporary empirical study*'.

The main analysis tool used was a pitch class profile [17] with high bin resolution, extracted from audio recordings with the aid of specifically designed algorithms. These were applied on a music collection of 94 Byzantine Chants labeled after the scale they are performed in. The overall behavior and consistency of empirical scale degree tuning was computed and contrasted to theory through a series of tests and experiments.

According to Mavroeidis [12] and Thoukidid [16], a mode (singular: *echos*, plural: *echoi*) in Byzantine Chant is defined by the following five characteristics: 1) the *scale degree steps (SD)* between consecutive scale degrees, 2) the most prominent scale degrees (two or three scale degrees out of which I - III and I - IV scale degrees are the most reoccurring pairs), 3) a short introductory phrase that marks the reference tone, 4) the cadences in the middle and the end of a phrase, and 5) the modulations (alterations) applied to particular scale notes depending on whether they are reached by an ascending or a descending melody. Subject to a reform in the 1880s in particular concerning the sizes of the intervals, the Chrysanthine notation method (from the 1820s) is used in the official chant books of the Greek Orthodox Church up to now [11]. In Chrysanthine theory, the octave is divided into 72 equal partitions, each of 16.67 cents (singular: *morio*, plural: *moria*). The scale degree steps are measured in multiples of a morio (cf. Table 1). According to this theory there are in total eight basic echoi, a system also referred to as *octoechos* ('eight' + 'mode'). These eight modes occur in pairs of *authentic* and corresponding *plagal* modes: *First Authentic, Second Authentic, Third Authentic, Fourth Authentic, First Plagal, Second Plagal, Grave, Fourth Plagal*. The plagal mode has a different reference tone (tonic) than its authentic counterpart, usually a perfect fifth lower than the one of the authentic mode [1] but it may share the same scale step sequence. Furthermore, both differ in melodic characteristics. The scale degree steps may vary according to the chant genre (*Heirmoi, Stichera, Papadika*) [12, 16]. Our study is limited to the basic and simplest echos scales [2]. We will not consider the fact that scale degree steps of an echos can be modulated (altered) based on the melodic characteristics of a chant or other criteria [12, 16] (cf. Section 4).

2. MUSIC CORPUS

The music corpus analysed in this study consists of recorded monophonic songs from the album series of Protop-

[1] The *Grave* shares the same scale degree steps with the Third Authentic as well as the same reference tone.

[2] Chants of Heirmoi genre of Second, Heirmoi of Second Plagal, Papadika of Grave, Stichera and Papadika of Fourth, and Papadika of Fourth Plagal use different scale steps than the eight basic echoi thus omitted in this study (cf. Table 1).

Echos	Chant Type	Step (in Moria) between Scale Degrees						
		I	II	III	IV	V	VI	VII I
First	All chants	10	8	12	12	10	8	12
First Plagal	All chants	10	8	12	12	10	8	12
Second	Stichera/Papadika	8	14	8	12	8	14	8
Second Plagal	Stichera/Papadika	6	20	4	12	6	20	4
Third	All chants	12	12	6	12	12	12	6
Grave	Heirmoi/Stichera	12	12	6	12	12	12	6
Fourth	Heirmoi	8	12	12	10	8	12	10
Fourth Plagal	Heirmoi/Stichera	12	10	8	12	12	10	8

Table 1. The scale structure of the eight modes (octoechos) measured in multiples of a morio. The two tetrachords are indicated.

saltes Georgios Kakoulides [8], Protopsaltes Ioannis Damarlakis [3], Protopsaltes Panteleimon Kartsonas [9], and Protopsaltes Dimitrios Ioannidis [7]. From the total of 94 recordings, 13 are in the First Authentic echos, 15 in First Plagal, 6 in Second Authentic, 6 in Second Plagal, 18 in Third Authentic, 9 in Grave, 10 in Fourth Authentic and 17 in Fourth Plagal.

3. COMPUTATIONAL ANALYSIS PROCESS

To study scale degree pitch empirically, pitch modulo octave histograms are investigated. Built on pitch histograms, pitch class profiles have been applied to detect key and tone centres in classical Western music [6, 14]. The authors in [2] adapted the latter approach to Raag recognition. Bozkurt [1] proposed a method to extract the tuning of a scale, applied to Turkish maqam. Moelants et al. [13] introduced a peak picking heuristics to extract the scale tuning from a modulo octave pitch histogram of African scales. Serrà et al. [15] used pitch class histograms to study scale tuning in Hindustani and Carnatic music, investigating whether this music follows equal temperament rather than just intonation.

The procedure followed in this article is summarized in Figure 1. First, the pitch ($f0$) trajectory is extracted from each audio recording. A pitch histogram is computed, compressed into one octave and smoothed. Peaks are extracted from the histogram and are employed in tonic detection. The pitch trajectory is then aligned to the estimated tonic. For recordings of a particular echos, the aligned pitch trajectories are used to compute the *echos histogram*. Peaks are then detected in the echos histogram, and peak locations are mapped to theoretical scale degree pitches. Pitches around the selected peak locations are used to determine a neighbourhoud of pitches around the empirical scale degrees. Finally, a sequence of statistical tests is used to compare the estimated practical scale tuning with the theoretical ones.

3.1 Pitch Trajectory via F0 Detection

In the current study we use the f_0 estimation Yin algorithm [4]. Considering the melodic characteristics of the analysed music as well as the particularities of the singing voice, the following post processing filters were designed: noise, silent gaps, octave/fifth error. As a result, a trajectory of N_m estimated pitches $p = (p_1, \ldots, p_{N_m})$ is generated for each recording m. Three experts evaluated the estimated pitch trajectory of 20 excerpts, presented as a sythesized version of the original melody, with an average 4.2 of 5(max).

3.2 Pitch Histogram via Kernel Smoothing

For the vector p, we define the *pitch histogram*

$$c_k^{p,b} = \Sigma_{n=1}^{N} q_r(\frac{p_n - b_k}{h}) \quad (1)$$

with the *rectangular kernel function*

$$q_r(u) = \begin{cases} 1 & \text{if } |u| \leq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

for K *center bins* $b = (b_1, \ldots, b_K)$, being multiples $b_k = (k-1).h + p^*$ of bin widths h and an offset p^*. The Chrysanthine theory divides the octave into 72 equal partitions and we multiply this division by 3 arriving at $K = 216$ center bins in the range of one octave, thereby yielding sufficient bin resolution and robustness. The choice of h is critical for the subsequent stage of peak picking, since a too high h can eliminate relevant peaks whereas a too small h can create spurious peaks in the pitch histogram.

Although a large h increases the smoothness of the pitch histogram, discontinuities in the histogram remain. These discontinuities are artefacts due to the partitioning of the pitches in a discrete set of predefined bins. The sharp-edged rectangular kernel function in (1) is replaced by a smooth *Gaussian kernel* function yielding the equation

$$c_k^{p,b} = \Sigma_{n=1}^{N} \frac{1}{\sqrt{2\pi h^2}} e^{\frac{-\|p_n - b_k\|^2}{2h^2}}. \quad (3)$$

The selection of the appropriate smoothing parameter h is guided by the task the histogram is used for. For the estimation of the SD tuning a relatively high smoothing factor is employed to avoid spurious peaks in a too detailed histogram. To determine an adequate h, the following assumption is made: Byzantine theory defines 4 moria (67 cents) as the smallest SD interval. Choosing the quartertone ($\delta_{\text{min}} = 50$ cents) as the smallest acceptable distance between two histogram peak locations allows for a margin

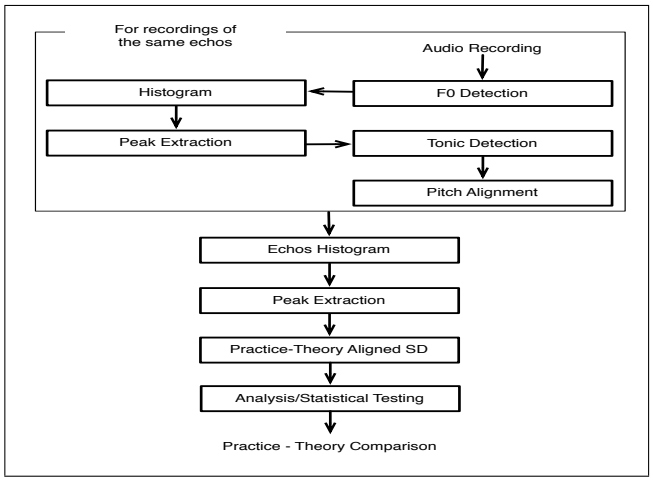


Figure 1. Flow diagram of analysis process of a recording yielding a theory-practice comparison of the scale tuning (SD=Scale Degree).

for investigating the deviations between theory and practice. Experimenting with h, this assumption is satisfied when h is set to 18 cents.

3.3 Peak Extraction

Using a peak extraction algorithm, from the smoothed pitch class histogram $c^{p,b}$, U peaks (λ, π) can be detected consisting of *peak locations* $\lambda = (\lambda_1, \ldots, \lambda_U)$ and *peak amplitudes* $\pi = (\pi_1, \ldots, \pi_U)$. In [13] the authors propose a number of heuristics to be used in peak picking, such as the size and height of peaks and intervals between peaks. Our proposed algorithm iteratively chooses the peak position $\lambda_u = b_{k_u}$ with maximum peak height $\pi_u = c_{k_u}$, then removing the potential location candidates in a $\pm\delta_{\text{min}}$ neighborhood around the selected peak position, to choose the next peak until U peaks are picked. U is defined as follows: According to Byzantine theory each echos has at least 7 scale notes. In addition, Byzantine theory knows of note alterations. To account for further intonation variants in practice, we set $U = 12$. $\delta_{\text{min}} = 50$ cents is defined as the minimum neighborhood.

3.4 Tonic Detection

To make the first bin b_1 corresponding to the tonic, the histogram has to be circularly (modulo K) shifted by p_0, the pitch of the tonic. The authors in [5] calculated the cross-correlation between all scale degree prototypes and all circularly shifted versions of a smoothed histogram. The pitch shift p_0 that gives the maximum cross-correlation is the estimated tonic and is used to circularly shift the second histogram. According to theory, the tonic of the Byzantine echos considered in this study is stated at the end of the phrase. Our tonic detection algorithm computes the pitch of the last phrase note from the onset and frequency information assuming that 1) the last note in the recording is the last note of the melodic phrase and that 2) the final phrase note lasts for at least half a second.

To compensate with inaccuracies [5], the tonic detection algorithm integrates pitch information from a set of maximum three onsets detected at the end of the phrase. Conditions apply to decide which of the three onsets correspond to the last phrase note considering vibrato and ornamentation (the main incaccuracies in this case), and the pitch is then estimated as an average of these. As a final step, the estimated pitch of the last phrase note is refined to the closest histogram peak that represents the closest empirical scale degree. Three experts found a wrongly automatically estimated tonic in 2 out of 20 excerpts.

3.5 Practice - Theory Comparison

For recordings m, $1 \leq m \leq M_i$, of echos i, the pitch trajectories $p^m = (p_1, \ldots, p_{N_m})$ are aligned to the estimated tonic and the *echos histogram* $C_i = c_k^{p^{1 \leq m \leq M_i}, b}$ is computed. The peak extraction algorithm applied on C_i yields the peak locations $\lambda = (\lambda_1, \ldots, \lambda_U)$ and peak amplitudes $\pi = (\pi_1, \ldots, \pi_U)$ of echos i.

3.5.1 Practice - Theory Aligned Scale Degrees

The normalized theoretical scale degrees are defined as a set of scale degree pitches (locations) $\nu^\theta = (\nu_1^\theta, \ldots, \nu_L^\theta)$, with normalization $\nu_0^\theta = 0$ (in cents) and $L = 7$[3]. From the preselected peaks (λ, π) of echos i, with pitches $\lambda = (\lambda_1, \ldots, \lambda_U)$ and amplitudes $\pi = (\pi_1, \ldots, \pi_U)$ we estimate the practical scale degree pitches $\nu^\pi = (\nu_1^\pi, \ldots, \nu_L^\pi)$. First, only those peaks (λ, π) are selected that correspond to theoretic scale notes ν^θ, i.e. those pitches λ_u that lie closer to and within a d distance, from a ν_l^θ, yielding the pitch vector ν. The distance d is set to 150 cents ($\frac{3}{4}$ of a whole tone) to allow enough margin of deviation between empirical peaks and theoretical scale notes. If two or more pitches λ_u fulfill this condition, the peak with highest amplitude π_l is selected, since it corresponds to a more prominent note such as a scale note. The estimated scale degree pitch $\hat{\nu} = (\lambda_{t_1}, \ldots, \lambda_{t_{L'}})$ is defined by $t_l = \arg_{|\lambda_t - \nu_l| \leq d} \max(\pi_t)$. If no pitches λ_u lie within a d range around the theoretical pitch ν_l^θ, t_l is not defined and the estimated scale $\hat{\nu}$ has less degrees than the theoretical scale ν^θ, $(1 \leq L' \leq L)$.

From the estimated scale $\hat{\nu}$ and for each recording m, $1 \leq m \leq M_i$, of echos i, pitches $\hat{p_l}^m$ are selected from the trajectory $p = (p_1, \ldots, p_{N_m})$ that lie within 2-moria distance of the estimated scale degree pitches $\hat{\nu}_l$, $1 \leq l \leq L'$. The 2-moria distance defines half the size of the smallest theoretical scale interval (cf. Table 1). Statistical testing is applied to check whether pitches $\hat{p_l}^{1 \leq m \leq M_i}$ are derived from distributions with mean equal to the theoretical scale degree pitches ν_l^θ for $l = 1, \ldots, L'$. For the pitches $\hat{p_l}^m$ and theoretical scale degree ν_l^θ, the *Mean Deviation*, *mean*$(\hat{p_l}^m - \nu_l^\theta)$ is also computed.

3.5.2 Statistical Testing

To assess the deviation between theoretic and practical scale degree pitches and steps, we apply a chain of tests as an

[3] Exceptions with $L = 8$ theoretical scale degree pitches exist in some variations of echoi not included here.

analytical instrument. As a first step, the Shapiro-Wilk test is applied, to determine whether the scale degree pitches ${\hat{p}_l}^{1\leq m\leq M_i}$ are normally distributed across all M_i instances of the same echos i. If the p value is above significance level $\alpha_n = 0.05$, we assume normal distribution and apply the t-test to ${\hat{p}_l}^{1\leq i\leq I}$. The t-test hypothesis is formulated that for echos i, the l-th estimated scale degree pitches ${\hat{p}_l}^{1\leq m\leq M_i}$ are derived from a distribution with mean equal to the l-th theoretical scale interval ν_l^θ. In case the Shapiro-Wilk rejects the normality hypothesis the Wilcoxon Signed-Rank test is applied instead. The significance level α_i for an individual test is set based on the *Bonferroni correction*; since $n = 48$ individual tests are applied and $\alpha = 0.05$ defines the confidence interval of the whole family of tests, each hypothesis is tested at the significance level $\alpha_i = \frac{\alpha}{n} = \frac{0.05}{48} \approx 0.001$.

If $p < \alpha_i$, for the probability p of observing $t_{i,l}$ under the null hypothesis, we reject the null hypothesis and conclude that theoretical scale degree pitch deviates significantly from the practical scale degree pitch. In addition, the histogram around an empirical scale degree pitch could be characterised by parameters such as variance, skewness, kurtosis, following [10].

4. RESULTS

The pitch histograms of all recordings of all echoi can be found in Figure 2. The tuning of the Byzantine scales has been investigated by comparing the pitches of empirical and theoretical scale notes. A series of statistical test was employed to determine for which scale notes practice and theory of tuning deviate. For all recordings of a given echos, the pitches $\hat{p}$ around the estimated scale degrees $\hat{\nu}$ (at the peaks of the histogram) are gathered. The Shapiro Wilk test on normality performed for pitches $\hat{p}$ showed that the normality hypothesis is rejected for all estimated scale degree pitches. The Wilcoxon Signed-Rank Test (W-test) with significance level $\alpha_i = 0.001\%$ (based on the Bonferroni correction) is therefore applied to test the null hypothesis that the median of the empirical pitches is the same as the theoretical pitch of a particular scale degree in a particular echos.

Table 2 reveals that the majority of scale degrees of all echoi differ significantly from theory. The null hypothesis was not rejected for the scale degrees V of First, VII of First Plagal, VI of Second Plagal, and II of Third and Grave. The Second, Fourth and Fourth Plagal echoi have all their empirical scale degrees significantly deviating from theory. The VI. scale degree of the First and First Plagal has a relatively large mean deviation value with negative sign, i.e., the empirical scale degree pitch is smaller than the theoretical one. According to theory, alterations of these echoi apply that diminish particularly the VI. scale degree when the melody is descending. Other large mean deviations appear for the VII. scale degree of Second Authentic as well as the III. and VII. scale degrees of Second Plagal. These scale degrees are reached by relatively large theoretical scale intervals; the VII scale degree of Second Authentic is reached by the VI-VII scale step of 14 moria whereas the III. and VII. scale degrees of Second Plagal are both reached by a scale step of 20 moria (cf. Table 1). The empirical scale degrees appear with negative deviation, i.e. large scale intervals are diminished in performance. Fourth is the only echos, in which the first tetrachord is extended by two moria comparised to the other echoi. In practice, the V. scale degree of Fourth Authentic is significantly diminished with respect to the theoretical scale degree pitch. An interpretation could be that the singer tends to diminish the abnormally high tetrachord pitch of this echos. In the same echos, also scale degree step VII-I (theoretically 10 moria) tends to be diminished towards the more common step VII-I of 6 moria.

One may argue that the assignment of empirical peaks to the nearest theoretical peak may introduce a dependency between the theoretical and practical peak positions and therefore bias the test. However this bias is limited due to the following reasons: An informal inspection reveals that the empirical peaks are relatively close to the theoretical peaks, and, in all but few exceptions, there are exactly seven empirical peaks that correspond to the theoretical scale degrees I-VII.

5. CONCLUSION

In this paper, a new method has been introduced to empirically study the tuning of scale degrees. The method has been used to investigate to what degree performance practice of Byzantine Chant follows the widely known Chrysanthine theory. The theoretic hypotheses have been tested on a corpus of recordings of chants of the octoechos. A combined method of pitch estimation with appropriate post filtering has been applied to the recordings. Among the novel methods proposed here are the histogram computation algorithm that comprises the use of Gaussian kernel and suitable tuning of the smoothing factor. The analysis gives support to the conjecture that the singer levels the extreme step and scale degrees particularities within the octoechos in performance practice of Byzantine Chant. The methodology introduced here has applications to a wide range of oral music traditions.

6. ACKNOWLEDGEMENTS

H. P. was supported in part by the German "Bundesministerium für Bildung und Forschung" (BMBF), Grant BFNT, No. 01GQ0850. We are grateful to Daniel Bartz from Berlin Institute of Technology for helpful discussion and proof reading the manuscript. Thanks to the Music Technology Group at Universitat Pompeu Fabra, Barcelona.

7. REFERENCES

[1] B. Bozkurt. An Automatic Pitch Analysis Method for Turkish Maqam Music. *Journal of New Music Research*, 37(1):1–13, March 2008.

[2] P. Chordia and A. Rae. Raag recognition using pitch-class and pitch-class dyad distributions. In *Proceedings of ISMIR*, pages 431–436, 2007.

Echos	Feature	Scale Degrees					
		II	III	IV	V	VI	VII
First (13)	p (W-test)	0	0	0	0.0032	0	0
	Mean Deviation	−16.56	11.25	10.45	0.84	−43.89	5.79
First Plagal (15)	p (W-test)	0	0	0.0001	0	0	0.8767
	Mean Deviation	28.06	−5.65	−0.91	4.98	−50.06	−0.06
Second (7)	p (W-test)	0	0	0	0	0	0
	Mean Deviation	5.79	−5.56	−6.55	10.36	20.95	−49.64
Second Plagal (7)	p (W-test)	0	0	0	0	0.1592	0
	Mean Deviation	11.20	−44.32	5.06	10.79	0.47	−138.47
Third (18)	p (W-test)	0.0048	0	0	0	0	–
	Mean Deviation	0.45	−21.70	−11.19	−5.48	−11.16	–
Grave (8)	p (W-test)	0.0233	0	0	0	0	–
	Mean Deviation	−0.44	−27.81	−11.05	−17.42	−10.53	–
Fourth (11)	p (W-test)	0	0	0	0	0	0
	Mean Deviation	5.27	5.58	−5.21	−39.14	11.13	38.85
Fourth Plagal (16)	p (W-test)	0	0	0	0	0	0
	Mean Deviation	22.06	11.17	16.63	17.86	17.00	22.25

Table 2. Significance of scale degree pitch deviation between practice and theory for all echoi (number of instances in brakets). Since all echoi are aligned to pitch 0 for scale degree I, only II-VII are shown. For scale degrees II-VII the p-value of the test statistic and the mean pitch deviation (in cent) between practice and theory are indicated. The p-value of the Wilcoxon Signed-Rank test (W-test) is denoted. Zero p-values correspond to values smaller than 10^{-4}. Practice-theory deviations (cf. Table 1) greater than two moria are colored and discussed in the text.

[3] I. Damarlaki. *Pws tha mathw na psallw praktika [Learn how to perform the Byzantine chants] [Audio CD]*. Polychronakis, Crete, 1999.

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

[5] A. C. Gedik and B. Bozkurt. Evaluation of the makam scale theory of arel for music information retrieval on traditional turkish art music. *Journal of New Music Research*, 38(2):103–116, 2009.

[6] E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, 2006.

[7] D. Ioannidis. *H theoria tis Byzantinis mousikis stin praxi [Theory of Byzantine music in practice]*. Ioannidis Dimitrios, Athens, 2005.

[8] G. I. Kakoulidis. *Pws na vriskoume praktika tous okto ihous tis Byzantinis mousikis [How to estimate empirically the eight modes of byzantine music]*. Athens, 1999.

[9] P. Kartsonas. *Praktiki Methodos Ekmathisis tis Psaltikis Tehnis [Practice Methods for Learning Byzantine Chant] [Audio CD]*. The Hunt of St. George Mount Athos.

[10] G. K. Koduri, J. Serrà, and X. Serra. Characterization of intonation in carnatic music by parametrizing pitch histograms. In *Proceedings of ISMIR*, pages 199–204, 2012.

[11] K. Levy and C. Troelsgå rd. *Byzantine chant*. Oxford University Press, accessed August 24, 2011, http://www.oxfordmusiconline.com/subscriber/article/grove/music/04494, 2011.

[12] M. Mavroeidis. *Oi mousikoi tropoi stin Anatoliki Mesogeio [The musical modes in East Mediterranean]*. Fagotto, Athens, 1999.

[13] D. Moelants, O. Cornelis, and M. Leman. Exploring African Tone Scales. In *Proceedings of ISMIR*, pages 489–494, 2009.

[14] H. Purwins, B. Blankertz, and K. Obermayer. A new method for tracking modulations in tonal music in audio data format. In *Proceedings of IJCNN*, volume 6, pages 270–275, 2000.

[15] J. Serrà, G. K. Koduri, M. Miron, and X. Serra. Assessing the tuning of sung indian classical music. In *Proceedings of ISMIR*, pages 157–162, 2011.

[16] A. Thoukidid. *Kripida Byzantinis mousikis [Foundations of Byzantine music]*. Kykkos Church, Cyprus, 2003.

[17] G. Tzanetakis, A. Ermolinskyi, and P. Cook. Pitch Histograms in Audio and Symbolic Music Information Retrieval. *Journal of New Music Research*, 32(2):143–152, 2002.

[18] I. Zannos. Intonation in Theory and Practice of Greek and Turkish Music. *Yearbook For Traditional Music*, 22(1990):42, 1990.

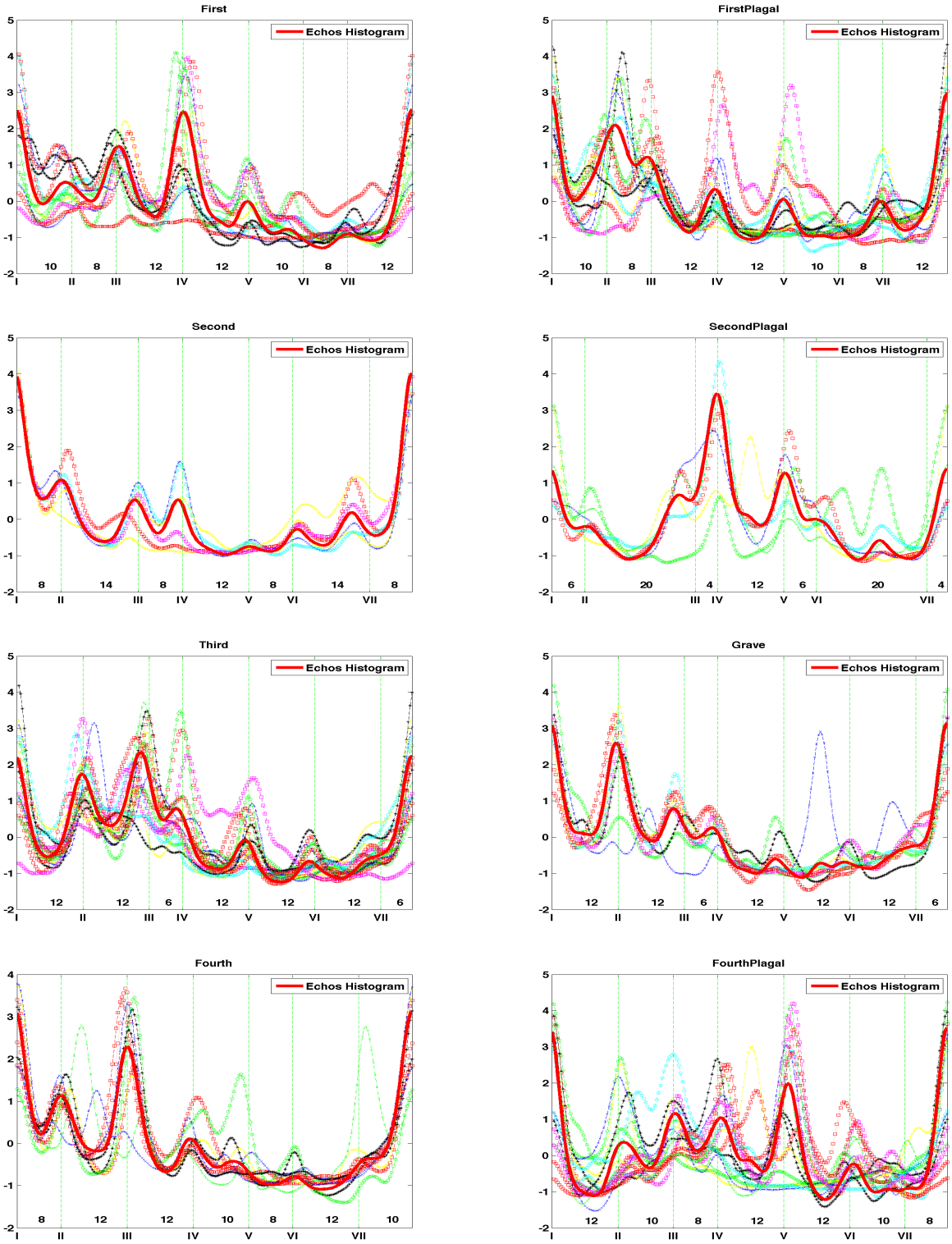


Figure 2. For all recordings of all echoi, the pitch histograms are displayed. The vertical lines indicate pitches of scale degrees according to Chrysanthine theory [16]. The y axis represents the normalized histogram count. The bold red lines represent the echos histogram computed from pitch trajectories across all recordings of the same echos.

SCORE INFORMED TONIC IDENTIFICATION FOR MAKAM MUSIC OF TURKEY

Sertan Şentürk, Sankalp Gulati, Xavier Serra
Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
{sertan.senturk, sankalp.gulati, xavier.serra}@upf.edu

ABSTRACT

Tonic is a fundamental concept in many music traditions and its automatic identification should be relevant for establishing the reference pitch when we analyse the melodic content of the music. In this paper, we present two methodologies for the identification of the tonic in audio recordings of *makam* music of Turkey, both taking advantage of some score information. First, we compute a prominent pitch and a audio kernel-density pitch class distribution (KPCD) from the audio recording. The peaks in the KPCD are selected as tonic candidates. The first method computes a score KPCD from the monophonic melody extracted from the score. Then, the audio KPCD is circular-shifted with respect to each tonic candidate and compared with the score KPCD. The best matching shift indicates the estimated tonic. The second method extracts the monophonic melody of the most repetitive section of the score. Normalising the audio prominent pitch with respect to each tonic candidate, the method attempts to link the repetitive structural element given in the score with the respective time-intervals in the audio recording. The result producing the most confident links marks the estimated tonic. We have tested the methods on a dataset of *makam* music of Turkey, achieving a very high accuracy (94.9%) with the first method, and almost perfect identification (99.6%) with the second method. We conclude that score informed tonic identification can be a useful first step in the computational analysis (e.g. expressive analysis, intonation analysis, audio-score alignment) of music collections involving melody-dominant content.

1. INTRODUCTION

Pitch relationships in pitch space and time constitute the fundamental building block of musical melody. In many musical styles, there is the concept of "tonic," which acts as the reference tuning pitch for the melody. The interrelations between the tonic and other pitches establish a hierarchical organisation, which is highly related to the perception, cognition and anticipation of music [9]. The automatic identification of the tonic of a piece is a musically relevant step in the computational analysis of many melodic characteristics.

Nevertheless, the tonic concept encompasses different meanings and characteristics within the cultural, musicological, acoustic (and linguistic) context of different music traditions. For example, in some music traditions, the tonic of a performance is changed according to instrument characteristics, personal preferences or for historical relevance. In such cases tonic identification is necessary to establish the reference pitch to carry further computational tasks such as tuning analysis, intonation analysis and melodic structure recognition. For systems that aim to analyse the melodic content of such musics, knowledge-based tonic identification approaches may be required [7, 14, 15].

Pitch distributions (PDs) and "octave-wrapped" pitch class distributions (PCDs) are commonly used for analysis of tonic and pitch organisation. Krumhansl and Shephard [11] used 12-dimensional PCDs to study the tonal organisation of euro-genetic musics. PCDs are also used for relevant tasks such as key detection and chord recognition [8, 16] for euro-genetic musics.

For musical styles involving microtonality, the pitch space must be extended beyond 12-dimensions to model, analyze and predict the melodic properties of the studied music [3, 4, 7]. Gedik and Bozkurt [7] propose a method for tonic and *makam* recognition for *makam* music of Turkey (MMT). The method generates a histogram based, fine-grained pitch distribution (FPD) for the test audio, and for each *makam* from the annotated audio recordings. Given the *makam* of the test audio, the first bin in the template FPD is assigned to an arbitrary frequency below audio FPD such that the FPDs do not overlap initially. Then, the method compares the template FPD with the audio FPD by shifting the template FPD at each step. In the best matching shift, the frequency of the tonic of the template is labeled as the estimated tonic.

In [2], a joint tonic and raag (melodic structure) recognition methodology was presented for North Indian classical music. Instead of generating a single template for each raag, the method generates multiple PCDs for each raag from the pitch tracks extracted from the annotated audio excerpts. Given the raag of a musical excerpt, the PCD computed from the excerpt is compared with each of the template PCDs of the same raag by circularly shifting the test PCD. The shift in the closest match indicates the tonic.

The disadvantage of using audio recordings for template computation is the necessity of adequate amount of training data. Moreover, the quality of the data has to be maintained so that the intervallic properties are represented well. Even so, a test distribution can substantially differ from the corresponding template. A common confusion is the estimation of another pitch (or pitch class) when its occurrence is comparable to the occurrence of the tonic. Moreover, in cases when an audio recording includes unrelated musical content in addition to a performed piece, e.g. improvisations or performances of other pieces in different modal structures, the audio distribution would be a mixture of the distributions of these distinct musical events. This might cause substantial confusions. This problem motivates the replacement of audio recordings with a more "definitive" information source in the template training step. If available, scores can be good sources, since they provide an easily accessible symbolic description of many relevant musical components.

When score information is available, utilising the sequential note information might bring a more effective solution to tonic identification. In [4], we introduce a method to link the musically relevant structural elements (sections) given in the score and the corresponding time-intervals in a audio recording. The method estimates all possible links in an audio recording by applying Hough transform to similarity matrices computed between the prominent pitch extracted from the audio recording and the synthetic pitch extracted from the score fragments. The estimated links are searched according to the section sequence given in the score to obtain the most-likely section links.

In this paper, we present two methodologies to identify the tonic of a performed piece by comparing the melodic contents extracted from the performance and the score of the piece. We use *makam* music knowledge and the findings from previous research [2, 4, 7] to specialise both the methodologies for the melodic aspects of *makam* music of Turkey. Both methods extract prominent pitch from the audio recording. Then a fine-grained, kernel-density pitch class distribution (KPCD) is computed from the audio prominent pitch, and tonic candidates are selected. Adapted from [2, 7], Method I applies circular shifting to the audio KPCD according to tonic candidates. Each shift is then compared with a score KPCD computed from the monophonic melody in the score. Method II normalises the prominent pitch with respect to each tonic candidate. Next, it attempts to link melodic fragments in the score with the respective time intervals in the audio by using the candidate link estimation approach explained in [4]. As the first experiments in "score-informed" tonic identification, we consider inter-linked audio-score collections, where the audio recording and the score are already known to be related with the same work (composition). We use musical scores, which include the *makam* of the piece, the boundaries of the structural elements and the sequence of these elements.

The remainder of the paper is as follows: Section 2 makes a brief introduction to *makam* music of Turkey. Section 3 explains the proposed methodologies. Section 4 presents the data collection used in the experiments. Section 5 explains the experiments done to test the methodologies and provides the results. Section 6 wraps up the paper with a discussion and conclusion.

2. MAKAM MUSIC OF TURKEY

The melodic structure of most traditional music repertoires of Turkey follows the concept of *makam*s. *Makam*s are modal structures, where the melodies typically revolve around a *başlangıç* (starting, initial) tone and a *karar* (ending, final) tone [6]. *Karar* is synonymous to tonic. There are a number of different transpositions (*ahenk*), any of which might be favored over others due to instrument/vocal range or aesthetic concerns [6]. The default ahenk is called "*bolahenk*." For an extended discussion of *ahenk*s, the readers are referred to [6, Appendix F].

Currently, Arel-Ezgi-Uzdilek (AEU) theory is the mainstream theory for the *makam* music of Turkey (MMT) [6]. AEU theory argues that there are 24 equal intervals and a whole tone is divided into 9 equidistant intervals. These intervals can be approximated from 53-TET (tone equal tempered) intervals, each of which is termed as a *Holdrian comma* (1 Hc ≈ 22.6415 cents) [6]. On the other hand, the intonation of some intervals in the performance might differ from the theoretical intervals as much as a semitone [6, 17]. [1]

Since early 20^{th} century, a score representation extending the Western music notation has been used in MMT [12]. This notation typically follows the rules of AEU theory. The scores tend to notate monophonic melodic lines. However performances may differ from the score substantially due to the heterophonic characteristics of makam music and artistic decisions such as non-notated embellishment, note/ phrase insertion, repetitions and omissions. The register information given in the notation is always relative to the instrument, i.e. the lowest tone of the same pitch class that an instrument can produce is always indicated with the same symbolic note in the score. In *bolahenk*, the notes are written a perfect fourth higher than it sounds, i.e. the *rast* note represented by the $G4$ in the staff is played in the pitch class of approximately $D4 = 293.66$ Hz [6].

In the experiments, we focus on *peşrev*, *saz semaisi* (the two most common instrumental forms) and *şarkı* (the most common vocal form) forms from the classical reperoire. Both peşrev and saz semaisi typically consist of a repetitive section called *teslim*. In the *şarkı* form, there is typically a repetitive section called *nakarat*.

3. PROPOSED APPROACHES

We define tonic identification as "estimating any frequency belonging to the same pitch class of the *karar* note." Even though the octave information is important in *makam*s, we

[1] Throughout the text we will represent a note name as [note letter][octave]{accidental}$^{\{\text{Hc distance}\}}$, e.g. the note *mahur* is written as G5$\flat^4$. Note that the performed frequency might be different.

chose to generalize the term tonic to the pitch class of the *karar* note due to some performance scenarios, where it is ambiguous to define the octave of the performance tonic. [2]

Given the problem definition, we propose two methodologies to identify the tonic of a performance of MMT using score information. In this paper, we focus on *interlinked* audio-score collections, where the scores and audio recordings are already related with the respective compositions via available metadata. Both methodologies take advantage of a machine readable score, which stores the value and the duration (i.e. the $\langle$*note-name*, *duration*$\rangle$ tuple) of each note. The tuple sequence form a symbolic monophonic melody. Additionally, the score is divided into sections, some of which are repeated. The *makam* and the tempo of the piece are provided in the score. Therefore, we do not need any structural analysis to find the repetitive structural element. The audio recording might include various expressive decisions such as musical materials that are not related to the piece, phrase repetitions/omissions and pitch deviations.

From music-theory [6], we compile a dictionary consisting on $\langle$*makam*, *karar*$\rangle$ pairs, which stores the *karar* of each *makam* (e.g. if the *makam* of the piece is Hicaz, the *karar* is A4.). *Karar* note is used as the reference symbol during the generation of synthetic pitch from the score (Section 3.1.1). We also refer to theoretical intervals defined in AEU theory to generate the score features from the machine-readable score (Section 3.1.1).

Method I generates a synthetic pitch from the monophonic melody given in the score and extracts a prominent pitch from the audio recording (Section 3.1.1). Kernel density estimation (KDE) is applied to these melodic features and the 160-D kernel-density pitch class distributions (KPCDs) are obtained (Section 3.1.2). The audio KPCD is circularly shifted according to its peaks and compared to the score KPCD. The candidate used to shift the audio KPCD, which results in the minimum distance to score KPCD, is selected as the tonic (Section 3.2).

Method II (Section 3.3) uses the note sequence information given in the score. Similar to the first method, it computes a prominent pitch and audio KPCD from the audio recording. From the score, the second method only extracts the synthetic pitch from the monophonic melody of the repetitive structural element (e.g. teslim, nakarat) indicated in the score. The method then normalizes the audio prominent pitch with respect to each peak in the audio KPCD. Then, it attempts to link the repetitive structural element in the score with its respective locations in the audio recording. The candidate of the section linking result, which outputs the most confident links, is selected as the tonic.

Before presenting the methodologies, we first explain the feature extraction and tonic candidate selection steps, which are shared by both the approaches.

[2] As an example, consider an ensemble performance, where each instrument performs the same melodic contour in its own register.

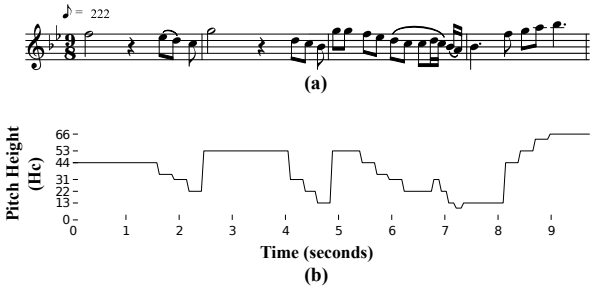


Figure 1. The first nakarat section of the composition, *Gel Güzelim*. a) Score, b) Synthetic pitch computed from the note symbols and durations.

3.1 Feature Extraction

Notation and audio recording are different representations of music. To compare these information sources, we need to extract features which adequately capture the musical content given in each representation. In [4], we found that prominent pitch is a highly effective and intuitive feature to analyze MMT due to the monophonic nature of the scores and the heterophonic practice. From the prominent pitch, we further compute a fine-grained pitch class distribution. FPCDs are shown to model the intervallic properties of the *makam*s adequately [7]. Moreover, FPCDs are able to capture the intonation information in a limited fashion, i.e. the width and shape of the peaks.

3.1.1 Synthetic Pitch and Prominent Pitch Computation

For the computation of synthetic pitch from the score and prominent pitch from the audio, we use the feature extraction step explained in [4]. Here we give a brief summary of the process.

To compute the synthetic pitch p_s for the desired score fragment (i.e. the whole score in Method I and the repetitive section in Method II), we extract the corresponding $\langle$*note-name*, *duration*$\rangle$ tuple sequence associated with the fragment. Then, we pick the *makam* of the composition, which is given in the score, and obtain the *karar*-name of the piece by checking the *makam* in the $\langle$*makam*, *karar*$\rangle$ dictionary. The note names are mapped to the Hc distances according to AEU theory with reference to the *karar* note. Finally, the synthetic pitch for the score fragment is generated at a frame rate of $\sim$46 ms, which provides sufficient time resolution to track all changes in pitch. Figure 1 shows the repetitive section given in the score of the composition, *Gel Güzelim*, [3] and the synthetic pitch computed for this fragment.

To obtain the audio prominent pitch p_a, we use the Essentia implementation [1] of the melody extraction algorithm proposed by [13]. The approach computes the main melody after separating salient melody candidates from non-salient ones. We include all the non-salient candidates to guess the prominent pitch since non-melodic intervals are very rare in MMT. Melody extraction is done using a pitch precision of 7.5 cents ($\approx \frac{1}{3}$ Hc), which is reported

[3] *http://tinyurl.com/lfp8x83*

as a suitable pitch precision for MMT [7]. The hop size is chosen equal to the frame rate of the p_s (∼46 ms).

3.1.2 Pitch Distribution Computation and Tonic Candidate Selection

A fine-grained pitch class distribution (FPCD) is computed from the audio prominent pitch to find tonic candidates in both methods. Method I also computes a score FPCD from the synthetic pitch of the whole score and shift-compares it with the audio FPCD. For the audio FPCD computation, the unit of the prominent pitch is converted from Hz to Hc with respect to the middle C (261.63 Hz). This reference is selected as a dummy value for the unit conversion.

While shift-comparing FPCDs to every possible pitch value is effective for template matching [7], in previous research [2] we showed that picking the location of the peaks in the PCD as tonic candidates greatly reduces the computation time with minimal losses in tonic accuracy. It was also observed that, kernel-density pitch class distributions (KPCDs), which are computed using kernel density estimation (KDE), perform significantly better than histogram-based pitch class distributions, when candidates are selected as the peak locations [2]. Hence, we use kernel density estimation (KDE) to compute the FPCDs. In KDE, an observation contributes to neighbouring bins according to a kernel. When the kernel is chosen as Gaussian, the continuous kernel density $\hat{f}(x)$ is given by:

$$\hat{f}(x) = \frac{1}{nh} \sum_{j=1}^{n} \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-p(j))^2}{2h^2}} \tag{1}$$

where h is the kernel width, $p(j)$ is the value of the j^{th} index of the prominent pitch p, and n is the total number of the prominent pitch values.

We use the function *ksdensity* in MATLAB [4] to obtain a discrete approximation of the kernel density. The discrete approximation of kernel density provides smoothness over histogram computation. KDE is especially helpful in score FPCD computation (Section 3.2), since the pitch spread provides robustness to the microtonal deviations in the tuning and intonation. We empirically set the kernel width h to 15 cents ($\approx \frac{2}{3}$ Hc) so that an observation practically contributes within an interval of 4Hc, slightly smaller than a semitone. Finally, the estimated kernel density is octave wrapped, and the KPCD is obtained. We can write a KPCD with N bins as $D = \langle d_1, \ldots, d_i, \ldots, d_N \rangle$, where d_i denotes the value of the pitch class index i. We retain the pitch precision of the prominent pitch (7.5 cents), resulting in $N = 160$ bins. In the audio KPCD (D_a), the first bin is initialised to the dummy value (261.63 Hz) used in Hc conversion. In the score KPCD (D_s), the first bin indicates the pitch class of the *karar* note, i.e. $53k$ Hc, where $k \in \mathbb{Z}$.

We select the index of all the peaks in D_a as the tonic candidates.

[4] *http://www.mathworks.com/help/stats/ksdensity.html*

3.2 Method I: Distribution Matching

In the first method, we compute a score and an audio KPCD from both score and audio recording, respectively. For the score KPCD computation, the whole note sequence is used. The locations of the peaks in the audio KPCD (D_a) are picked as the candidate tonics. Using the template matching approach [2, 7] we apply circular-shift the audio KPCD such that each candidate is carried to the first bin. Circular-shift can be simply formulated as:

$$D^i = \langle d_i, d_{i+1} ..., d_N, d_1, ..., d_{i-1} \rangle \tag{2}$$

Next, the shifted audio KPCD D_a^i is compared to D_s. We use Bhattacharyya distance, which was shown to outperform common L_n distances (e.g. City Block, Euclidean) in PD comparison [2]. Bhattacharyya distance Δ between the score KPCD and shifted audio KPCD can be written as:

$$\Delta(D_s, D_a^i) = -\ln \left(\sum_{k=1}^{N} \sqrt{D_s(k) D_a^i(k)} \right) \tag{3}$$

where k denotes an index of the D_s and D_a^i, respectively.

Bhattacharyya distance is computed between the score and each shifted audio KPCD. The index i which results in the minimum distance, indicates the estimated pitch class. The estimated tonic c_i can be represented as:

$$c_i = 261.63\, k * 2^{\frac{7.5(i-1)}{1200}},\ k \in \mathbb{Z} \tag{4}$$

3.3 Method II: Repetitive Section Linking

Using PCDs, we can only take an advantage of the interval and some limited intonation information. Nevertheless, scores also include note sequence information. In Method II, we attempt to link a melodic fragment from the score with the audio recording by using the candidate link estimation presented in [4].

We first compute the audio prominent pitch p_a. We also compute audio KPCD and obtain the candidate tonics. Using Equation 4 and setting $k = 1$, the candidate indices are converted back to Hz, i.e. $c_i(k = 1)$, where i is the index of the tonic candidate. Next we convert the prominent pitch values from Hz to Hc with respect to each *karar* candidate such that the *karar* candidate has a value of 0 Hc. The *normalised audio prominent pitch* can be expressed as:

$$p_a^i = 53 * \log_2 \left(\frac{p_a}{c_i(k=1)} \right) \tag{5}$$

Next, synthetic pitch of the repetitive section indicated in the score p_s is extracted. We compute a similarity matrix S^i between p_s and p_a^i:

$$S_{jk}^i = \begin{cases} 1, & \big(|p_s(j) - p_a^i(k) + \alpha| \bmod 53\big) - \alpha < \beta \\ 0, & \big(|p_s(j) - p_a^i(k) + \alpha| \bmod 53\big) - \alpha \geq \beta \end{cases} \tag{6}$$

where mod indicates the modulo operation. Each element in the similarity matrix indicates whether two pitch values can be deemed as the same pitch class within the binarization threshold β. α is a dummy value greater than β to

ensure pitch differences between $[53k - \beta,\ 53k], k \in \mathbb{Z}$ are treated as similar. In [4], we found that 3 Hc is an optimal value for β. In the similarity matrix S^i, diagonal line segments are observed which indicate the possible performed locations of the repetitive section. We apply Hough transform to detect the diagonal line segments [5]. We observe that the tempo of a performance typically varies between between 0.55 and 1.5 times the tempo indicated in the score [4], which we use to restrict the searched angles between $-28.81°$ and $-56.31°$. We obtain a set of links $\{l_1^i, \ldots, l_m^i\}$ for each tonic candidate, where m refers the number of links found using the tonic candidate. The number of non-zero pixels forming the line segment is normalised by the length of the line segment, giving the weight $w(l_k^i)$, $1 \leq k \leq m$, of the segment. We combine the weight of each link and obtain a accumulated weight for each tonic candidate. The accumulated weight is given as:

$$w^i = \sqrt[3]{\sum_{k=1}^{m} w(l_k^i)^3} \tag{7}$$

Equation 7 ensures that (possibly erroneous) links with low weights are greatly suppressed with respect to the links with high weights. The tonic is estimated as the pitch class c_i, which has the highest accumulated weight w^i.

4. DATA COLLECTION

For our experiments, we collected 116 audio recordings of 24 preşrevs, 84 audio recordings of 19 saz semaisis, and 57 audio recordings of 14 şarkıs (257 audio recordings of 57 compositions in total). The compositions are taken from the classical repertoire, in which the makam and the karar note are clearly defined in music theory. The *makam* of each composition is included in the metadata. [5] The pieces cover 28 different *makam*s.

The scores are obtained from the symbTr database. The symbTr-score is machine-readable text format, which stores the value and the duration of the note sequences [10]. The symbolic representation also contains information about the structure of the composition, i.e. the section sequences and the indices of the initial and final note of each section are indicated.

The audio recordings are selected from the CompMusic collection, and they are either in public-domain or commercially available. Some recordings include musical events which do not belong to the composition such as improvisations and even performances of other compositions.

The ground truth is obtained by manually marking the tonic frequency using *Makam Toolbox* [7]. Figure 2a and Figure 2b shows the distribution of the annotated tonic with respect to the pitch class C and the distribution of the transpositions with respect to *bolahenk*, respectively. It can be seen that the annotated tonic are mostly distributed around the semitones with microtonal deviances. Apart from *bolahenk*, the tonic is mostly performed with a transposition

[5] The metadata is stored in MusicBrainz: *http://tinyurl.com/mfvop6l*

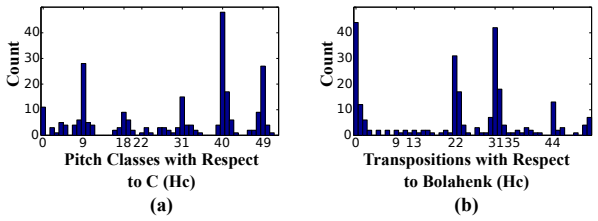


Figure 2. Distribution of the annotated tonics in the data collection. a) Pitch class histogram of the annotated tonic with respect to the pitch class C, b) Histogram of the transpositions with respect to *bolahenk*

around the perfect fourth, perfect fifth and minor seventh. Nevertheless a considerable number of tonic annotations reside in microtonal pitch classes.

5. EXPERIMENTS AND RESULTS

We use the methodologies explained in Section 3 to identify the tonic. We compare the estimated tonic from each algorithm with the manually annotated tonic. If the distance between the estimated and the annotated tonic are less than 1 Hc, the estimation is marked as correct. [6]

Tonic identification by repetitive section linking fails only in one piece (99.2% success rate). In this recording, [7] the vocalist sings a *gazel* (vocal improvisation) in almost three fifth of the duration of the recording with skillful vibratos extending up to ≈ 200 cents peak-to-peak. These vibratos occasionally cross *mahur* (G5♭[4]) and less frequently reach to *gerdaniye* (G5), which is in the pitch class of the tonic. Throughout the piece the pitch class G♭[4] is visited more than G and it shows a wide spread towards G such that no peak is formed in the vicinity of the tonic pitch class. In this case the pitch class G♭[4] is estimated as the tonic, having a 2.33 Hc deviation. [8]

Using distribution matching, we are able to identify the tonic of 244 performances out of 257 (94.9% success rate). Most of the errors occur in *makam*s Kürdilihicazkar (3 recordings), Muhayyer (3 recordings), Suzidilara (2 recordings), Isfahan and Mahur (1 recordings each), which have complex pitch distributions. The errors are distributed mostly to the fourth (7 recordings) and fifth (4 recordings) of the scale degree. In 4 recordings the tonic is identified as the başlangıç (initial) note, which is the other melodic center of the *makam*. The average distance between the annotated tonic and the correctly estimated tonics is 0.23 Hc with a standard deviation of 0.21 Hc for both methods.

For comparison, we also modify and test the approach in [7] using the Makam Toolbox implementation. We use the audio prominent pitch as the input to improve the f0-estimation. The *makam* of the piece is provided to the algorithm. We use a subset of the collection with 152 au-

[6] The results are available in *http://compmusic.upf.edu/node/164*.

[7] *http://tinyurl.com/n42g5dh*

[8] Interestingly 2 out of 3 section links produced by the erroneous tonic are correct, since the Hc distance between the annotated and estimated tonic (2.33 Hc) is less than the optimal binarization threshold $\beta = 3$ Hc. In the next step, we can align the audio and score in the note-level and correct any errors and micro-deviances in the tonic.

dio recordings. The number of failed identifications is 46, 10 and 1 for audio-based template matching, PCD matching and repetitive section linking, respectively. The results from both of our methods are substantially better than the results obtained from the Makam Toolbox.

6. DISCUSSION AND CONCLUSION

We proposed two novel methods that use score information to identify the tonic of audio recording. Assuming the most played pitch classes as tonic candidates, the first method compares pitch class distributions computed from the audio and score, and the second method searches for a repetitive score fragments in the audio. We tested the methodologies in a scenario of audio-score collections of MMT, where the audio and score are already linked with each other at the document level and the score includes the notes, as well as the structural organization, the *makam* and the tempo of the piece. The results indicate that score information greatly simplifies the tonic identification task. Moreover, the pitch deviances between the estimated tonic and the annotated tonic are mostly indiscernible. These findings point out the computational potential of knowledge-driven methodologies using multi-modal information.

While template distributions computed from audio are similar to the testing distributions with respect to the tuning and limited intonation information in *makam* level, score distributions indicate these similarities in the (more definitive) composition level. On the other hand, the distribution matching method is still susceptible to the errors seen us audio-based template matching. In the majority of the recordings where distribution matching failed, it was observed that the piece has modulations to pitches that do not belong to the scale of the *makam*. These contrastive notes and any event can be grouped into characteristic fragments, melodic progressions and structural elements; eventually building the unique the music piece. In general, the lack of such temporal information is the main problem of distribution matching.

By linking repetitive sections, we only missed the tonic of one performance. These results indicate the usefulness of the temporal information in pitch related tasks. The successful results obtained from tonic identification and previously from section linking [4] motivates adapting the "fragment linking" methodology for further computational tasks. First is to generalize the method to less "complete" scores, where structure information is unknown. Our initial tests show that tonic estimation by linking non-repetitive fragments and score fragments as short as 7 seconds is possible. In the next step, we want to work on linking audio and scores in the document level by trying to link sections in each score with corresponding audio recordings. Highly ranked links will indicate the scores and audio recordings related to the same work.

Another interesting direction is to generate predictive models from the scores of each *makam*. The models can be used to discover characteristic phrases, which could be linked with the audio to further carry relevant tasks such as *makam* recognition, melodic similarity analysis and expression analysis. Previously we found that multiple viewpoints may be highly predictive in modelling MMT [3]. We plan to take advantage of these computational methodologies and models to discover, navigate through and appreciate cultural-specific aspects of *makam* music of Turkey and other music genres/traditions involving melody-dominant content.

7. ACKNOWLEDGEMENTS

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).

8. 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 *Proceedings of ISMIR*, 2013.

[2] P. Chordia and S. Şentürk. Joint recognition of raag and tonic in North Indian music. *Computer Music Journal*, 37(3), 2013.

[3] S. Şentürk. Computational modeling of improvisation in Turkish folk music using variable-length Markov models. Master's thesis, Georgia Institute of Technology, 2011.

[4] S. Şentürk, A. Holzapfel, and X. Serra. Linking scores and audio recordings in makam music of Turkey. *Journal of New Music Research*, (submitted).

[5] R. O. Duda and P. E. Hart. Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[6] E. B. Ederer. *The Theory and Praxis of Makam in Classical Turkish Music 1910-2010*. PhD thesis, University of California, Santa Barbara, September 2011.

[7] Ali Cenk Gedik and Barış Bozkurt. Pitch-frequency histogram-based music information retrieval for Turkish music. *Signal Processing*, 90(4):1049–1063, 2010.

[8] Emilia Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, 2006.

[9] D. B. Huron. *Sweet anticipation: Music and the psychology of expectation*. MIT press, Cambridge, Massachusetts, 2006.

[10] K. Karaosmanoğlu. A Turkish makam music symbolic database for music information retrieval: SymbTr. In *Proceedings of ISMIR*, pages 223–228, 2012.

[11] C. L. Krumhansl and R. N. Shepard. Quantification of the hierarchy of tonal functions within a diatonic context. *Journal of experimental psychology: Human Perception and Performance*, 5(4):579–594, 1979.

[12] E. Popescu-Judetz. *Meanings in Turkish Musical Culture*. Pan Yayıncılık, Istanbul, 1996.

[13] 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.

[14] J. Salamon, S. Gulati, and X. Serra. A multipitch approach to tonic identification in Indian classical music. In *Proceedings of ISMIR*, pages 499–504, 2012.

[15] X. Serra. A multicultural approach in music information research. In *Proceedings of ISMIR*, pages 151–156, 2011.

[16] D. Temperley and E. W. Marvin. Pitch-class distribution and the identification of key. *Music Perception: An Interdisciplinary Journal*, 25(3):193–212, 2008.

[17] Yalçın Tura. *Türk Musıkisinin Meseleleri*. Pan Yayıncılık, Istanbul, 1988.

Oral Session 3:

Text Processing

PLACING MUSIC ARTISTS AND SONGS IN TIME USING EDITORIAL METADATA AND WEB MINING TECHNIQUES

Dimitrios Bountouridis, Remco C. Veltkamp, Jan Van Balen
Utrecht University, Department of Information and Computing Sciences
{d.bountouridis,r.c.veltkamp,j.m.h.vanbalen}@uu.nl

ABSTRACT

This paper investigates the novel task of situating music artists and songs in time, thereby adding contextual information that typically correlates with an artist's similarities, collaborations and influences. The proposed method makes use of editorial metadata in conjunction with web mining techniques, aiming to infer an artist's productivity over time and estimate the original year of release of a song. Experimental evaluation over a set of Dutch and American music confirms the practicality and reliability of the proposed methods. As a consequence, large-scale correlational analyses between artist productivity and other musical characteristics (e.g. versatility, eminence) become possible.

1. INTRODUCTION

Many real-world music collections show a lack of metadata when it comes to placing their constituent music entities in a semantic or quantitative context. As a result, content management and disclosure become challenging tasks. Meanwhile, in the emerging field of the digital humanities, well-documented collections are becoming increasingly essential for high quality research.

The lack of contextual information has been typically addressed by content-based approaches, where knowledge is extracted after the actual audio is processed and analysed. Contrarily, Web-Music Information Retrieval (MIR) techniques exploit the "wisdom of the crowd" and use the Web or music metadata hubs in order to estimate the desired information. Typical applications include artist similarity [3, 9], classification [10], country of origin determination [8, 11] and many more exceeding our scope.

This paper focuses on quantifying artists' productivity over time, and estimating the original release time of songs. The productive period of a music artist is important information that is typically highly correlated to his style, influences and similarities to other artists. Teitelbaum et. al. [12] have shown that the activity span is strongly associated with artist collaborations. As such, the productive years constitute a reliable, additional feature for various MIR tasks: as the authors of [13] argue, listeners typically show a certain affection for music related to particular periods of their lives, and therefore time information could act as a basis for music recommendation.

The practical applications of productivity profiles exceed the MIR domain. According to [5], productivity in absolute terms may be the most important factor for a comprehensive understanding of the creativity in music. Based on that, Kozbelt [1] investigated the correlation between productivity and musical characteristics such as versatility and eminence. From a musicological perspective, a quantitative representation of productivity can offer valuable insights in music trends, significant musical events, significant social events, and the mutual influence that may exist between them.

1.1 Problem Definition and Related Work

We define a song's year of release as the year on which it was first released in a recording. We further define as an artist's productivity profile (APP) the distribution of years in which the artist was alive and musically active, meaning recording and releasing albums, singles, etc. The productivity of an artist for a given year corresponds to the number of recorded songs released throughout that year.

Time information regarding an artist's output is typically provided by services and hubs such as MusicBrainz [1], Last.fm [2], Allmusic.com [3] etc. in the form of editorial metadata, i.e. data related to prescriptive knowledge about the music [6], in addition to domain-free sources such as Wikipedia. However, when dealing with old and lesser-known artists, any provided information is highly probable to be erroneous, incomplete or even non-existent. For example, the MusicBrainz profile for the popular Dutch singer Willy Derby (1886-1944) includes a series of compilation albums, released after his death, and only 16 singles out of his huge catalogue. To the best of our knowledge, the only published work aiming at automatically providing time information for artists and songs, although inside a recommendation framework, is [4]. Bogdanov and Herrera address the issue of determining a record's original epoch, meaning the years when the music was first recorded, produced and consumed. This task is handled by

[1] musicbrainz.org
[2] ww.last.fm
[3] www.allmusic.com

finding the release on Discogs [4] with the earliest date and by propagating it using a decreasing weighting scheme. Similarly to MusicBrainz though, Willy Derby's Discogs profile is limited (2 albums, 3 singles and one compilation) and therefore this approach is destined to fail.

The only content-based method we know [13] acts as a proof of the Million Song Dataset's applicability, and aims at estimating a song's year of release based on its audio features. Such an approach shows a small mean error of 6 years but actually estimates the year that the song would best fit in and not its actual year of release. Considering that audio data for old and lesser-known artists, such as Willy Derby, is typically hard to find, this method can be rendered useless in this context. The same holds for audio fingerprinting methods such as [7] and commercial services such as Shazam [5], which in addition face the metadata scarcity problem.

Based on the previous, the need for a reliable method that overcomes the scarcity of the editorial metadata of lesser-known artists and is based on high-level metadata only (e.g. artist name - song title), becomes apparent.

1.2 Contribution

The contribution of this paper is multiple. First, it introduces the novel task of accurately placing music entities, and especially artists, in a time context. In addition, we provide a publicly available testset. Secondly, it employs a methodology that combines both editorial metadata and web mining techniques, a conjunction rarely investigated. The fusion of the different sources is aided by musically meaningful heuristics offering room for research. Thirdly, our method incorporates generic techniques for birth-date and death-date estimation that can find applications outside the music context.

The remainder of this paper is organized as follows. Sections 3 and 4 describe our proposed method for the artist productivity profile and year of release estimation respectively. Sections 5 and 6 summarize the evaluation and experiment results, while section 7 presents our conclusions.

2. GENERAL FRAMEWORK

Our generic method for determining a person's productivity profile (singer, actor, author etc.) is performed in various steps. Web search engines are initially mined to provide the "preliminary" productivity profile. Secondly, editorial metadata hubs are queried for time data about the person's life and works. This generates what we call a "shaping" profile. The information gathered from this process is merged and applied on the first, to attenuate any noise and shape the final productivity distribution (see Figure 1). In the possible case of absent lifespan editorial metadata, our method estimates the birth and death dates based on the preliminary profile.

[4] www.discogs.com
[5] www.shazam.com

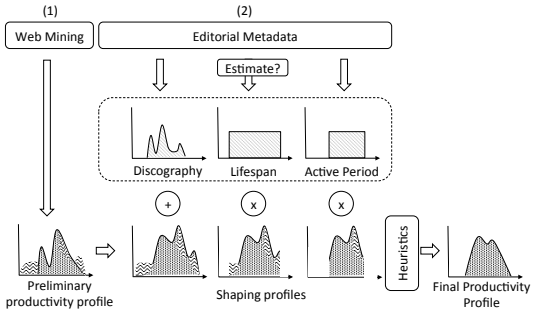


Figure 1. A graphical representation of the artist productivity profile estimation process. The initial Web-mined profile is subsequently noise-filtered, based on editorial (or estimated) medatada and heuristics.

3. DETERMINING ARTIST PRODUCTIVITY PROFILE

3.1 Editorial Metadata Retrieval

Given A an artist name and S^A a set of song titles corresponding to the artist's recordings, we first try to match any of the tuples $\langle A, s_j \rangle$ where $s_j \in S^A$ to the databases of Last.FM, EchoNest and MusicBrainz. Besides being well established and previously used by MIR researches, the employed metadata hubs provide convenient APIs. The following pieces of information are desirable for each artist: *a)* discography, *b)* lifespan and *c)* active years period.

Given that we have retrieved two values corresponding to the start and end years (s, e) of an artist's activity, we create a 130-bin profile $P_{\text{act}} \in [0,1]^{1\times 130}$, spanning the years 1880 to 2010, with both $P_{\text{act}}(s)$ and $P_{\text{act}}(e)$ set to 1. Similarly, we create a profile P_{life} for the lifespan data.

A P_{disc} profile is also created for the discography data, but populating it is more sophisticated. The discography data comprises of album names, song titles accompanied by their release date and "release group" information. Our method assumes that some release groups (e.g. singles) are more reliable than others (e.g. compilations) with regard to the original date determination. Given $D = \{(year_1, weight_1),..., (year_n, weight_n)\}$ the retrieved data from MusicBrainz, with $year_i \in \{1880, .., 2010\}$ and $weight_i \in [0, 1]$, the P_{disc} gets the following values:

$$P_{\text{disc}}(y) = \sum_{\forall i: year_i = y} \frac{weight_i}{N_{\text{release group}_i}} \qquad (1)$$

$N_{\text{release group}}$ is a normalization factor corresponding to the number of recordings belonging to that particular release group.

3.2 Web Mining

The second step is concerned with identifying the Web pages related to the artist under consideration. Our method queries Google and Bing with the scheme "A+music" as in [8], and retrieves the 100 and 80 top-ranked URLs, denoted as sets G and B respectively. Fetching and indexing

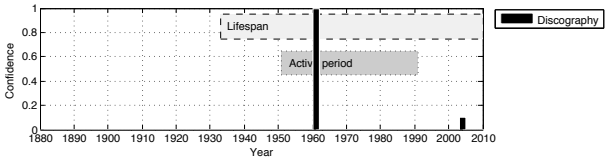


Figure 2. A representation of the three profiles, derived from editorial metadata, for the artist Corry Brokken. Discography data correspond to just one single (1960) and one compilation (2003).

the webpages in $B \cup G$ are performed by the Apache Nutch web crawler [6] and Apache Lucene [7] respectively.

3.2.1 Profile Construction

At this stage we aim at generating a probability distribution that would best model the artist's productivity as it is documented on the Web. This is performed using the following:

- **Returned counts**: The number of returned Lucene pages pc_q for query q, acts a draft estimate of the query's relevance inside the pool of retrieved documents.
- **Co-occurrence analysis:** Considering returned counts as probabilities, the conditional probability of a term t_1 to co-occur in the same page as t_2 can be written as $p(t_1|t_2) = pc_{t_1,t_2}/pc_{t_1}$.
- **Relevance score:** Apache Lucene employs a sophisticated variation of the *tf*idf* and Boolean mode to calculate the similarity between a query q and a document d, denoted $sim(q,d)$. The details of the so-called Lucene Practical Scoring Function [8], are omitted here.

For each year $y_j \in \{1880, 1881, \dots, 2010\}$ we query Lucene with "$A + y_j$". Incorporating a proximity factor to the query, ensures that when A and y_j are separated by over 300 words, the document containing them would be considered irrelevant. The value of 300 was chosen based on a set of preliminary experiments.

Our method assigns a score to each query "$A+y_j$" using the following formula:

$$s(A, y_j) = \frac{score(A, y_j)}{score(A)} \quad (2)$$

where:

$$score(q) = max_{1<k<pc_q}[sim(q, d_k)] \times pc_q \quad (3)$$

A profile P_{web} of 130 bins is populated such that $P_{\text{web}}(y_j) = s(A, y_j)\ \forall y_j \in Y$. Often, however, time information is not explicitly stated. For instance, it is quite common for an artist that was active during the period 1980-1990, to be considered and identified as an "80's" artist. Based on this, we have identified a set of terms T that semantically correspond to decades. For example, the terms "1960's", "sixties", "60's", "jaren 60" and "jaren zestig" correspond to the period 1960-1970. The two latter country-specific terms are introduced manually but can be automated based on a country of origin estimation process [8,11]. Therefore finally, we query Lucene with "$A + t_j$", where $t_j \in T$, and then increase the value of the ten corresponding bins by $s(A, t_j) \times 0.1$.

3.3 Profile Fusion

By the end of the previous process, the system has acquired four separate profiles (P_{act}, P_{disc} , P_{life} and P_{web}). We combine those pieces of information in a two-step procedure.

It is very likely for the discography to be incomplete for apparent reasons. Our method tries to compensate for that fact by smoothing P_{disc} with a sized-5 Gaussian window. A profile P_{f} is later created such that its bin values hold the weighted sum of the normalized P_{web} and P_{disc} .

The system exploits lifespan and active years data by setting all the coefficients of P_{f} that fall outside the P_{life}, P_{act} boundaries, to zero. Despite this process, the P_{f} data inside the lifespan or active period may still contain a significant amount of noise. Further noise removal is based on the observation that an artist's productivity is usually maximized during his first 20 years of adulthood. This is supported by the dotted line in Figure 3, which represents the distribution of single-type releases across the artist's lifespan, as generated from a Musicbrainz subset of 518, pre-1950's artists. This behaviour is modelled by our method as an envelope-probability density function W (solid line Figure 3). Our envelope's decay slope is less steep in order to accommodate for non-single releases. W is aligned with the artist's birth-date, as provided by P_{life}, and used to weigh P_{f} which constitutes the final artist productivity profile estimate.

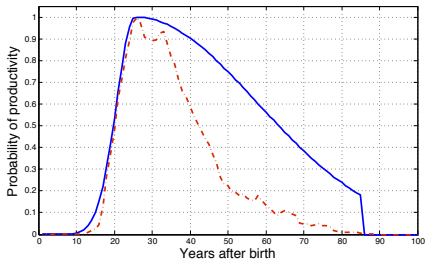


Figure 3. (Solid line) the probability density function W for artist productivity. (Dotted line) the distribution of single-type releases across lifespan.

3.3.1 Birth-Date & Death-Date Estimation

As previously mentioned, it occurs very often that no lifespan information can be found. In this case we use P_{web} which ideally amplifies important years in the life of an

[6] nutch.apache.org
[7] lucene.apache.org/core
[8] lucene.apache.org/core/old_versioned_docs/versions/3_5_0/api/all/org/apache/lucene/search/Similarity.html

artist, represented as sharp peaks. For the birth-date estimate, the idea is to locate those and pick the one that best fits our productivity probability assumptions, as modelled by W. This is achieved by traversing and aligning W with each found peak and then multiplying it with P_{web}. The peak that yields the largest area under the profile is considered the birth-date estimate.

The selection of the peak candidates is a critical procedure, considering that the noise-to-signal ratio can be significantly high. Our method aims at emphasising those peaks that exhibit high surprisingness or unexpectedness, in contrast to noise-generated peaks. This is achieved by scanning the P_{web} profile from left to right; the coefficients of the surprisingness vector $S_v \in [0,1]^{1\times 130}$ are then computed such that:

$$S_v(j) = \frac{P_{\text{web}}(j)}{\sum_{k=1}^{j-1} P_{\text{web}}(k)} \quad (4)$$

Peak candidates are then selected using a simple thresholding function (Figure 4).

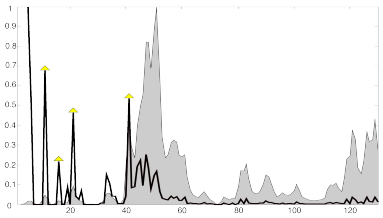


Figure 4. P_{web} and its surprisingness vector S_v (solid line). The more data is processed (from left to right) the less surprising peaks become.

Death-date estimation is based on the assumption that the artist's productivity, as documented on the Web, will be limited or non-existent after his death. Noise however, challenges this assumption; therefore our method uses a peak-picking technique that employs the following pieces of information and heuristics: 1) Birth-date, estimated or known, 2) max life expectancy, set to 90 years and 3) a P'_{web} profile generated with a proximity factor of 10 (instead of 300) which aims at capturing co-occurrences instances of the type "Artist (Year of birth - Year of death)".

The peak picking is done as follows: by using a simple thresholding function on P'_{web}, we firstly pick the peak candidates p. Each one of them is assigned a confidence-probability value $prob(p)$ based on its distance from the birth-date; assuming that peaks far away from the birth-date have higher chance of corresponding to death-date.

Ideally, the peak that maximizes the ratio between distributions to its left and to its right should correspond to the correct death-date. However, considering once again the noise and its non-uniform distribution (e.g. album re-releases after the artist's death), a more sophisticated technique is required. Our method, aiming at capturing the short and long term distributions around each peak, creates four sub-profiles from P_{web}, centred on p, with window sizes of 40 and 10 (P^{40}_{left}, P^{40}_{right}, P^{10}_{left}, P^{10}_{right}). Each candidate p is then assigned a value such that:

$$v_p = \left(\frac{mean(P^{40}_{\text{left}})}{mean(P^{40}_{\text{right}})} + \frac{mean(P^{10}_{\text{left}})}{mean(P^{10}_{\text{right}})} \right) \times prob(p) \quad (5)$$

The candidate with the maximum v_p is considered the death-date estimate.

We evaluated our lifespan estimation method on a test-set of 100 MusicBrainz artists with known lifespans, ranging from 1880 to 2000. The mean error for the birth-date and death-date was estimated at approximately 11.5 and 10.6 years respectively. This might not be exactly accurate, yet the death-date estimation is not a goal in itself, rather a pragmatic strategy to remove some of the noise in the profiles.

4. DETERMINING YEAR OF RELEASE

Estimating year of release is based on a similar approach as the one employed for artist productivity profiles. The basic idea is to create four separate profiles for each $\langle A, s_j \rangle$: a release information profile from MusicBrainz, a productivity profile for A and two Web profiles for s_j and $A + s_j$ respectively. Processing and fusing them is the final step.

We first query the metadatabases with tuples of the form $\langle A, s_j \rangle$ in order to retrieve the discography. It is now easy to search into the discography of A for any of the songs in S^A. If there is a match, the release group is "Single" and the year of release y is available, then a profile P_{disc} for the song in hand is populated, such that $P_{\text{disc}}(y) = 1$.

During web-mining, two set of queries for each tuple $\langle A, s_j \rangle$ are applied, and more specifically "$s_j + year_k$" and "$s_j + A + year_k$" . Eventually two profiles, $P_{\text{web,s}}$ and $P_{\text{web,s+}A}$, are calculated.

Calculating the year of release estimate is based on the assumption that the input vectors correspond to mixture components. Their conical sum P_{CS} is:

$$P_{\text{CS}} = \mathbf{W_v}^T \times [P_{\text{web,s}}, P_{\text{web,s+A}}, P_{\text{f},A}, P_{\text{disc}}] \quad (6)$$

with $\mathbf{W_v} = [w_1, w_2, w_3, w_4]$ the weight vector and $P_{f,A}$ the artist's productivity profile. Finding the optimal weights is solved with the employment of a genetic algorithm on a training set. The final year of release estimate is just the P_{CS} coefficient with the maximum corresponding value.

5. EXPERIMENT

5.1 Test Collection

There exists no standardized or previously used data set for this kind of task, therefore we built one from scratch [9]. Note that the requirements are very specific: on the one hand, an evaluation dataset of artist productivity profiles should only include complete or near-complete discographies, and on the other hand, should focus enough on the more challenging artists for which no complete catalogues are readily available on the web. The low commercial distribution and rarity of pre-60's music was ideal for our

[9] Available at www.projects.science.uu.nl/COGITCH

purposes; therefore, we manually gathered from our personal music collection, books [2] and "deep" Web sources, 639 Dutch and American song titles, corresponding to 15 artists (see Table 1), accompanied by original release dates, ranging between the period of 1900 to 1959. Overall, 26832 documents were downloaded and indexed for the year of release and 3391 for artist productivity profile including a set of "noise" webpages (irrelevant to the artists themselves).

The difficulty in assessing the "obscurity" of an artist and the amount of effort required to gather and cross-check his complete discography without using the Web, resulted in a rather small artist productivity profile test set. It should be noted that a direct comparison of our method to others is unfortunately impossible. The work of [4] exploits Discogs which offers limited, or non-existent, data for our particular testset. Regarding year of release estimation, the work of [13] uses purely audio features which are almost impossible to acquire, considering our method's prerequisite for music rarity and oldness.

5.2 Evaluation Measures

For the artist productivity profiles the idea is to examine the overlap between the ground truth APP and estimated APP^* distributions. This is achieved by using precision, recall and their harmonic mean, usually called F-measure. In addition, given the mean values of both profiles' Gaussian fits, denoted m and m^* respectively, we compute $Er = |m - m^*|$ as a measure of the error in terms of time context placement.

Given N, the number of songs in S, YoR_{s_j} the true release date of the song s_j and $YoR^*_{s_j}$ the estimate, "Accuracy" for a year-window of size x is defined by the following formula:

$$Accuracy_x = \frac{\sum_{s_j \in S} f_x(YoR^*_{s_j}, YoR_{s_j})}{N} \quad (7)$$

where

$$f_x(t,e) = \begin{cases} 1 & \text{for} \quad |t-e| \leq x \\ 0 & \text{for} \quad |t-e| > x \end{cases} \quad (8)$$

6. RESULTS AND EVALUATION

The results for the artist productivity profile task are presented in Table 1. Our approach shows low error with regard to the time context placement. It is worth examining certain illustrative cases, starting with "August De Laat" (Figure 5), which presents one of the lowest precisions. The artist is well placed into the time context but our approach assumes a considerable amount of productivity from 1940 to 1954. This misbehaviour relies on the fact that even after De Laat's last recording in 1941, he remained active in non-music areas such as theatre [10].

In contrast to the previous case, "Bob Scholte" shows both accurate time-context placement and distribution modelling (Figure 6). In the case of "Louis Davids" (1883-1939) presented in Figure 7, lifespan or active years information from the metadabases was unavailable. Our method estimated the correct birth and death dates by performing the peak picking algorithm presented in 3.3.1. Filtering the profile by applying the productivity assumptions, as modelled by W, also attenuated a considerable amount of noise right after the artist's birth.

[10] www.thuisinbrabant.nl/personen/l/laat,-august-de

Artist Name	*Precision%*	*Recall%*	*F%*	*Er*
August De Laat	48.36	83.97	61.38	2.45
Bob Scholte	65.1	90.82	75.84	1.15
Kees Pruis	73.86	80.21	76.9	1.45
Lou Bandy	51.9	97.97	67.85	3.95
Louis Davids	50.2	99.85	66.81	0.11
Willy Derby	90.21	81.93	85.87	1.25
B. Schoepen	52.75	85.11	65.13	6.02
Cole Porter	66.33	88.2	75.72	1.94
Corry Brokken	81.89	88.65	85.14	1.21
Eddy Christiani	85.06	78.22	81.49	0.56
George Gershwin	61.31	83.64	70.76	2.66
Harold Arlen	81.6	68.36	74.39	5.69
Jerome Kern	78.11	63.12	69.82	3.95
Richard Rodgers	46.74	97.13	63.11	7.55
Wim Sonneveld	55.74	95.85	70.49	1.01
Mean	**65.5**	**86.135**	**72.665**	**2.56**

Table 1. Precision, recall and F-measure for the 15 artists in the test set.

Table 2 presents the $Accuracy_x$ for the year-of-release estimation task for windows ranging from 1 to 5. The mean error is 2.91 years. As a general evaluation measure we consider $Accuracy_2$, assuming that this level of detail is appropriate for artists of the era 1900 - 1959. Therefore, for a 2-year window around 81% of the cases are identified as hits; significantly outperforming the random, baseline estimation (mean error = 33.4, $Accuracy_2 = 8.92\%$).

Window Size	1	2	3	4	5
Accuracy	0.66	0.816	0.856	0.888	0.907

Table 2. Accuracy for window size ranging from 1 to 5.

7. CONCLUSIONS

The aim of this report has been to determine an artist's productivity profile and a song's original year of release. Our approach is based on the exploitation of editorial metadata from sources such as MusicBrainz and EchoNest, in addition to Web harvested data. The evaluation demonstrates the strength of the proposed method for year of release estimation; around 81% of the estimates fall within a $\pm$2-year window, with a mean error of 2.91 years.

The results for determining productivity profiles are less impressive though it should be noted that the ground truth generation assumes complete knowledge of the artist's discography, which is not always the case. In fact, in the cases for which we are most certain we have the complete discographies, modelling the productivity distribution is accurate.

With our novel methods, it is possible for the first time to perform large-scale, correlational analysis between productivity and various musical characteristics. Therefore, work such as [1, 5], which is based on manually gathered classical music, cannot only be significantly aided but also expanded to include artists from various eras and of varying popularity.

Given certain enhancements and modifications, our approach can be generalized to accommodate non-music domains. Wikipedia instead of MusicBrainz or EchoNest can be employed for determining an author's productivity profile or the original dates of his publications. Metadata hubs such as IMDB [11] can be used to extend our approach in the movies domain as well.

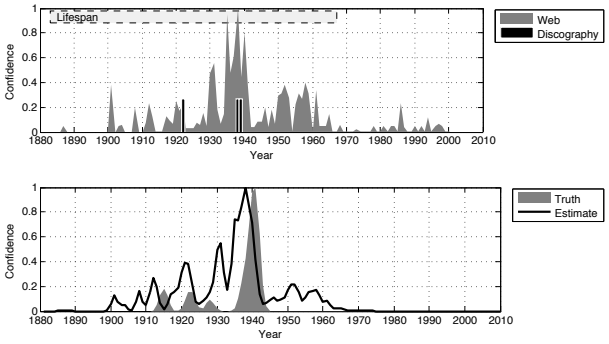


Figure 5. (Top) the input profiles, (bottom) the ground truth against the estimated artist productivity profile (APP) for August de Laat.

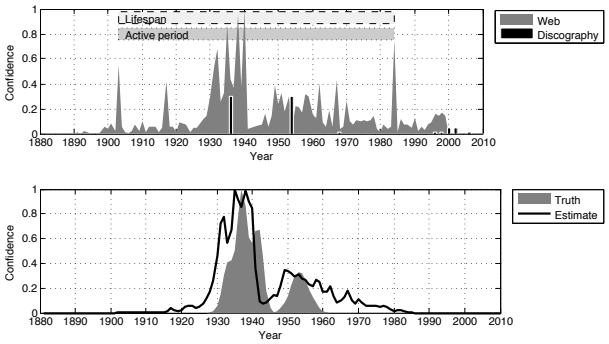


Figure 6. (Top) the input profiles, (bottom) the ground truth against the estimated APP for Bob Scholte.

8. ACKNOWLEDGMENTS

This research is supported by the NWO CATCH project COGITCH (640.005.004), and the FES project COMMIT/. The authors would like to thank Frans Wiering for his useful comments.

9. REFERENCES

[1] A. Kozbelt: "Performance time productivity and versatility estimates for 102 classical composers," *Psychology of Music*, Vol. 37, No. 1, pp. 25-46, 2009.

[2] A. Wilder: *American Popular Song: The Great Innovators, 1900 - 1950*, Oxford University Press, New York, 1990.

[11] www.imdb.com

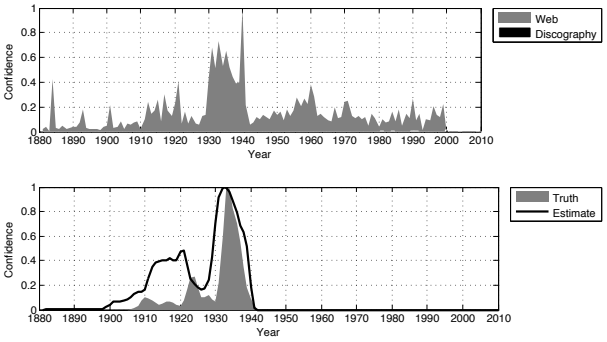


Figure 7. (Top) the input profiles, (bottom) the ground truth against the estimated APP for Louis Davids.

[3] B. Whitman and S. Lawrence: "Inferring descriptions and similarity for music from community metadata," *ICMC Conference Proceedings*, pp. 591-598, 2002.

[4] D. Bogdanov and P. Herrera: "Taking advantage of editorial metadata to recommend music," *CMMR Conference Proceedings*, pp. 618-632, 2012.

[5] D. K. Simonton: "Creative productivity: a predictive and explanatory model of career landmarks and trajectories," *Psycological Review*, Vol. 104, No. 1, pp. 66-89, 1997.

[6] F. Pachet, A. La Burthe, J. Aucouturier and A. Beurive: "Editorial metadata in electronic music distribution systems: between universalism and isolationism," *Journal of New Music Research*, Vol. 34, No. 2, pp. 173-184, 2005.

[7] J. Haitsma and T. Kalker: "A highly robust audio fingeprinting system," *ISMIR Conference Proceedings*, pp. 107-115, 2002.

[8] M. Schedl, C. Schiketanz and K. Seyerlehner: "Country of origin determination via web mining techiniques," *AdMIRe Proceedings*, pp. 1451-1456, 2010.

[9] M. Schedl and D. Hauger: "Mining microblogs to infer music artist similarity and cultural listening patterns," *WWW Conference Proceedings*, pp. 877-886, 2012.

[10] P. Knees, E. Pampalk and G. Widmer: "Artist classification with web-based data," *ISMIR Conference Proceedings*, pp. 517-524, 2004.

[11] S. Govaerts and E. Duval: "A web-based approach to determine the origin of an artist," *ISMIR Conference Proceedings*, pp. 261-266, 2009.

[12] T. Teitelbaum, P. Balenzuela, P. Cano, and J. M. Buldu: "Community structures and role detection in music networks," *Chaos journal*, Vol. 18, No. 4, pp. 043105, 2005.

[13] T. Bertin-Mahieux, D.P.W. Ellis, B. Whitman and P. Lamere: "The million song dataset," *ISMIR Conference Proceedings*, pp. 591-596, 2011.

THE MILLION MUSICAL TWEETS DATASET: WHAT CAN WE LEARN FROM MICROBLOGS

David Hauger
Johannes Kepler University Linz
Austria
david.hauger@jku.at

Markus Schedl
Johannes Kepler University Linz
Austria
markus.schedl@jku.at

Andrej Košir
University of Ljubljana
Slovenia
andrej.kosir@ldos.fe.uni-lj.si

Marko Tkalčič
Johannes Kepler University Linz
Austria
marko.tkalcic@jku.at

ABSTRACT

Microblogs and Social Media applications are continuously growing in spread and importance. Users of Twitter, the currently most popular platform for microblogging, create more than a billion posts (called tweets) every week. Among all the different types of information being shared, some people post their music listening behavior, which is why Twitter became interesting for the Music Information Retrieval (MIR) community. Depending on the device and personal settings, some users provide geographic coordinates for their microposts.

Having continuously crawled and analyzed tweets for more than 500 days (17 months) we can now present the "Million Musical Tweet Dataset" (MMTD) – the biggest publicly available source of microblog-based music listening histories that includes geographic, temporal, and other contextual information. These extended information makes the MMTD outstanding from other datasets providing music listening histories.

We introduce the dataset, give basic statistics about its composition, and show how this dataset allows to detect new contextual music listening patterns by performing a comprehensive statistical investigation with respect to correlation between music taste and day of the week, hour of day, and country.

1. INTRODUCTION

Microblogs and social media have continuously been growing in importance over the past years — for end users, but also for industry and academia. Compared to other sources of information, they show high actuality and benefit from a large number of users. Twitter [1], for instance, the currently largest platform for microblogging, already has about 500 million users as of October 2012, according to Twitter CEO Dick Costolo [2] (the last official numbers date back to 200 million users in April 2011 [3]).

Microblogs have already been proven successful in a number of different contexts (see Section 2), but up to now they have hardly been exploited within the field of Music Information Retrieval (MIR). As there are no strict rules or specified formats for the up to 140 characters a Twitter post (tweet) consists of, Twitter is a relatively noisy source of information. However, thanks to hashtags and plugins for music players that automatically post music listening events, Twitter is a valuable source of information for MIR, when trying to incorporate or explicitly evaluate the user context. Compared to other sources like Last.fm [4], Twitter also provides information on geographic positions if available (for instance, if posted from a GPS-enabled device).

In this paper, we present a novel dataset of information derived from microblogs (tweets), describing the music listening habits of users. It is composed of preprocessed tweets, being consistently mapped to artists and songs from MusicBrainz [5]. As far as we are aware of, this dataset is currently the largest source of information on geospatial music listening events publicly available.

The remainder of the paper is structured as follows. First we give an overview of existing datasets and their applications in Section 2. In Section 3 we provide information about the acquisition of the dataset and some basic statistics. In Section 4 we conduct a statistical analysis of the correlation between music taste (measured via genre distribution) and temporal as well as geographical properties. In Section 5 we briefly present some ideas on how the dataset can be exploited, focusing on music visualization and contextual clustering. In Section 6 we summarize the work and outline possibilities of further exploiting the dataset.

2. RELATED WORK

Microblogging services like Twitter are continuously growing in importance. They have already been exploited

[1] http://www.twitter.com

[2] http://www.telegraph.co.uk/technology/twitter/9945505/Twitter-in-numbers.html

[3] http://huffingtonpost.com/2011/04/28/twitter-number-of-users_n_855177.html

[4] http://www.last.fm

[5] http://musicbrainz.org

for research in different areas, for instance, detection of breaking news [16], trends [12] [23], earthquakes [17], or health issues [14], even exploring spatio-temporal dynamics [10].

Nevertheless, within the MIR community, microblogs are still a relatively new source of information. As recent work stresses the importance of adding contextual information to music recommendation [1] [22], we believe that microblogs in general, and the proposed dataset in particular, provide valuable information for MIR. `Twitter` has already been used as a source for music similarity estimation [20], music recommendation [24], and for identifying cultural listening patterns [19].

Unfortunately, most existing datasets of `Twitter` posts, such as "The Edinburgh Twitter Corpus" [15], are not suited for geospatial MIR tasks as only very few tweets are related to music and less than 3% of the tweets have geolocalized information available. Also the frequently used dataset of the TREC 2011 and 2012 Microblog tracks [6] [13] is not suited. Although it contains approximately 16 million tweets, this dataset is not tailored to music-related activities, i.e. the amount of music-related posts is marginal.

Moreover, tweets have no specific format ($\leq$ 140 characters of unstructured text) and therefore need preprocessing to be assigned to a specific artist and/or song. Nevertheless, `Twitter` is one of the few sources with geolocalized listening information being available. Therefore, this source could be valuable for previous work like [2], where `Last.fm` was used as a source for listening histories. The most recent similar, but much smaller dataset for music-related microblogs is the "MusicMicro" dataset [18]. Besides being smaller, the "MusicMicro" dataset does not offer genre information nor other musical metadata (except for artist and song name).

As for datasets targeted at the music domain, the "Million Song Dataset" [3] and the "Yahoo! Music Dataset" [5] are quite popular. However, they do either not provide listening information ("Yahoo! Music Dataset") or only taste profiles without geo infomation ("Million Song Dataset"). The "Million Musical Tweet Dataset" (MMTD) presented here, in contrast, provides geolocalized listening information and links plain text tweets to respective artists and tracks, which allows for combination with content-based [4] and other contextual features [7]. The MMTD currently provides the biggest publicly available dataset for geolocalized music listening behavior. It can be downloaded from `http://www.cp.jku.at/datasets/MMTD/`.

The two pieces of information that make the MMTD unique are *temporal information* and *geographic information*. The former was shown to be extremely useful in the recommendation systems domain. Koren et al. [9] developed a matrix factorization-based method for modeling temporal dynamics in order to improve the rating prediction in the movies domain. Similarly, Koenigstein et al. [8], used this method to improve the rating prediction on the "Yahoo! Music Dataset" in the 2011 KDD Cup. Information on countries are valuable for culture-specific MIR approaches [21].

[6] `http://trec.nist.gov/data/tweets`

rank	country code	number of users	rank	country code	number of tweets
1	US	70,204	1	US	227,432
2	ID	30,605	2	DE	153,163
3	BR	24,985	3	BR	145,049
4	MY	14,771	4	GB	130,951
5	FR	13,890	5	ID	94,245
6	GB	9,006	6	FR	65,525
7	RU	5,234	7	MY	50,648
8	NL	5,223	8	CA	27,370
9	MX	4,538	9	RU	23,542
10	TR	2,878	10	MX	18,717
11	ES	2,847	11	NL	18,320
12	SG	2,422	12	TR	14,479
13	PH	2,385	13	ES	11,811
14	CA	2,344	14	SG	7,637
15	IT	1,521	15	IT	6,412
16	JP	1,501	16	AR	6,319
17	ZA	1,297	17	PH	5,723
18	DE	1,133	18	JP	5,529
19	UA	1,062	19	UA	4,940
20	AR	874	20	ZA	3,733

Table 1. Top-20 countries by number of users and tweets.

3. DATA ACQUISITION AND BASIC STATISTICS

In this section, we provide the background of the data acquisition and processing that led to the presented dataset. We further give some basic dataset statistics.

3.1 Data Acquisition

Between September 2011 and April 2013 we crawled the `Twitter` Streaming API [7], which provides a random subset of 1% of all tweets. We retained only tweets with geographic information attached (less than 3% of all tweets) and including potentially music-related hashtags that have already been proven successful [6], e.g. `#nowplaying`, `#np`, `#itunes`, `#musicmonday` and `#thisismyjam`. We employed the pattern-based approach described in [6] to map the content of the tweets to artists and tracks. We used the `MusicBrainz` database for indexing, which covers $\approx$ 30% of all tweets including the desired hashtags. Of course this method creates some bias in terms of cultural music listening patterns as the dataset is restricted to `Twitter` users posting musical information and there might be other conventions for the usage of hashtags in non-western countries.

To enable experimenting with the MMTD on a semantic level, we used a web service provided by `Mapquest` [8] to map geographic coordinates to cities, countries, and other geographic entities. To get comparable local time (which is not directly provided by `Twitter`) we used `GeoNames` [9] for retrieving the time zones for the geographic coordinates. The top 20 countries in terms of number of users, respectively number of tweets, are listed in Table 1.

In addition to this contextual information, we added genre information by querying `Last.fm` for tags on the

[7] `https://dev.twitter.com/docs/streaming-apis`
[8] `http://www.mapquest.com`
[9] `http://geonames.org`

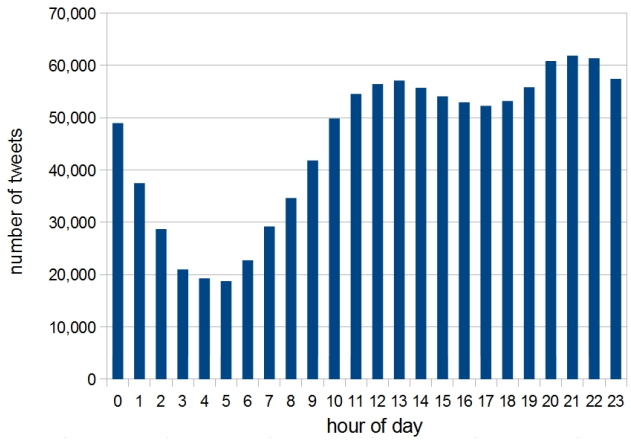


Figure 1. Number of tweets per hour of the day.

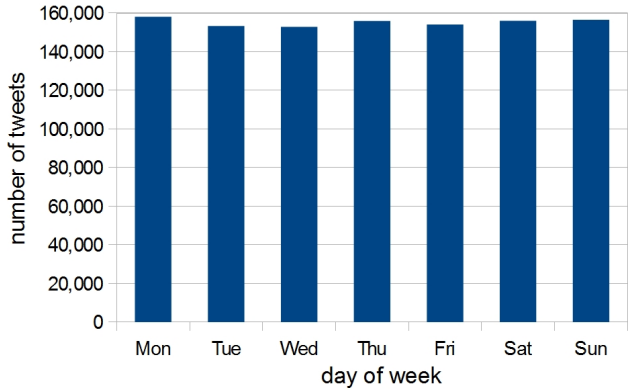


Figure 2. Number of tweets per day of the week.

artist and song level, filtering them by the 20 most popular genres from `Allmusic` [10] . This resulted in a multi-genre feature vector for each tweet.

3.2 Basic Statistics

The "Million Musical Tweets Dataset" (MMTD) is based on 1,086,808 tweets referring to 133,968 unique tracks ($mean = 8.11$ tweets/track; $\sigma = 44.94$; $median = 1$) by 25,060 different artists ($mean = 43.37$ tweets/artist; $\sigma = 327.03$; $median = 3$). The tweets were created by 215,375 users from 202 different countries. On average we have 1,078 users per country ($\sigma = 5{,}848$; $median = 29.5$) and 5,381 tweets per country ($\sigma = 25{,}032.45$; $median = 74.5$), each user creating on average 5.08 tweets ($\sigma = 268.19$; $median = 1$).

Analyzing the temporal distribution of tweets shows that twitterers are less active during night, as expected (see Figure 1). However, there is no significant difference between the days of the week (see Figure 2).

Using the pre-filtered `Last.fm` tags for multiple genre assignment, we obtained 276,697 tweets per genre ($\sigma = 256{,}662$). The distribution of genres is unbalanced as can be seen in Figure 3. For future work, different genre classifications could be investigated, for instance, using ontologies or synonyms for the provided tags.

[10] `http://www.allmusic.com`

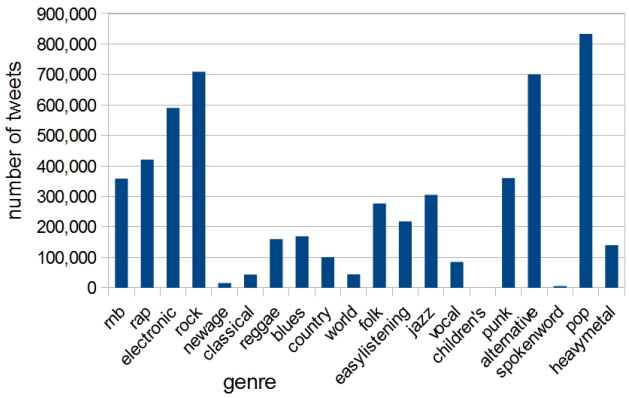


Figure 3. Number of tweets per genre.

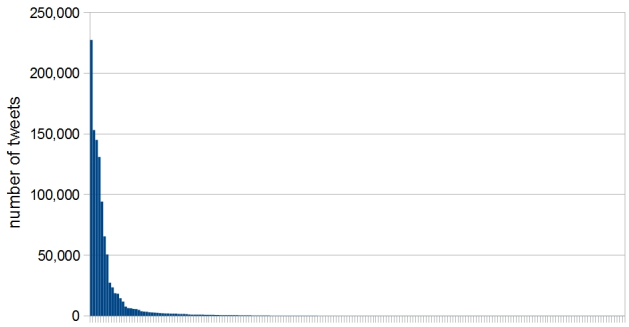


Figure 4. Number of tweets per country.

4. SPATIO-TEMPORAL STATISTICAL ASPECTS

In order to show the utility of the presented dataset we performed two experiments: (i) a geographical analysis of the listening preferences reflected in the dataset and (ii) a temporal analysis. For that purpose, we aggregated the dataset on a geographical and on a temporal basis. The tweets were first grouped by country. Then for each country the tweets were grouped by the day of the week. Finally, for each country and day of the week, the tweets were grouped by the hour of the day (e.g., all tweets from 2:00 to 2:59 were grouped together). When grouping, we summed the genre vectors of all the tweets belonging to a group, which yielded a summed vector, or histogram, of the genre distribution in each group. An excerpt of the aggregated data is shown in Table 2.

The distribution of tweets among countries, on the daily basis, and on the hour-of-the-day basis are shown, respectively, in Figures 4, 2, and 1.

4.1 Geographical analysis of the dataset

In this experiment we addressed the research question The research question addressed in this experiment is whether there are any differences between musical tastes among different countries that are reflected in the dataset. When we use the term *musical taste* we refer to how often specific genres have been played in the observed country (or other cluster in the next subsection).

To answer this question we first conducted the Kruskal-Wallis ANOVA (Analysis of Variance) (see [11] for de-

country	day of week	time of day	rnb	rap	el	ro	na	cl	re	bl	co	wo	fo	el	ja	vo	ch	pu	al	sw	pop	hm
Brazil	4 (Fri)	00	199	343	544	752	10	52	121	167	78	62	290	157	273	44	0	464	707	1	778	262
Brazil	4 (Fri)	05	29	30	49	70	2	8	13	19	5	6	22	17	28	6	0	39	65	0	69	17
Brazil	4 (Fri)	12	312	476	700	920	4	74	190	186	103	58	331	250	313	59	0	535	880	1	950	296
Brazil	4 (Fri)	17	260	424	647	873	6	60	150	164	99	44	299	201	315	45	0	533	847	3	880	310
Brazil	4 (Fri)	21	332	578	862	1104	6	60	194	220	149	62	385	247	382	63	0	684	1073	0	1164	426
France	6 (Sun)	00	213	277	194	208	4	10	90	50	17	30	69	56	90	21	0	80	196	6	260	24
France	6 (Sun)	05	22	29	22	18	1	0	8	4	0	1	4	3	9	2	0	6	17	0	24	0
France	6 (Sun)	12	422	529	440	417	4	12	173	105	27	51	112	116	203	40	0	173	403	3	528	41
France	6 (Sun)	17	280	325	276	280	1	13	108	75	26	32	72	89	113	28	0	118	302	5	366	20
France	6 (Sun)	21	265	331	283	283	1	20	113	72	30	22	91	83	129	34	0	106	268	2	348	22
Indonesia	0 (Mon)	00	128	145	253	352	7	19	60	88	72	18	173	155	154	41	1	187	322	2	395	76
Indonesia	0 (Mon)	05	27	34	42	59	2	5	13	12	13	1	26	23	27	6	0	32	57	0	67	15
Indonesia	0 (Mon)	12	206	223	359	509	3	29	58	85	81	17	185	211	175	47	0	226	466	0	580	104
Indonesia	0 (Mon)	17	245	295	428	619	16	35	84	108	84	17	206	255	201	55	0	296	569	1	706	110
Indonesia	0 (Mon)	21	273	316	511	722	25	68	81	141	137	31	302	360	287	85	0	360	680	2	843	144
Malaysia	2 (Wed)	00	133	169	245	306	8	23	40	58	47	15	140	98	122	32	0	163	295	2	358	59
Malaysia	2 (Wed)	05	9	9	18	21	1	1	2	3	1	2	8	10	6	4	0	13	20	0	29	5
Malaysia	2 (Wed)	12	75	84	137	173	6	8	26	29	27	5	62	53	63	14	0	92	158	0	209	37
Malaysia	2 (Wed)	17	167	178	269	327	9	12	53	45	53	20	133	100	111	38	0	157	302	2	392	72
Malaysia	2 (Wed)	21	173	180	271	355	4	18	52	51	68	22	133	150	123	39	0	157	332	2	432	69
United States	1 (Tue)	00	992	1062	865	923	12	47	392	218	145	59	313	267	466	122	0	502	994	13	1348	143
United States	1 (Tue)	05	248	266	219	251	2	9	100	53	40	14	80	79	125	29	0	132	259	4	346	42
United States	1 (Tue)	12	632	757	667	762	15	30	283	183	124	45	266	198	347	75	0	415	785	18	973	144
United States	1 (Tue)	17	696	833	678	712	7	35	270	145	107	35	239	210	341	69	0	359	792	14	991	104
United States	1 (Tue)	21	1125	1311	1076	1168	15	52	459	297	180	57	405	361	543	126	0	617	1241	17	1649	153

Table 2. Some random examples for aggregated data on a per-country, per-day-of-week, and per-hour-of-day basis from the Top-5 countries in terms of numbers of users. The order of genres corresponds to that in Figure 3.

tails) on the whole dataset grouping the data by genre and comparing each genre separately. The analysis showed that all p-values were $p < 0.001$, meaning that there are significant differences among all variables according to the geographical location of tweets.

After the ANOVA showed significant differences, we proceeded with a pair-wise comparison between the countries. Since there are 202 countries, this would mean roughly $20{,}000$ comparisons which makes it very hard, if not impossible, to interpret. As such an analysis would be hard to perform, we opted to choose a subset of 20 countries to perform the pairwise comparison. We chose the 20 countries with the biggest number of tweets (see Table 1).

In the pairwise comparison, we compared the histograms of all the 20 genres, among 20 selected countries using the Chi-square goodness of fit test (see [11] for details). The test result shows a p-value $p < 0.001$ for all pairs. Hence, all countries are significantly different from each other (compare to previous experiments on cultural listening patterns in [19]).

4.2 Temporal analysis of the dataset

We compared the distribution of genre preferences among different days of the week using the Kruskal-Wallis ANOVA to see whether listening habits are different on weekdays and on weekends. However, the test did reveal no significant differences among days of the week. Even when we narrowed the selection of countries down to five culturally distinct countries (Saudi Arabia, Malaysia, Germany, United Kingdom and Brazil), the differences were not significant. This means that the listening habits of the users present in this dataset do not differ between days. Two possible explanations for this finding are: (i) although someone might listen to the own favorites on week-ends and to artists reflecting the "common taste" among colleagues on working days, only music that is really liked is also posted via `Twitter` (if it is not automatically tweeted), or (ii) the aggregation already reflects this "common taste" for any day of the week.

Based on the distribution of the listening habits (tweets grouped by the hour of the day), as depicted in Figure 1, we clustered the tweets in the following hour-of-day groups, using the minimum at $5am$, and the local maxima at $noon$ and $10pm$ as borders:

- from 05:00 to 11:59 (morning)
- from 12:00 to 21:59 (afternoon/evening)
- from 22:00 to 04:59 (night)

Performing the cross-group test for all countries with the Kruskal-Wallis ANOVA showed that all p-values are $p < 0.05$, which means that there are significant differences among all the hour-of-day groups, e.g. people listen to different music on Monday mornings vs. Saturday nights. Repeating the test for the 5 culturally different countries, however, not all the p-values are $p < 0.05$, which we report in Table 3. This means that, for these countries, having $p > 0.05$, some genres do not show significantly different playing frequency among the various hour-of-day groups.

5. USING THE MILLION MUSICAL TWEETS DATASET FOR VISUALIZATION

Having discussed statistics of our dataset, the current section briefly points out an example of how the information within the MMTD may be used for clustering and visualizion. As multi-genre-vectors are not suited to map tweets to a certain color, we decided to use the approach presented by Hauger and Schedl [6] using non-negative matrix factorization of `Last.fm` genre tags and latent factors to assign tweets to a color.

We aggregated the tweets of our dataset by their geographic coordinates and displayed them as circles on a map, where the size of the circle represents the number of tweets for this area and the color represents the most

	rnb	rap	el	ro	na	cl	re	bl	co	wo	fo	el	ja	vo	ch	pu	al	sw	pop	hm
Chi-square	15.86	25.74	23.61	23.59	2.09	5.41	11.61	6.82	8.05	4.58	13.22	15.74	10.50	2.53	1.99	21.61	24.44	1.92	23.99	16.00
df	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Asymp. Sig.	0	0	0	0	0.352	0.067	0.003	0.033	0.018	0.101	0.001	0	0.005	0.282	0.371	0	0	0.384	0	0

Table 3. Kruskal-Wallis ANOVA results for the differences among the hour-of-day variable grouped by genre for the 5 culturally different countries.

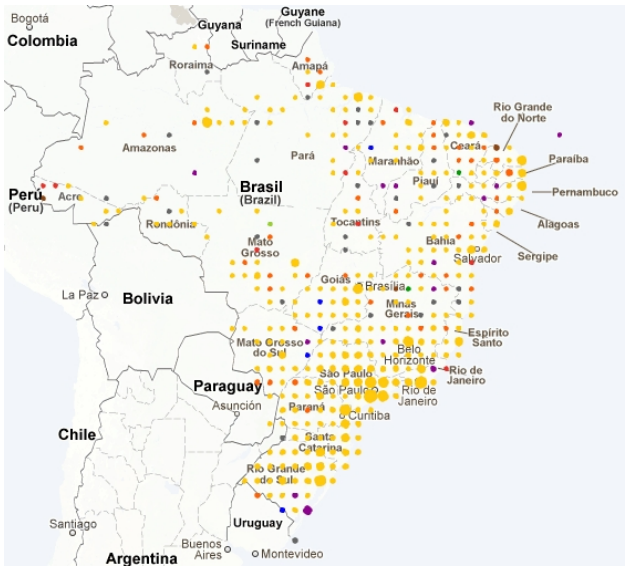


Figure 5. Visualizing genres / latent factors for Brazil.

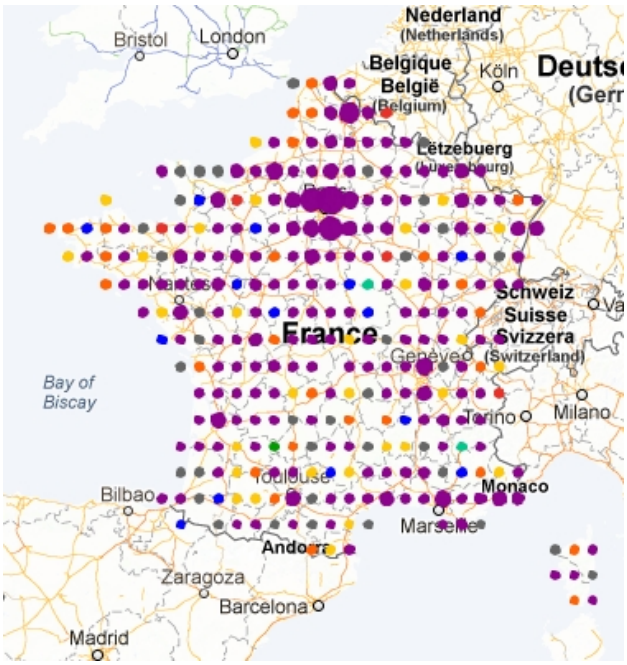


Figure 6. Visualizing genres / latent factors for France.

popular genre or latent factor within this set of tweets. According to our findings that there are significant differences in the genre distribution for countries, we hence visually show differences between countries.

Figure 5 shows that in Brazil the genre cluster for "Rock" (yellow) is by far the most popular one. Comparing it to France, the European country with the largest number of `Twitter` users, we see that French twitterers show a strong preference for the group representing the "Rap" and "Hip-Hop" cluster (violet).

6. DISCUSSION AND FUTURE WORK

In this paper, we presented the "Million Musical Tweets Dataset" (MMTD). Its unique property of including time and geo-location data allows the research community to follow novel research avenues.

The results of the statistical tests showed that there are significant differences in musical tastes (expressed through multi-genre vectors) among different clusters (both on the geographical basis and on the temporal basis). These differences could be exploited for developing adaptive systems/services on the geo-temporal basis (e.g., contextual filtering of music based on country and/or time of day). We also presented one way of visualizing differences in geospatial music listening patterns.

The proposed dataset is publicly available and may be used, for instance, for contextual music recommendation or similarity estimation. As for the latter, the MMTD could be used to build hybrid similarity functions including audio features, contextual music features (tags, playlist co-occurrences), and user context information.

Future work will put emphasis on user-related examination of the dataset, looking into other user-centric properties like age, gender, or twitting activity, as well as on the search for "cultural clusters" of different countries and/or cities. In this vein, it would also be interesting to examine whether there are differences between urban and rural areas.

7. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Funds (FWF): P22856, P25655, and by the European Union FP7 programme through the PHENICX project (grant agreement no. 601166).

8. REFERENCES

[1] Linas Baltrunas, Bernd Ludwig, Stefan Peer, and Francesco Ricci. Context-Aware Places of Interest Recommendations and Explanations. *Joint Proceedings of the Workshop on Decision Making and Recommendation Acceptance Issues in Recommender Systems (DEMRA) and the 2nd Workshop on User Models for Motivational Systems: The affective and the rational routes to persuasion (UMMS)*, 2011.

[2] D. Baur, F. Seiffert, M. Sedlmair, and S. Boring. The streams of our lives: Visualizing listening histories in

context. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1119–1128, 2010.

[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.

[4] Michael A. Casey, Remco Veltkamp, Masataka Goto, Marc Leman, Christophe Rhodes, and Malcolm Slaney. Content-Based Music Information Retrieval: Current Directions and Future Challenges. *Proceedings of the IEEE*, 96:668–696, April 2008.

[5] Gideon Dror and Noam Koenigstein and Yehuda Koren and Markus Weimer. The Yahoo! Music Dataset and KDD-Cup'11. In *Proceedings of KDDCup 2011*, 2011.

[6] David Hauger and Markus Schedl. Exploring Geospatial Music Listening Patterns in Microblog Data. In *Proceedings of the 10th International Workshop on Adaptive Multimedia Retrieval (AMR)*, Copenhagen, Denmark, October 2012.

[7] Peter Knees and Markus Schedl. A Survey of Music Similarity and Recommendation from Music Context Data. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 2013.

[8] Noam Koenigstein, Gideon Dror, and Yehuda Koren. Yahoo! music recommendations. *Proceedings of the 5th ACM Conference on Recommender Systems (RecSys)*, page 165, 2011.

[9] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89, April 2010.

[10] Chung-Hong Lee, Hsin-Chang Yang, Tzan-Feng Chien, and Wei-Shiang Wen. A Novel Approach for Event Detection by Mining Spatio-temporal Information on Microblogs. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 254–259, July 2011.

[11] E L Lehman, Joseph P. Romano, and Erich L. Lehmann. *Testing Statistical Hypotheses*. Springer Texts in Statistics. Springer New York, New York, NY, 2005.

[12] Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, pages 1155–1158, New York, NY, USA, 2010. ACM.

[13] Richard McCreadie, Ian Soboroff, Jimmy Lin, Craig Macdonald, Iadh Ounis, and Dean McCullough. On Building a Reusable Twitter Corpus. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Portland, OR, USA, August 12-–16 2012.

[14] Michael J. Paul and Mark Dredze. You Are What You Tweet : Analyzing Twitter for Public Health. *Artificial Intelligence*, pages 265–272, 2011.

[15] Saša Petrović, Miles Osborne, and Victor Lavrenko. The edinburgh twitter corpus. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*, pages 25–26, Los Angeles, California, USA, June 2010. Association for Computational Linguistics.

[16] S. Phuvipadawat and T. Murata. Breaking news detection and tracking in twitter. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 3, pages 120–123, 2010.

[17] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake Shakes Twitter Users: Real-Time Event Detection by Social Sensors. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*, May 2010.

[18] Markus Schedl. Leveraging Microblogs for Spatiotemporal Music Information Retrieval. *Lecture Notes In Computer Science*, 7814:796–799, 2013.

[19] Markus Schedl and David Hauger. Mining Microblogs to Infer Music Artist Similarity and Cultural Listening Patterns. In *Proceedings of the 21st International World Wide Web Conference (WWW): 4th International Workshop on Advances in Music Information Research: "The Web of Music" (AdMIRe)*, Lyon, France, 2012.

[20] Markus Schedl, David Hauger, and Julián Urbano. Harvesting microblogs for contextual music similarity estimation - a co-occurrence-based framework. *Multimedia Systems*, 2013.

[21] Xavier Serra. Data Gathering for a Culture Specific Approach in MIR. In *Proceedings of the 21st International World Wide Web Conference (WWW): 4th International Workshop on Advances in Music Information Research (AdMIRe)*, Lyon, France, April 17 2012.

[22] Bo Shao. *User-centric music information retrieval*. PhD thesis, Florida International University, Miami, FL, USA, 2011. AAI3472062.

[23] Beaux Sharifi, Mark-Anthony Hutton, and Jugal Kalita. Summarizing Microblogs Automatically. In *Proceedings of NAACL HLT*, June 2010.

[24] Eva Zangerle, Wolfgang Gassler, and Günther Specht. Exploiting twitter's collective knowledge for music recommendations. In *Proceedings of the 21st International World Wide Web Conference (WWW): Making Sense of Microposts (#MSM2012)*, pages 14–17, 2012.

VERIFYING TAG ANNOTATIONS THROUGH ASSOCIATION ANALYSIS

Tom Arjannikov
University of Lethbridge
tom.arjannikov@uleth.ca

Chris Sanden
University of Lethbridge
sanden@cs.uleth.ca

John Z. Zhang
University of Lethbridge
zhang@cs.uleth.ca

ABSTRACT

Music tags provide descriptive and rich information about a music piece, including its genre, artist, emotion, instrument, etc. While many work on automating it, at present, tag annotation is largely a manual process. It often involves judgements and opinions from people of different background and level of musical expertise. Therefore, the resulting tags are usually subjective, ambiguous, and error-prone. To deal with this situation, we seek automatic methods to verify and monitor this process. Furthermore, because multiple tags can annotate each music piece, our task lends itself to multi-label methods which capture the inherent associations among annotations in a given music repository. In this paper, we propose a novel approach to verify the quality of music tag annotations via association analysis. We demonstrate the effectiveness of our approach through a series of simulations using four publicly available music datasets. To our knowledge, our work is among the initial efforts in verifying music tag annotations.

1. INTRODUCTION

Due to the advances in technology, such as data storage and compression, media processing, information retrieval, and the Internet, digital music collections have been growing enormously in volume. Millions of songs previously in physical formats are now readily available through on-line accesses, presenting many new challenges in related fields.

Among them is *Music Information Retrieval* (*MIR*), an interdisciplinary area that attracts practitioners from computer science, cognitive science, information retrieval, musicology, psychology, etc. One of its current tasks is the design and implementation of algorithmic approaches for managing large collections of digital music; these include but are not limited to music classification, automatic tag annotation, recommendation and playlist generation. [6]

Currently, it is a common strategy to organize and access digital music collections via *textual meta-data*, usually *tags*, such as *genre*, *style*, *mood*, *artist*, etc. The process of annotating a music piece with appropriate meta-data is referred to as *music tag annotation*. It relies heavily on music experts as well as amateurs [19]. Due to the ambiguity and subjectivity introduced in the annotation process, music tags can be inconsistent, incomplete and sometimes even error-prone, making it difficult to maintain them in large music collections [12]. Moreover, the manual annotation process can be complex, involving substantial financial and labor costs [5]. These are among the many issues, which automatic approaches aim to tackle.

1.1 Previous work

Automatic music tag annotation is an important problem in MIR with numerous applications. For instance, an accurate prediction of tags is often the first step toward making music recommendations. As of recently, music tag annotation has received considerable attention and many related techniques have been proposed.

Turnbull *et al.* [18] employ a generative probabilistic model and propose one of the first automatic tag annotation systems. In addition, they create a music dataset called *The Computer Audition Lab 500* (*CAL500*), which has since become a *de facto* benchmark for evaluating the performance of tag annotation systems.

Hoffman *et al.* [8] present another probabilistic model, called the *Codeword Bernoulli Average*, which attempts to predict the probability that a tag can be associated with a music piece. In addition, Bertin-Mahieux *et al.* [3] propose *Autotagger*, a model that uses ensemble learning schemes to associate tags with music pieces. Ness *et al.* [10] describe how *stacked generalization* of the probabilistic outputs of a *Support Vector Machine* (*SVM*) can be used to improve the performance of automatic tag annotation.

Shen *et al.* [16] propose a framework called *MMTagger* that combines advanced feature extraction techniques and high-level semantic concept modeling for music tag annotation. The proposed framework uses a multilayer architecture that gathers multiple *Gaussian mixture models* and SVMs.

Sanden and Zhang [13] treat music tag annotation as a multi-label classification problem and discuss various issues related to tag annotation through extensive experiments using a set of a multi-label classifiers and a set of evaluation measures.

Neubarth *et al.* [11], use association analysis to find different, musicologically motivated, associations between various folk music genres and their geographical distribution.

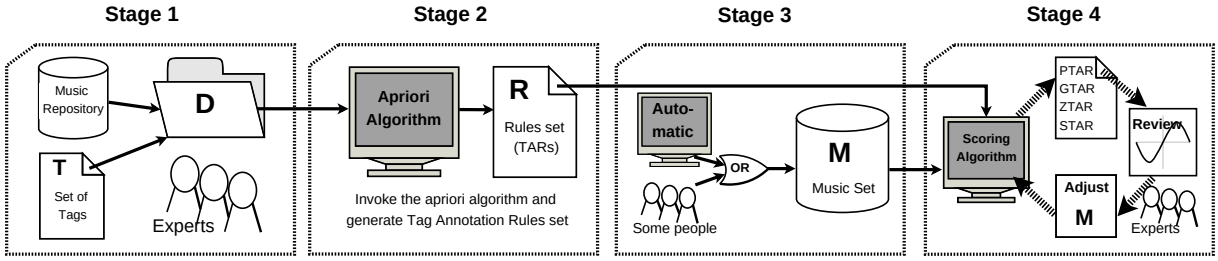


Figure 1: The four stages of our proposed approach to verification of tag annotations through association analysis.

1.2 Tag Annotation

We follow Ness *et al.* [10] in their formulation of the music tag annotation process as follows. Given a set of tags $T = \{t_1, t_2, ..., t_A\}$ and a set of music pieces $M = \{m_1, m_2, ..., m_R\}$, then each music piece $m_j \in M$ is an annotation vector $A = (a_1, a_2, \cdots, a_A)$, where $a_i > 0$ if tag t_i has been associated with the piece, and $a_i = 0$, otherwise. These a_i's, referred to as *semantic weights*, describe the strength of the semantic correspondence between a tag and its music piece. When mapped to a binary assignment of {0,1}, the semantic weights can be interpreted as class labels, i.e., whether a tag is assigned to the music piece or not. Naturally, each music piece can have multiple tags [10, 13].

Tags for a given music piece reveal the inherent musical nature that it attempts to convey and express. As a coherent expression, these tags represent features that distinguish this music piece from others. This expression intuitively shows a strong association of these tags to the music piece or to a set of similar music pieces in terms of their musical nature. We work toward this intuition and aim to capture associations between tags and utilize them to verify the annotation process.

1.3 Association Analysis

Association analysis attempts to discover the inherent relationships among data objects in an application domain. Such relationships are represented as association rules. An example of such application domain is the basket analysis in supermarkets, where one tries to discover the relationships among commodities in baskets. For example, the association rule $\{milk\}$,$\{eggs\} \rightarrow \{bread\}$ implies that, if *milk* and *eggs* are bought together by a customer, then *bread* is likely to be bought as well, i.e., they have some inherent statistical relationships.

Let $I = \{i_1, i_2, \cdots, i_n\}$ be a set of n items and $T = \{t_1, t_2, \cdots, t_m\}$ be a transactional database, where each transaction $t_i \in T$ is a nonempty subset of I. An association rule is of the form $A \rightarrow B$, where A and B are called *itemsets* and $A \subseteq I$ and $B \subseteq I$. It is required that $A \bigcap B = \phi$. A is called the *antecedent* of the rule while B is its *consequent*. We say that the rule $A \rightarrow B$ holds for T with *support* s, where s is the percentage of transactions in T that contain both A and B, and with *confidence* c, where c is the percentage of transactions containing A that also contain B, i.e, $support(A \bigcup B)/support(A)$.

It should be noted that our work presented in this paper focuses on the application of association analysis to tag annoation in music. We employ one of the many algorithms that find the association rules in a transactional database, the *Apriori* algorithm [1, 2], which generates association rules that satisfy user-specified *minimum support* and *minimum confidence*. For a simple but illustrative example of association rule generation, see [7].

2. VERIFYING TAG ANNOTATIONS

In our experience, we observe that tag annotation by music experts, though at a high cost, is relatively more accurate, due to their music expertise, as compared to the one performed by amateurs. Based on this, we propose to take advantage of the judgments and opinions of music experts for a tag annotation task. It is our belief that tags of high quality reveal inherent associations among music pieces in a repository and can be utilized in a tag annotation process. Our proposed approach to verifying tag annotations is conducted in four stages, as shown in Figure 1.

First, given an annotation task, a group of experts with extensive music background and experience are solicited. They examine the repository and select a set D comprised of those songs that best represent the repository as a whole. Then, they manually annotate this set using a predetermined and fixed set of tags. Consequently, the tag annotations contain expert knowledge and show authoritative judgments and opinions of those experts regarding the representative music pieces.

In the second stage, we treat the selected music pieces as a transactional database, where tags are the transactional items and music pieces are the transactions. We then invoke the *Apriori* [1] algorithm to find the association rules in it under the current minimum support and minimum confidence, both of which are specified in advance. We denote the resulting association rule set $R = \{r_1, r_2, \cdots, r_k\}$, where each r_i is called a *tag annotation rule*.

During the third stage, we introduce new music pieces whose annotations are to be verified. They could be samples from the same music repository, which would be the case where the task is to verify the annotations of a given music repository. They could also be an addition to the existing repository, which would happen when the repository grows and the newly added music pieces need to contain annotations that are consistent with the existing repository. We collectively call these new music pieces $M = \{m_1,$

m_2,$\cdots$, m_n}; they are annotated manually or automatically as discussed in Section 1.1. We represent each m_i as a set of tags.

In the last stage, we verify and adjust the annotations of M. First, we run the Scoring Algorithm and generate a set of evaluation measures, both of which are discussed in the following two subsections. Then, we examine the evaluation and adjust M accordingly. Once M reaches a desired level of agreement with R, and hence the repository, the whole process of verifying tag annotations comes to completion.

2.1 Scoring Algorithm

The *annotation score* $S(m_i)$ of a music piece m_i represents the number of rules in R that m_i satisfies. It is calculated as follows. For each tag annotation rule $A \rightarrow B$, if the song m_i contains both, the antecedent ($A \in m_i$) and the consequent ($B \in m_i$), then we increment the song's score $S(m_i)$ by 1. However, there could be situations where $A \rightarrow B$ and $A \rightarrow C$ coexist in R, representing the multi-label nature of tag annotation. If a music piece misses the first rule but satisfies the second one, its score still increments by 1 instead of 0. To achieve this, we iterate through R twice. At first, we build a list of rules (hit_list), which are satisfied by the song m_i, and increment $S(m_i)$ accordingly. Then we look for the rules which are not satisfied by the song's annotations, such as $A \rightarrow B$, where $A \in m_i$ but $B \notin m_i$. If their antecedents are not found in the hit_list, then we decrement the song's score $S(m_i)$ by 1 for each problematic rule.

Scoring Algorithm: used together with R during Stage 4.

```
1.   for each m_i ∈ M {
2.     S(m_i) = 0;
3.     for each rule r_i = (A → B) ∈ R
4.       if A ∈ m_i then
5.         if B ∈ m_i then {
6.           record the rule in hit_list
7.           S(m_i)++
8.         }
9.     for each rule r_i = (A → B) ∈ R
10.      if A ∈ m_i then
11.        if B ∉ m_i then
11.          if A ≠ any antecedent in the hit_list then
14.            S(m_i)--
15.  }
```

2.2 Evaluation Measures

If a music piece has a positive score, we say that it has a *sound tag annotation* (*STA*). Otherwise, we say that it has a *problematic tag annotation* (*PTA*). Furthermore, we attempt to distinguish music pieces that have a certain degree of ambiguity and subjectivity. A music piece has a *gray tag annotation* (*GTA*) if its annotation score is between [l,h], where l and h are user-specified range values. For instance, they can be -1 and +1 respectively. In our simulations below we use $[-2, 0]$ for this range. A music expert, depending on her/his musical expectation and experience, may set a different range.

We calculate four measures. The first is the *Problematic Tag Annotation Rate* (*PTAR*) for the annotation process. It is the ratio between the number of music pieces with PTA and the total number of music pieces. The second is the *Sound Tag Annotation Rate* (*STAR*), which is the ratio between number of STA music pieces and the total number of music pieces. These two rates represent the quality of the tag annotation process. It is obvious that the higher PTAR is, the worse the tag annotation quality; while for STAR it is the opposite. In addition, we calculate the *Gray Tag Annotation Rate* (*GTAR*), which is the ratio between the number of GTA music pieces and the total number of music pieces in the dataset. It represents the uncertainty in the annotation process. The fourth measure that we calculate is the *Zero Tag Annotation Rate* (*ZTAR*), which represents the percentage of music pieces that do not contain any of the tag annotation rules. These measures divide the whole music set into partitions and add up to 1.

3. SIMULATIONS

We are facing three major challenges when examining the effectiveness of our proposed approach to tag annotation verification.

It would be ideal to examine our approach using the process as depicted by Figure 1. However, such a process involves a great amount of financial and labor costs. Given our current situation and circumstances, it would be extremely hard, if not impossible, for us to deploy such a process, although involving a group of experts with extensive music background and experience could prove invaluable, as they would inevitably provide important feedback about our approach.

To make the situation worse, it is often the case that different music repositories and datasets vary in the sets of tags used to annotate their music pieces. Furthermore, association analysis captures each dataset's own associations that do not necessarily translate to others. Therefore, it is possible for two datasets to be completely incompatible in such a way that one could not be used to verify another.

Moreover, there is a lack of good quality datasets that could be used for benchmarking, although the MIR community is making efforts to come up with such datasets, which is evidenced by the Million Song Dataset [4, 9] and the Million Song Dataset Benchmarks [14].

Taking these issues into consideration, we design and implement a series of simulations to demonstrate the effectiveness of our proposed approach. Through these simulations, we aim to achieve three goals: (G_1) demonstrate that our approach is stable, in that it will not behave arbitrarily when given different music datasets or, put in another way, given similar music datasets, it should behave similarly; (G_2) assess the four music datasets using our evaluation measures and confirm that they maintain different relationships among their tags due to the differences in their annotation processes; and (G_3) confirm that, when the quality of annotations in a music dataset improves, our proposed measures reflect this improvement.

We run our simulations with different minimum support and confidence value pairs; for each minimum support value ranging from 5 to 95 we use a different minimum confidence ranging from 5 to 95.

3.1 Music Datasets

For our simulations, we use four datasets described below and presented in Table 1. They have been previously annotated with tags by music experts and amateurs.

The *CAL500* [18] dataset, denoted as S_{CAL500}, is a collection of 502 Western songs recorded by different artists. Each song is manually annotated by at least three human annotators using a vocabulary of 174 tags, which is distributed across six categories: Mood, Genre, Instrument, Song, Usage, and Vocal. All tags are manually generated under controlled experimental conditions. In our work, we use the "hard" annotations found in CAL500, which give a binary value for all tags in every song indicating whether a tag applies to a song.

The *CAL10K* [17] dataset, S_{CAL10K}, is comprised of meta-data collected from the Pandora website. It consists of 10,886 songs annotated by expert musicologists who maintain a high level of agreement. This dataset contains 1053 unique tags. Furthermore, it contains a set of 55 tags in common with S_{CAL500}. We use this information to find the same subset of tags in the remaining datasets.

The *Magnatagatune* [15], S_{MAGNA}, contains annotations of 21,642 music clips with a subset of 188 different tags. The annotations are collected through an on-line game, referred to as "TagATune", developed to collect tags for music and sound clips. Each clip, 29 seconds in length, is an excerpt of music provided by (magnatune.com) and (freesound.org). All of the tags in the collection have been verified, i.e. a tag is associated with a clip only if it is generated independently by more than two players. Moreover, only those tags that are associated with more than 50 clips are included in the collection. It is important to note that Magnatagatune has not been used as widely as CAL500 due to its size and skewed tag distribution.

The *Million Song Dataset* (*MSD*) [4], is the largest MIR dataset publically available to date. Its purpose is to encourage research of scalable solutions and to provide a reference dataset for benchmarking. Unlike all other datasets, MSD is a conglomeration of complimentary datasets. For our simulations, we use the portion provided by LastFM (www.lastfm.com) excluding the known duplicates. For this reason, we denote it as S_{LASTFM}.

Since the S_{CAL500} dataset is too small to produce good sample size, we perform our simulation 10 times and obtain an arithmetic mean for each of our evaluation measures, similar to the 10-fold cross validation. Each time we choose the training set at random, and the remainder of the dataset becomes the testing set. This is done to preserve the 30/70 split used in our simulations, as described below. The other three datasets are large enough and do not require this kind of repeated random sub-sampling validation.

Dataset name	Number of songs	Number of labels	Label cardinality	Songs with at least 2 tags
S_{CAL500}	502	174	26.04	502
S_{CAL10K}	10886	1053	11.88	10886
S_{MAGNA}	25863	188	3.46	18097
$S_{\text{MAGNA-CLN}}$	25863	188	4.74	18097
$S_{\text{MAGNA-ADJ}}$	25863	166	5.15	13991
S_{LASTFM}	449503	522366	15.94	449503
$S_{\text{LASTFM-CLN}}$	449503	1046	8.76	280922
$S_{\text{LASTFM-ADJ}}$	449503	287	7.13	315254

Table 1: Music datasets and their statistics. The *label cardinality* of a dataset is the arithmetic mean of the number of labels per music piece in the dataset. Datasets denoted by CLN undergo simple data cleaning, such as the removal of songs with label cardinality less than 1. The datasets denoted by ADJ undergo the adjustment process depicted by Stage 4 in Figure 1 and discussed in Simulation 3.

3.2 Simulation 1

This simulation demonstrates G_1, that our approach is stable. Here, we randomly split a dataset in half and see if the resulting halves, H_1 and H_2, behave similarly in terms of our evaluation measures as outlined in Section 2.2. In this simulation, for each half, we go through all of the sages depicted by Figure 1 except the review and adjust steps in stage 4. First, we split each half into two subsets at random: we call one (30%) the *training set*, it corresponds to D, and the other (70%) corresponds to M and we call it the *testing set*. Then we compare the results from the scoring Algorithm between H_1 and H_2.

3.3 Simulation 2

Similarly to Simulation 1, we randomly divide each music dataset into a training set (30%), from which we derive association rules, and a testing set (70%), which we score against the association rules. They respectively correspond to D and M in Figure 1. Then we compare our evaluation measures between different datasets and hope to confirm G_2 that they maintain different relationships among their tags.

In this simulation we also use one dataset to verify another. Since CAL10K provides 55 tags that appear in all four datasets, we reduce all datasets to just those tags. Then we use one dataset as the training set D and another as the testing set M and perform all steps outlined in Figure 1 except the adjustment cycle.

3.4 Simulation 3

To fully demonstrate stage 4, we adjust the two datasets, S_{MAGNA} and S_{LASTFM}, by amalgamating some equivalent tags into one, thus reducing the diversity of tags in the datasets. For example, we convert several tags like $\{instrument\ singer\ male\}$, $\{male\ singer\}$ and $\{male\ voices\}$ into one tag $\{male\ vocals\}$. After this adjustment, we run the scoring Algorithm and hope to observe that the new PTAR is lower while the new STAR is higher.

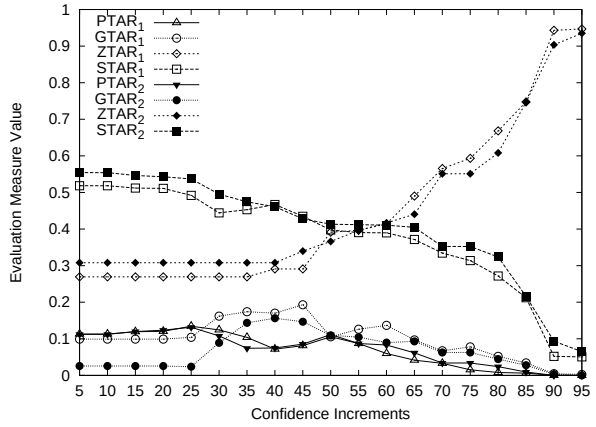


Figure 2: Comparison between H_1 (denoted by subscript 1) and H_2 (denoted by subsript 2) for S_{CAL10K} H_1 using our four evaluation measures.

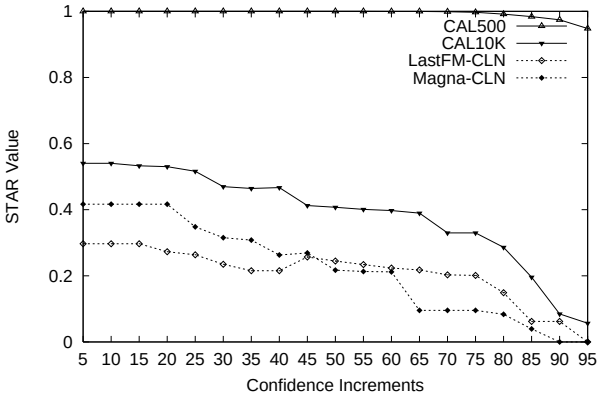


Figure 3: Comparison of STAR between four datasets.

4. RESULTS AND DISCUSSIONS

Simulation 1 demonstrates that our association-based tag annotation verifiction is stable. Figure 2 illustrates the different measure values that we obtain when we apply our approach to S_{CAL10K}. Here, the minimum support is set to 5% while the confident increases from 5% to 95%. We observe that STAR values are very similar at all confidence levels. The same applies to all other measures that we discuss in Section 2.2. We find similar results in the other three datasets across various minimum support thresholds and thus conclude our work toward G_1, i.e., our approach is stable. Due to space limit, we do not show all of the figures in this paper.

Simulation 2 applies our approach to all four music datasets. We obtain the values of our proposed four measures and report the STAR values for all four datasets in Figure 3. The figure shows that S_{CAL500} clearly achieves the highest STAR values, when compared to the other three datasets. The same argument is applied to other measures. For instance, we have observed lower PTAR values on S_{CAL500} but higher ones on $S_{\text{LastFM-CLN}}$. Due to the space consideration, we do not report all of the figures here.

Our simulations clearly show that dataset S_{CAL500} has a better tag annotation quality than the other three datasets. Some similar observations have also been made in previous

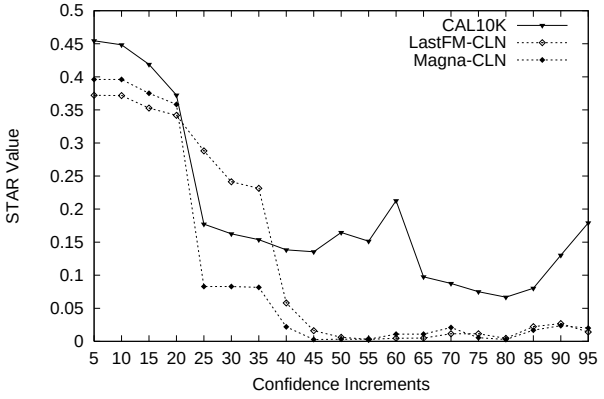


Figure 4: Using S_{CAL500} to verify the other datasets.

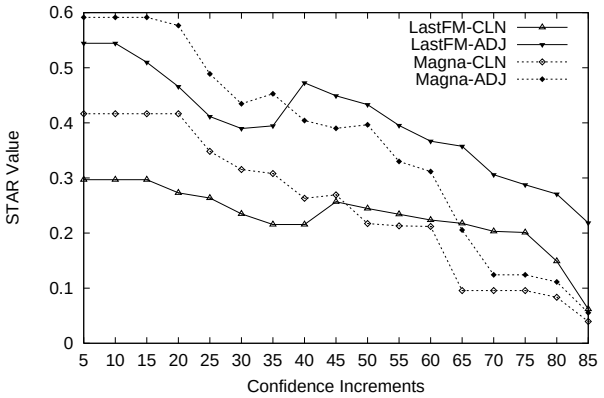


Figure 5: Comparison of STAR in two datasets before and after the adjustment step.

works, as mentioned in Section 1.1, and further suggested by the annotation processes as discussed in Section 3.1. Therefore, we used S_{CAL500} as the representative dataset D in Figure 1 to evaluate the other datasets, each considered as M in the same figure. As clearly shown in Figure 4, except for a few lower confidence ranges, S_{CAL10K} outperforms the other two datasets in terms of STAR. The same applies to the other measures, which are not shown due to space limitation.

Towards our goal (G_3), we adjusted the two datasets that were evaluated as the worst in Simulation 2, namely the $S_{\text{LastFM-CLN}}$ and the $S_{\text{Magna-CLN}}$. We then applied our approach to them again, as outlined in Stage 4 in Figure 1, hoping to see some improvements in terms of our evaluation measures. As presented in Figure 5, we clearly see that improvement. After the adjustment step, both datasets show better performance than before in terms of STAR. The same can be said about the other measures as well, which are not shown for the sake of space.

5. CONCLUSION

In this work, we have presented a novel approach to the verification of music tag annotation. We believe that there exist inherent associations among music tags that can be further utilized to verify and monitor a tag annotation process. The simulations presented here show the effectiveness of our approach.

There are several directions along which we can extend our work. It would be very interesting to explore whether we can use content-based information, such as MFCC or ZCR [5], in our analysis and verification process. We conjecture that this additional information will help improve our approach. Furthermore, we also plan to examine the individual rules that were generated for each music piece.

In addition, we could calculate the tag annotation rate for a specific category, such as style, mood and instrument. Furthermore, we could consider the representative music pieces of a single tag. For example, we could examine the tag annotation rules of the music genre *pop*. These rules may provide more insight into the nature of this genre and why a music piece is associated with it as opposed to others.

6. REFERENCES

[1] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, volume 22, pages 207–216. ACM, 1993.

[2] Rakeshl Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, volume 1215, pages 487–499. Morgan Kaufmann Publishers Inc., 1994.

[3] Thierry Bertin-Mahieux, Douglas Eck, Francois Maillet, and Paul Lamere. Autotagger: A model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2):115–135, 2008.

[4] 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. University of Miami, 2011.

[5] Antti Eronen. *Signal Processing Methods for Audio Classification and Music Content Analysis*. PhD thesis, Tampere University of Technology, Tampere, Finland, June 2009.

[6] Joe Futrelle and Stephen J. Downie. Interdisciplinary communities and research issues in music information retrieval. In *Proceedings of the 3rd International Society for Music Information Retrieval Conference*, pages 121–131, 2002.

[7] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., the second edition, 2006.

[8] Matthew D Hoffman, David M Blei, and Perry R Cook. Easy as CBA: A simple probabilistic model for tagging music. In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, volume 9, pages 369–374, 2009.

[9] 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, 2012.

[10] Steven R Ness, Anthony Theocharis, George Tzanetakis, and Luis Gustavo Martins. Improving automatic music tag annotation using stacked generalization of probabilistic svm outputs. In *Proceedings of the 17th ACM International Conference on Multimedia*, pages 705–708. ACM, 2009.

[11] Kerstin Neubarth, Izaro Goienetxea, Colin Johnson, and Darrell Conklin. Association mining of folk music genres and toponyms. In *ISMIR*, pages 7–12, 2012.

[12] Franois Pachet. Content management for electronic music distribution. *Communications of the ACM*, 46(4):71–75, 2003.

[13] Chris Sanden and John Z. Zhang. An empirical study of multi-label classifiers for music tag annotation. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 717–722, 2011.

[14] Alexander Schindler, Rudolf Mayer, and Andreas Rauber. Facilitating comprehensive benchmarking experiments on the million song dataset. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, pages 469–474, 2012.

[15] Klaus Seyerlehner, Gerhard Widmer, Markus Schedl, and Peter Knees. Automatic music tag classification based on block-level features. In *Proceedings of the Sound and Music Computing Conference*, pages 126–133, 2010.

[16] Jialie Shen, Wang Meng, Shuichang Yan, HweeHwa Pang, and Xiansheng Hua. Effective music tagging through advanced statistical modeling. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 635–642. ACM, 2010.

[17] Youngmoo E. Kim Tingle, Derek and Douglas Turnbull. Exploring automatic music annotation with acoustically-objective tags. In *Proceedings of the international conference on Multimedia Information Retrieval*, pages 55–62. ACM, 2010.

[18] Douglas Turnbull, Luke Barrington, David Torres, and Gert Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):467–476, February 2008.

[19] Kris West. *Novel Techniques for Audio Music Classification and Search*. PhD thesis, University of East Anglia, UK, September 2008.

THE ROLE OF AUDIO AND TAGS IN MUSIC MOOD PREDICTION: A STUDY USING SEMANTIC LAYER PROJECTION

Pasi Saari*, Tuomas Eerola*, György Fazekas†, Mathieu Barthet†, Olivier Lartillot*, Mark Sandler†
*Finnish Centre of Excellence in Interdisciplinary Music Research, University of Jyväskylä, Finland
†Centre for Digital Music, Queen Mary University of London, United Kingdom
*{firstname.lastname}@jyu.fi, †{firstname.lastname}@eecs.qmul.ac.uk

ABSTRACT

Semantic Layer Projection (SLP) is a method for automatically annotating music tracks according to expressed mood based on audio. We evaluate this method by comparing it to a system that infers the mood of a given track using associated tags only. SLP differs from conventional auto-tagging algorithms in that it maps audio features to a low-dimensional semantic layer congruent with the circumplex model of emotion, rather than training a model for each tag separately. We build the semantic layer using two large-scale data sets – crowd-sourced tags from Last.fm, and editorial annotations from the I Like Music (ILM) production music corpus – and use subsets of these corpora to train SLP for mapping audio features to the semantic layer. The performance of the system is assessed in predicting mood ratings on continuous scales in the two data sets mentioned above. The results show that audio is in general more efficient in predicting perceived mood than tags. Furthermore, we analytically demonstrate the benefit of using a combination of semantic tags and audio features in automatic mood annotation.

1. INTRODUCTION

Our daily experiences with music, together with strongly corroborated research evidence [1], suggest that music has a remarkable ability to induce as well as to express emotions or moods. For this reason, the mood associated with a musical piece is often a key aspect in music listening. This provides clear motivations for creating Music Information Retrieval (MIR) systems to organize, navigate or access music collections based on mood. These systems typically rely on mood models and appropriately selected machine learning techniques [2, 3]. Among several models proposed for emotions, the Circumplex model [4, 5] connecting mood terms to underlying emotion dimensions of valence (positive / negative) and arousal (active / passive) is one of the most popular [6]. On the other hand, Thayers variant [7] of this model suggests dimensions of tension and energy diagonal to arousal and valence. However, training machine learning models that automatically associate musical pieces with moods require high quality human mood annotations that are laborious to create, hence typically limited in amount.

Mood-related tags, i.e., free-form labels applied to artists, albums, tracks, etc., are abundantly available from popular online services such as Last.fm [1], while editorial track-level mood tags are vital in large production music catalogues. However, due to issues related to noise and ambiguity in semantic relations between tags, uncovering reliable mood representations from tag data requires typically filtering and semantic analysis [8, 9]. Previous research showed that semantically processed information using track-level Last.fm tags is congruent with listener ratings of valence, arousal, tension and various mood terms [10]. In a test set of 600 popular music tracks, moderate to high ($.47 < r < .65$) correlation was found using the Affective Circumplex Transformation (ACT) technique, that is based on Latent Semantic Analysis (LSA) and the circumplex model of emotions. These results outperformed several conventional semantic analysis techniques, and notably, raw tag frequency scores ($.16 < r < .47$). The robustness of ACT was also demonstrated in [11], by applying the technique to editorial tags from a production music library of about 250,000 tracks.

In a wider context, modelling mood, and thus estimating mood tags may be seen as a specific form of auto-tagging, which is a popular research topic in MIR. A system is typically trained using audio features extracted from a collection of tracks and their associated tags. Then, the trained model is utilised to label new untagged tracks automatically given their features. Typical auto-tagging studies have trained models independently for each tag [12–14], omitting semantic associations between tags, while results in [15] and [16] showed that post-processing auto-tags according to their semantic similarity increases the performance. These techniques have produced promising results for mood tags, possibly due to the use of cleanly-labeled tag data collected for research purposes. As shown in [10], a considerable semantic gap exists between raw crowd-sourced mood tags and verified listener ratings. However, semantic computing provides a promising direction, not yet exploited to the full extent in auto-tagging, for capturing reliable information from large tag collections.

Previous studies in auto-tagging have compared predict-

[1] http://www.last.fm

ed tags with human-labeled tags as a way to assess performance. By considering listener ratings as ground-truth rather than tags, in this paper we analytically compare auto-tags to actual tags as predictors. The two representations relate to two distinct assumptions in tag estimation: either human-labeled tags or audio is available for each new track. Our aim is to challenge these assumptions by highlighting the benefit of semantic computing in the context of music mood auto-tagging. Semantic Layer Projection (SLP) proposed in [17] provides a robust method for projecting the audio feature space to multi-dimensional semantic layer based on ACT. SLP followed by linear regression with the projected feature components outperformed state-of-the-art regression models in predicting listener ratings of valence in 600 tracks from Last.fm. In this paper we evaluate the benefits of SLP in auto-tagging using two corpora: tracks and crowd-sourced mood tags from Last.fm tags as well as tracks and curated editorial mood tags obtained from the I Like Music (ILM) production music catalogue. We predict listener ratings of moods in separate test sets extracted from these corpora.

The rest of the paper is organised as follows: Section 2 describes the tag data and the ACT technique for building the semantic space of moods based on the tags, the set of audio features, and the SLP technique for predicting mood in new tracks based on ACT and the features. Section 3 gives a detailed account of the experimental setup, the data sets used for SLP evaluation, baseline techniques, and the method for comparing mood prediction based on tag or audio information of new tracks. Section 4 shows the results of the experiments and conclusions are drawn in Section 5.

2. METHODOLOGY

2.1 Affective Circumplex Transformation

We used two sources of tracks and tags in the analysis: 259,593 tracks from Last.fm and 226,344 tracks from I Like Music (ILM) production music catalogue, associated with 357 and 288 mood terms, respectively. To create these data sets, tags associated to track sets from the two sources were first lemmatized and identified from a vocabulary of 560 mood terms, aggregated from mood words obtained from selected research papers in affective sciences, music psychology and MIR, as well as from the Allmusic.com web service. In both data sets, tracks with only one tag, and tags associated with less than 100 tracks were then excluded. Finally, the tag data was normalised using term frequency-inverse document frequency (TF-IDF) weights. A detailed account of the data sets and the above process is given in [10, 11].

The following process was applied to Last.fm and ILM sets separately. To uncover semantic similarity between individual mood terms, a low-rank approximation of the TF-IDF matrix was computed using Singular Value Decomposition (SVD) and Multidimensional Scaling (MDS) as in [10]. SVD decomposes a sparse TF-IDF matrix N into orthogonal matrices U and V, and a diagonal matrix S with singular values in decreasing order, such that $N = USV^T$. A rank k approximation of N is then computed by $\bar{N}_k = U^k S^k (V^k)^T$, where each row vector U_i^k represents the terms w_i with k relative weights for each dimension. Similarly, V_j^k represents track t_j as k relative weights. Based on the rank k approximation, dissimilarity between terms w_i and $w_{\hat{i}}$ can be computed using the cosine distance between the $U_i^k S^k$ and $U_{\hat{i}}^k S^k$ vectors. To represent mood terms explicitly in a low-dimensional space that resembles the arousal-valence space, MDS was applied on the term distances to obtain a three-dimensional configuration. The choice of using three dimensions instead of two is motivated by the debate around whether two dimensions is enough to capture relevant variance in moods. Past research have proposed various candidates for the third dimension, such as dominance, potency, or movement.

Next we applied the Affective Circumplex Transformation (ACT) to conform the MDS configuration to the space of *arousal* and *valence* (AV), using AV values of 101 mood terms given in [4, p. 1167] and [5, p. 54]. This technique takes advantage of the Procrustes transformation [18] involving translation, reflection, orthogonal rotation, and isotropic scaling using sum of squared errors as goodness-of-fit. The motivation for this is to *i)* increase the interpretability of the MDS configuration, and *ii)* enable direct prediction of arousal and valence from the semantic space. The technique yields a mood term configuration $x_i = (x_{1,i}, x_{2,i}, x_{3,i}), i = 1, ..., nterms$. A subset of Last.fm and ILM mood term configurations are visualised in Fig. 1 (with $k = 16$). The frequencies of the terms across tracks (co-occurence counts) range from 110 ("vindictive") to 79,524 ("chill") for Last.fm, and 346 ("narrative") to 39,892 ("uplifting") for ILM. Each track j was projected onto the resulting space by taking the Euclidean mean of the term positions, weighted by the sparse TF-IDF vector q_j of the track:

$$t_j = (\Sigma_i q_{j,i} x_i)/(\Sigma_i q_{j,i}). \quad (1)$$

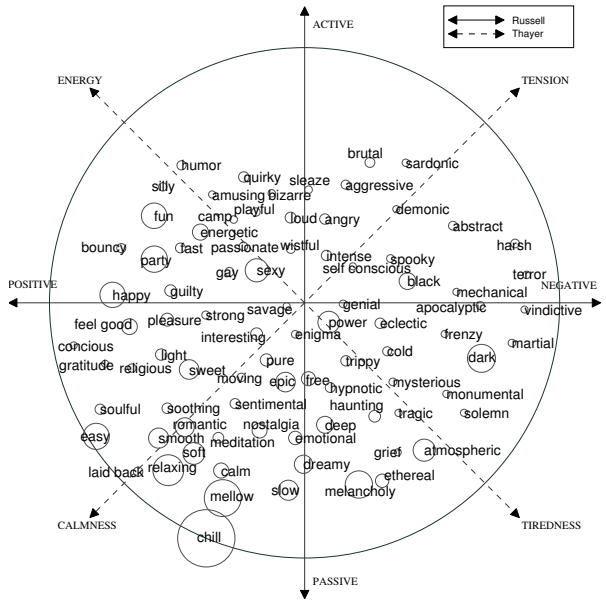
Finally, explicit mood-term-specific weights for the track with position t_j were computed using:

$$P_{j,i} = (x_i/|x_i|) \cdot t_j, \quad (2)$$

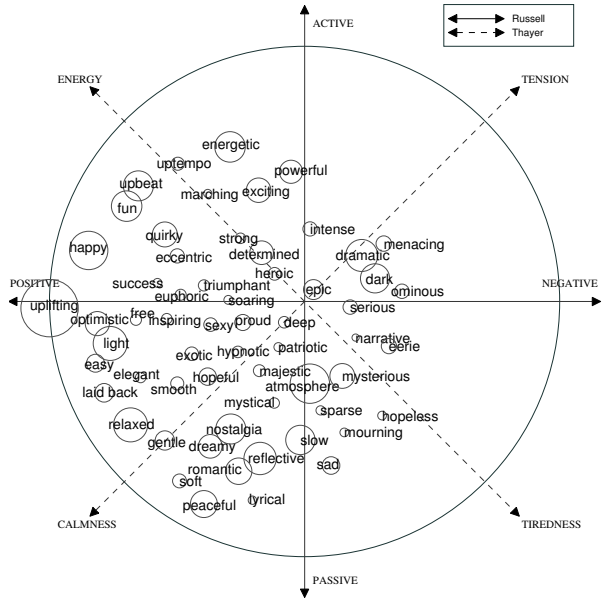
whereas arousal and valence for a track was estimated directly by using the positions along corresponding dimensions. Tension was obtained by projecting tracks along the direction (-1,1,0) as suggested in [7] (see Fig. 1). The analysis in [10] showed that the value of the rank parameter k in SVD computation has minor effect on ACT performance. Therefore we chose to use a heuristically selected value $k = 16$ in our analysis.

2.2 Audio Feature Extraction

Audio features describing dynamics (RMS energy, Low-energy ratio, Attack time, Attack slope), rhythm (Fluctuation pos. & mag., Event density, Pulse clarity, Tempo), pitch (avg. pitch, Chromagram unwrapped centroid), harmony (Key clarity, Mode [majorness], Harmonic change, Roughness), timbre (Brightness, Irregularity, Zerocrossings, Spectral Centroid, Flatness, Skewness, Entropy, Flux



(a) Last.fm.



(b) ILM.

Figure 1. Two first dimensions (valence–arousal) of the three-dimensional mood term configurations obtained with ACT ($k = 16$) for (a) Last.fm and (b) ILM.

and Spread), and structure (Spectral, Rhythmic and Registral repetition) as well as 13 MFCCs, Δ MFCCs, and $\Delta(\Delta)$ MFCCs were extracted from the data sets presented in Table 1 using the MIRtoolbox [19]. To characterise tracks using audio features, statistical means and standard deviations were computed for each feature extracted over short 50% overlapping time frames, yielding a 128 element feature vector for each track. For the features describing the rhythmic repetition and zero crossing rate, we used longer frame lengths of 2s, whereas for chromagram-based features such as the repetition of register, key clarity, centroid, mode, harmonic change, and roughness we used a frame length of 100ms. For other features the frame length was 46.4ms, except for low-energy ratio which is a track-level feature by definition.

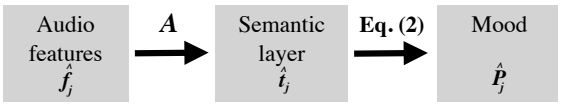


Figure 2. Mapping process in SLP for a novel track represented by audio features $\hat{f}_j$.

	Last.fm		ILM	
	SET10K	SET600	SET5K	SET205
# Tracks	9,662	600	4,692	205
# Terms	357	357	288	288
Term density (%)	1.59	2.43	1.86	2.38

Table 1. Statistics of the mood term sets.

2.3 Semantic Layer Projection

Semantic Layer Projection (SLP), originally proposed in [17], is a technique for the automatic annotation of audio tracks with mood using audio features. SLP is trained on a large collection of audio features and associated tag information. The difference between SLP and conventional auto-tagging is that audio features are not directly mapped to individual tags, but to three-dimensional semantic representation of the mood space obtained by ACT. The mapping is determined by a training stage using the Partial Least Squares (PLS) method. PLS has been found efficient at handling high dimensional and collinear input variables, and it is shown to be robust when using a large number of observations [20].

Given a set F of m audio features related to n tracks $F^{n\times m} = (f_1, f_2, ..., f_n)$, and a semantic layer representation $T^{n\times 3} = (t_1, t_2 ... t_n)$ for the corresponding tracks (see Eq. 1), the mapping matrix A between F and T is determined using PLS so that $T \approx AF$. We optimize the number of components in PLS by applying (50, 100)-fold cross-indexing [21]. Cross-indexing tackles problems of model overfitting when choosing the optimal parameterisation from several candidates.

The explicit mood of a previously unseen track represented by audio features $\hat{f}_j$ are estimated by first $\hat{t}_j = A\hat{f}_j$, and then by $\hat{P}_{j,i} = (x_i/|x_i|) \cdot \hat{t}_j$ as in Eq. 2. This process is summarised in Fig. 2.

3. EXPERIMENTAL SETUP

3.1 Data Sets

For both data sources, Last.fm and ILM, the semantic models are built from the full set of tracks (250,000 approx.), whereas mappings between audio features and the semantic space in SLP are trained on subsets of these large corpora, namely SET10K and SET5K. The performance of the model is evaluated using listener ratings of perceived moods in separate test sets: SET600 and SET205. Statistical measures of these data sets in terms of semantic mood content are summarised in Table 1.

SET10K consists of 9,662 tracks and was also used in [17]. The set was sampled from the Last.fm corpus in a balanced manner by *i)* optimising mood variance in terms

of track projections in the ACT space, *ii)* favouring tracks with many listeners according to Last.fm, and *iii)* including only unique artists. The audio content of SET10K consists of 15-30s Last.fm preview clips. The clips are typically samples of full tracks in the 128kB/s mp3 format, starting from 30s-60s into the beginning. We assume that these samples are sufficiently representative of the whole tracks. SET600 collected in [10] consists of 15s excerpts of 600 popular music tracks containing no overlapping artists with SET10K, and no tracks overlapping with the large Last.fm corpus. The set was fetched from Last.fm in a similar balanced manner as SET10K, with additional balancing across multiple popular music genres (jazz, pop, rock, electronic, folk and metal), and favouring tracks with many associated mood tags. SET600 was annotated in a listening test [10], with 59 participants rating the excerpts in terms of perceived mood expressed by music. Moods were rated in nine point bipolar Likert-scales for the mood dimensions of valence (negative / positive), arousal (calm / energetic), and tension (relaxed / tense), as well as in unipolar scales for individual mood terms atmospheric, happy, dark, sad, angry, sensual, and sentimental.

The 4,692 tracks from SET5K were picked up randomly from the ILM production music catalogue by *i)* keeping tracks with a duration of at least 60s (in order to discard short instances of the tracks), and *ii)* discarding instrumental stems, i.e. individual tracks from multitrack recordings. Six main genres were represented (jazz, dance, rock, electronic, folk and orchestral). 30s audio clip versions of the tracks were produced in the 128kB/s mp3 format. SET205, described in [11], consists of 205 clips of 30s duration from SET5K. The tracks were sampled in a similar fashion as for the Last.fm test set, but without taking listener statistics into account. The set was annotated by 46 participants in a similar manner as SET600, but for bipolar scales of valence (negative / positive), arousal (calm / energetic), tension (relaxed / tense), dominance (submissive / dominant), romance (cold / romantic), and humour (serious / funny) [11].

Features extracted from SET10K and SET5K were normalised using the z-score transform. All feature values with more than 5 standard deviations from zero were considered outliers and truncated to the extremes $[-5, 5]$. The features associated with SET600 and SET205 were then normalised according to the means and standard deviations of the larger feature sets.

3.2 Modelling Techniques

To show the efficiency of the mappings from audio features to the semantic layer, we compare SLP to two baseline techniques (BL1 and BL2) aiming at predicting mood ratings of e.g. valence, arousal, and tension in the test corpora. Prediction rates are computed as squared correlation coefficients (R^2) between the estimates and ratings over the test sets. The difference between the three techniques lies in how the semantic relationships between mood terms are exploited in the modelling. **BL1** uses mappings between audio features and individual mood terms directly, in order to predict mood ratings for the corresponding terms in the test corpora. This is analogous to the techniques used in [12–14]. **BL2** uses mappings between audio features and individual mood terms to predict each (term-track) pair in the test corpora. Test tracks are then projected using Eq. 1 and Eq. 2 based on the inferred tags. This is analogous to the techniques presented in [15, 16]. The **SLP** technique has been described in Section 2.3.

In short, BL1 does not use information about mood term relationships at all, while BL2 exploits the semantic information after producing a mapping from audio features to mood terms. SLP, on the other hand, maps audio features directly to the semantic layer.

Mappings in BL2 were trained for terms appearing at least ten times in SET10K and SET5K, amounting to 287 and 201 terms, respectively. Since valence, arousal, or tension are not explicitly modeled by BL1 (and no tags "valence" or "arousal" exist in either of the tag corpora), we use terms corresponding to the bipolar labels of the mood scales in the listening tests for modelling these ratings. Tags "positive", "energetic", and "relaxing" / "relaxed" were applied more often than tags "negative", "calm", and "tense" in both SET10K and SET5K, so we use the aforementioned tags to model the corresponding mood dimensions. Similarly, for dominance, romance, and humour that were rated in bipolar scales in SET205, we use tags "powerful", "romantic", and "funny".

Evaluating the role of tags and audio in predicting moods is achieved by comparing SLP and ACT prediction rates. While both of these techniques rely on the same semantic representation of moods, for each novel track, SLP uses only audio features and automatically inferred moods. ACT however uses actual tags associated with the track. We use these techniques in conjunction by computing the weighted mean of these two estimates for each track, and comparing that to the mood ratings. We vary the weights $[w, 1-w]$ ($w \in [0, 1]$) for the techniques so that the case $w = 0$ corresponds to using ACT, whereas the case $w = 1$ corresponds to using SLP.

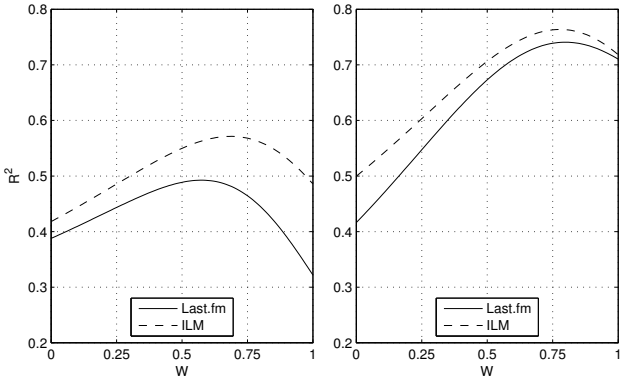
4. RESULTS AND DISCUSSION

4.1 Evaluation of SLP

Table 2 presents the comparison of SLP with the baseline methods. In case of Last.fm, prediction rates of SLP span from moderate ($R^2 = 0.248$ for happy) to considerably high ($R^2 = 0.710$ for arousal). SLP consistently outperforms both baseline methods, except in one case, where BL1 gives marginally higher performance for sad ($R^2 = 0.313$). The differences between the baseline techniques and SLP are however small for the arousal, angry, and sensual dimensions. We also note that valence and related moods (happy, sad, and angry) are the most difficult to predict with all of the models, and in turn, arousal is the easiest to predict. This is consistent with past studies in music emotion recognition [22]. Although BL1 suffers from the lack of explicit tags for valence, arousal, and tension to infer explicit predictions, results for the seven mood

		BL1	BL2	SLP
Last.fm	Valence	0.045	0.244	**0.322**
	Arousal	0.693	0.662	**0.710**
	Tension	0.198	0.469	**0.560**
	Atmospheric	0.075	0.541	**0.581**
	Happy	0.073	0.183	**0.248**
	Dark	0.264	0.314	**0.370**
	Sad	**0.313**	0.295	0.310
	Angry	0.475	0.465	**0.497**
	Sensual	0.505	0.523	**0.546**
	Sentimental	0.218	0.354	**0.390**
	Mean	0.286	0.405	**0.453**
ILM	Valence	0.156	0.330	**0.486**
	Arousal	0.680	0.672	**0.718**
	Tension	0.478	0.501	**0.588**
	Dominance	**0.461**	0.376	0.352
	Romance	0.274	0.301	**0.351**
	Humour	0.209	0.362	**0.502**
	Mean	.376	.424	**.499**

Table 2. Prediction rates (R^2) for the Last.fm and ILM test sets using SLP and two baseline methods (BL1 and BL2). For each dimension, best scores are reported in bold.



(a) Valence (b) Arousal

Figure 3. Prediction rate obtained when relying on different information in Last.fm and ILM test sets: tags ($w = 0$), audio ($w = 1$), or combination ($0 < w < 1$).

terms show that exploiting semantic associations between tags is highly beneficial. Moreover, as SLP outperforms BL2 for all mood dimensions, mapping tags to the semantic layer directly rather than projecting individual auto-tags to the layer is efficient.

In the case of the ILM data sets, our results show patterns that are highly consistent with those of Last.fm – in general SLP outperforms the baseline methods, while BL1 obtains the lowest performance, on average. However, the performance for valence is considerably higher ($R^2 = 0.486$) than for the Last.fm data set. A clear exception to this pattern is the higher performance of BL1 for dominance ($R^2 = 0.461$) compared to the other techniques. Since dominance is not directly captured either by the tags or the semantic layer dimensions, using other tags than "powerful" would have changed the modelling. In fact, tags "airy", "intimate", and "soft" yielded the highest performance for SLP ($R^2 > 0.57$), the tag "relaxed" yielded the highest performance for BL1 ($R^2 = 0.493$), and the tag "airy" yielded the highest performance for BL2 ($R^2 = 0.543$).

Overall, the results show the advantage of mapping audio features directly to a semantic layer to train predictive models for moods. This solution provides increased performance over methods not exploiting semantic associations at all, or projecting auto-tags to the semantic layer in a later stage, after mapping from audio features to mood tags.

4.2 Tags vs. Audio Features in Mood Prediction

To assess the importance of tags and audio in conjunction, systematic evaluation of using SLP and ACT separately or in conjunction using the weights was carried out. Overall, the results of such comparisons (see Fig. 3 and Table 3) first suggest that the predictions driven by audio features alone yield better performance. However, the combination of audio features and tags lead to a notable increase, especially for moods that are the most difficult for SLP (valence, happy, and sad). For Last.fm, the mean of the maximum performance when using audio and tags in conjunction is higher ($R^2 = 0.531$) compared to the individual use of tags ($R^2 = 0.334$) and audio ($R^2 = 0.453$). Similar patterns can be observed with the ILM data, for which audio content-based methods outperform tag-based methods for all mood scales (0.062 and 0.164 increases in mean R^2 when both audio and tags are used in conjunction, compared to audio and tags alone, respectively). As can be seen on Fig. 3, the optimal weight for the combination varies to a small degree in both data sets, but lies around 0.70 (mean). In other words, the best prediction of mood is achieved when the acoustic features are attributed a higher weight and are supplemented by tag data, both projected via a semantic space.

However, there are significant exceptions to the conclusions drawn from simply tallying up the prediction rates across the models and data sets. In the Last.fm data set, audio features are actually worse than tags in explaining the ratings of the valence and happy dimensions. This is in line with a number of previous studies in mood prediction with audio features, such as [22], and may have to do with the fact that valence is an elusive concept in music, and maybe particularly dependent on music genres. Further research that extends the mutual patterns between mood and genre is required to untangle such specific results.

5. CONCLUSIONS

In this study, we demonstrated that mood prediction is efficient when relying on large-scale music tag data and audio features, and is boosted by exploiting semantic modelling. The results suggest that higher prediction rates are achievable using the semantic layer projection (SLP) technique when compared to baseline techniques related to conventional auto-tagging that do not incorporate semantic modelling into mappings from audio features.

We conclude that building large-scale predictive models for moods in music can be done more efficiently for certain mood dimensions by relying on audio features rather than

		Tags	max(R^2)	w	Audio
Last.fm	Valence	0.388	0.492	0.57	0.322
	Arousal	0.416	0.741	0.80	0.710
	Tension	0.392	0.618	0.71	0.560
	Atmospheric	0.298	0.607	0.83	0.581
	Happy	0.357	0.429	0.53	0.248
	Dark	0.328	0.506	0.71	0.370
	Sad	0.300	0.393	0.58	0.310
	Angry	0.221	0.518	0.84	0.497
	Sensual	0.371	0.584	0.73	0.546
	Sentimental	0.271	0.422	0.72	0.390
	Mean	0.334	0.531	0.70	0.453
ILM	Valence	0.418	0.571	0.69	0.486
	Arousal	0.500	0.764	0.78	0.718
	Tension	0.497	0.667	0.69	0.588
	Dominance	0.271	0.386	0.73	0.352
	Romance	0.261	0.386	0.75	0.351
	Humour	0.437	0.590	0.67	0.502
	Mean	0.397	0.561	0.72	0.499

Table 3. Prediction rate for the Last.fm and ILM test sets using tags (ACT), audio (SLP), or a weighted combination.

associated tags. This is supported by the higher overall performance of audio compared to tags, and by the overall stable performance of the predictions between the models in two different data sets, crowd-sourced tags from Last.fm and a curated production music corpus (ILM). These data sets consisted of nearly 250,000 tracks each, out of which different subsets were carefully utilized in model training and evaluation. The results also imply that mood tags for novel tracks are not crucial for the automatic annotation of tracks along most mood dimensions. However, for moods related to valence, the use of tags yields a considerable increase in the predictive performance when combined with audio feature-based estimations. In the future we will factor in music genre to the approach presented here.

Acknowledgements This work was partly funded by the Academy of Finland (The Finnish Centre of Excellence in Interdisciplinary Music Research) and the TSB project 12033 - 76187 Making Musical Mood Metadata (TS/J002283/1).

6. REFERENCES

[1] J. A. Sloboda and P. N. Juslin. *Music and Emotion*, chapter Psychological Perspectives on Music and Emotion, pages 71–104. Oxford University Press, New York, 2001.

[2] Z. Fu, G. Lu, K. M. Ting, and D. Zhang. A survey of audio-based music classification and annotation. *IEEE Transactions on Multimedia*, 13(2):303 –319, 2011.

[3] M. Barthet, G. Fazekas, and M. Sandler. Multidisciplinary perspectives on music emotion recognition: Recommendations for content- and context-based models. In *Proc. of the 9th Int. Symposium on Computer Music modelling and Retrieval (CMMR)*, pages 492–507, 2012.

[4] J. A. Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178, 1980.

[5] K. R. Scherer. *Emotion as a multicomponent process: A model and some cross-cultural data*, pages 37–63. CA: Sage, Beverly Hills, 1984.

[6] T. Eerola and J. K. Vuoskoski. A review of music and emotion studies: Approaches, emotion models and stimuli. *Music Perception*, 30(3):307–340, 2012.

[7] R. E. Thayer. *The Biopsychology of Mood and Arousal.* Oxford University Press, New York, USA, 1989.

[8] C. Laurier, M. Sordo, J. Serra, and P. Herrera. Music mood representations from social tags. In *Proceedings of 10th International Conference on Music Information Retrieval (ISMIR)*, pages 381–86, 2009.

[9] M. Levy and M. Sandler. A semantic space for music derived from social tags. In *Proceedings of 8th International Conference on Music Information Retrieval (ISMIR)*, 2007.

[10] P. Saari and T. Eerola. Semantic computing of moods based on tags in social media of music. *IEEE Transactions on Knowledge and Data Engineering*, In press 2013.

[11] P. Saari, M. Barthet, G. Fazekas, T. Eerola, and M. Sandler. Semantic models of mood expressed by music: Comparison between crowd-sourced and curated editorial annotations. In *IEEE International Conference on Multimedia and Expo (ICME 2013): International Workshop on Affective Analysis in Multimedia*, 2013.

[12] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):467–476, 2008.

[13] M. I. Mandel and D. PW Ellis. A web-based game for collecting music metadata. *Journal of New Music Research*, 37(2):151–165, 2008.

[14] M. I. Mandel and D. P. Ellis. Multiple-instance learning for music information retrieval. In *Proceedings of 9th International Conference of Music Information Retrieval (ISMIR)*, pages 577–582, 2008.

[15] R. Miotto and G. Lanckriet. A generative context model for semantic music annotation and retrieval. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(4):1096–1108, 2012.

[16] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: A model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2):115–135, 2008.

[17] P. Saari, T. Eerola, G. Fazekas, and M. Sandler. Using semantic layer projection for enhancing music mood prediction with audio features. In *Sound and Music Computing Conference*, 2013.

[18] J. C. Gower and G. B. Dijksterhuis. *Procrustes problems*, volume 3. Oxford University Press Oxford, 2004.

[19] O. Lartillot and P. Toiviainen. A matlab toolbox for musical feature extraction from audio. In *Proceedings of the 10th International Conference on Digital Audio Effects*, 2007.

[20] S. Wold, M. Sjöström, and L. Eriksson. Pls-regression: a basic tool of chemometrics. *Chemometrics and intelligent laboratory systems*, 58(2):109–130, 2001.

[21] P. Saari, T. Eerola, and O. Lartillot. Generalizability and simplicity as criteria in feature selection: Application to mood classification in music. *IEEE Transactions on Speech and Audio Processing*, 19(6):1802 –1812, 2011.

[22] Y. H. Yang, Y. C. Lin, Y. F. Su, and H. H. Chen. A regression approach to music emotion recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):448–457, 2008.

Poster Session 2

CONVERTING PATH STRUCTURES INTO BLOCK STRUCTURES USING EIGENVALUE DECOMPOSITIONS OF SELF-SIMILARITY MATRICES

Harald Grohganz, Michael Clausen
Bonn University
{grohganz,clausen}@cs.uni-bonn.de

Nanzhu Jiang, Meinard Müller
International Audio Laboratories Erlangen
{nanzhu.jiang,meinard.mueller}@audiolabs-erlangen.de

ABSTRACT

In music structure analysis the two principles of repetition and homogeneity are fundamental for partitioning a given audio recording into musically meaningful structural elements. When converting the audio recording into a suitable self-similarity matrix (SSM), repetitions typically lead to path structures, whereas homogeneous regions yield block structures. In previous research, handling both structural elements at the same time has turned out to be a challenging task. In this paper, we introduce a novel procedure for converting path structures into block structures by applying an eigenvalue decomposition of the SSM in combination with suitable clustering techniques. We demonstrate the effectiveness of our conversion approach by showing that algorithms previously designed for homogeneity-based structure analysis can now be applied for repetition-based structure analysis. Thus, our conversion may open up novel ways for handling both principles within a unified structure analysis framework.

1. INTRODUCTION

The task of music structure analysis with the objective of partitioning a given audio recording into temporal segments and of grouping these segments into musically meaningful categories constitutes a central task in the field of music information retrieval [14]. Because of different structure principles including temporal order, repetition, contrast, variation, and homogeneity, finding the musical structure is a challenging and often ill-defined problem [18]. In particular, the two principles of repetition and homogeneity have been in the focus of previous research efforts [14, 15]. On the one hand, repetition-based methods target at identifying recurring patterns and, on the other hand, homogeneity-based methods try to determine passages that remain unchanged with respect to some musical property. When converting the given audio recording into a suitable feature sequence and then deriving a self-similarity matrix (SSM), repetitions typically lead to path-like structures, whereas homogeneous regions yield block-like structures. In previous research, numerous extraction and clustering techniques have been proposed that either allow for handling path structures or block structures, see, e g., [1, 5, 8, 12, 13, 14, 15]. However, dealing with both structural elements at the same time has turned out to be a challenging or even irreconcilable task.

Only few approaches that try to apply several segmentation principles at the same time exist. In [13], a unifying optimization scheme that jointly accounts for path and block structures is proposed. In [17], structural changes with regard to path and block elements are captured to derive segment boundaries. In [4, 8], approaches for homogeneity-based structure analysis are introduced, where smoothing and clustering techniques are applied for enhancing the block structure in a pre-processing step. A very interesting research direction is sketched in [16], where an audio recording is locally classified according to the properties of being repetitive or homogeneous with the goal to locally adapt the segmentation strategy.

Along these lines of research, we deal in this paper with the task of converting a repetition-based into a homogeneity-based structure analysis problem. Opposed to [4, 8], who try to enhance already latent block structure by applying smoothing and image processing techniques, we generate the block structures from path structures. As our main technical contribution, we propose a novel procedure that is based on an eigenvalue decomposition of the SSM. We show that certain path structures induce some disjoint properties of the supports (non-zero entries) of the eigenvectors. This in turn allows us to generate block-like structures when converting back the suitably clustered eigenvectors into some SSM. A typical result of our procedure is shown in Figure 1, which shows the original path structure in (b) and the resulting block structure in (e). The underlying piece of music has the musical form [1] $A_1B_1A_2B_2C_1C_2D_1D_2A_3B_3A_4B_4$. Note that in our procedure an explicit extraction of the path structure, often a fragile step used in repetition-based structure analysis, is not necessary.

The general idea of using eigenvalue decompositions of SSMs with applications to audio segmentation is not new, see e. g. [2]. However, in [2] this techniques is used for the purpose of dimensionality reduction and clustering, whereas we exploit specific properties of the eigenvectors

[1] As in [14], musical parts are denoted by the letters $A, B, C, \ldots$ in the order of their first occurrence, where indices are used to indicate repetitions.

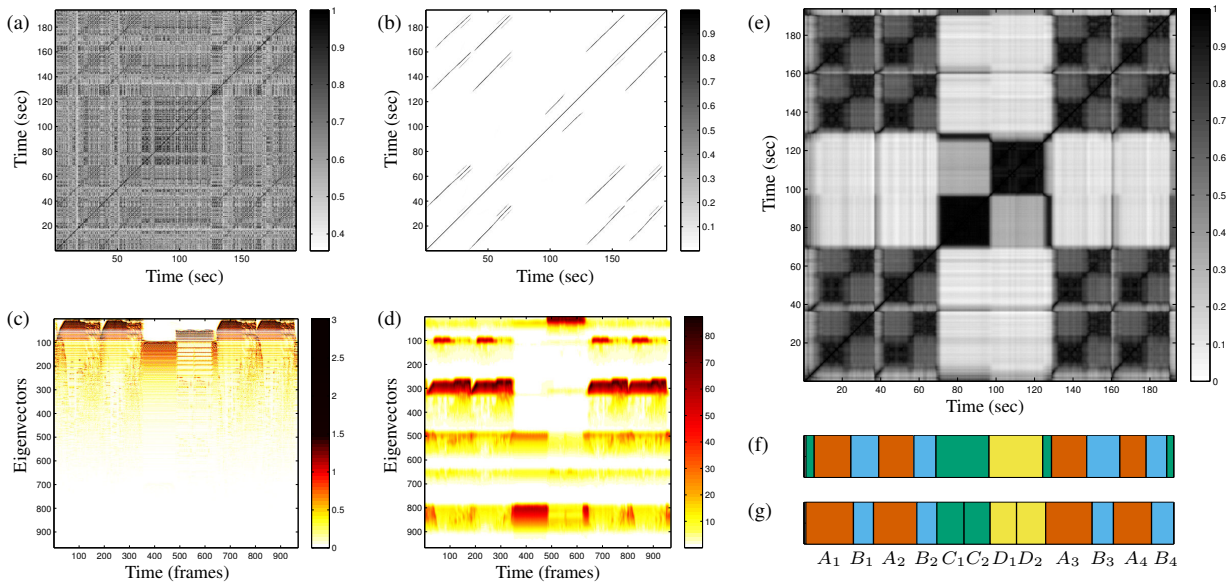


Figure 1: Illustration of the algorithmic pipeline for converting path into block structures. **(a)** Original SSM. **(b)** Path-enhanced SSM as typically used for repetition-based structure analysis. **(c)** Eigenvectors (rows) of the SSM weighted and sorted by the corresponding eigenvalues. **(d)** Eigenvectors (rows) after clustering and post-processing. **(e)** SSM obtained from (d). **(f)** Structure analysis result obtained from (e) applying some homogeneity-based clustering procedure. **(g)** Manually labeled structure (ground truth).

for converting path into block structures.

We demonstrate the effectiveness of our conversion approach by discussing a number of explicit examples and by presenting some quantitative evaluation based on two known datasets. In particular, we show that upon our conversion standard clustering procedures that are designed for homogeneity-based structure analysis can then be applied for repetition-based structure analysis. As a results, our conversion may open up novel ways for combining different music segmentation principles at an early stage of an audio structure processing pipeline.

In the remainder of this paper, we first describe the algorithmic details for our conversion procedure (Section 2), then report on our experiments (Section 3), and conclude with an outlook on future research directions (Section 4).

2. ALGORITHMIC PIPELINE

In this section, we describe our procedure for converting SSMs with path-like structures into SSMs with block-like structures, see also Figure 1 for an illustration of the overall pipeline. We first summarize a procedure for computing an SSM with path structures as typically used for repetition-based structure analysis (Section 2.1). Such matrices constitute the input of our conversion procedure. We then describe some properties of the eigenvectors obtained from such matrices (Section 2.2) and show how the eigenvectors can be used to derive SSMs with block-like structures (Section 2.3).

2.1 Computing SSMs with Path Structures

Repeating segments in music often share the same melodic and harmonic progression while showing differences in instrumentation and timbre. Therefore, in most approaches for repetition-based structure analysis, the audio signal is converted into twelve-dimensional chroma-based audio features, which closely correlate to the aspect of harmony and have become a widely used tool in processing and analyzing music data [3]. In the following, we use a chroma variant referred to CRP (Chroma DCT-Reduced Log Pitch) features [2], which show a high degree of invariance to changes in timbre [10]. In our experiments, we adapt the feature rate according to the length of the considered audio recording, which results in feature rates (features per second) between 2 Hz and 6 Hz. Normalizing the features, we use the inner product as a similarity measure to compute a self-similarity matrix $\mathcal{S}$ by comparing the elements of the feature sequence in a pairwise fashion, see Figure 1a.

To further enhance the path structure of $\mathcal{S}$, one typical procedure is to apply some kind of smoothing filter along the direction of the main diagonal, resulting in an emphasis of diagonal information in $\mathcal{S}$ and a denoising of other structures. In our implementation, we use a smoothing variant similar to [12], which can deal with local tempo variations. Furthermore, we apply image processing and thresholding techniques to eliminate short and weak path fragments, see also [17] for similar strategies.

The resulting path-enhanced SSM, as illustrated by Figure 1b, constitutes the input of our conversion algorithm. Note that the implementation details to obtain the path-enhanced SSM are not important at this stage. Our conversion procedure is generic and works well as long as the SSM has a relatively sparse structure only showing the most relevant paths.

[2] An implementation of these features is available at `www.mpi-inf.mpg.de/resources/MIR/chromatoolbox/`, see also [11].

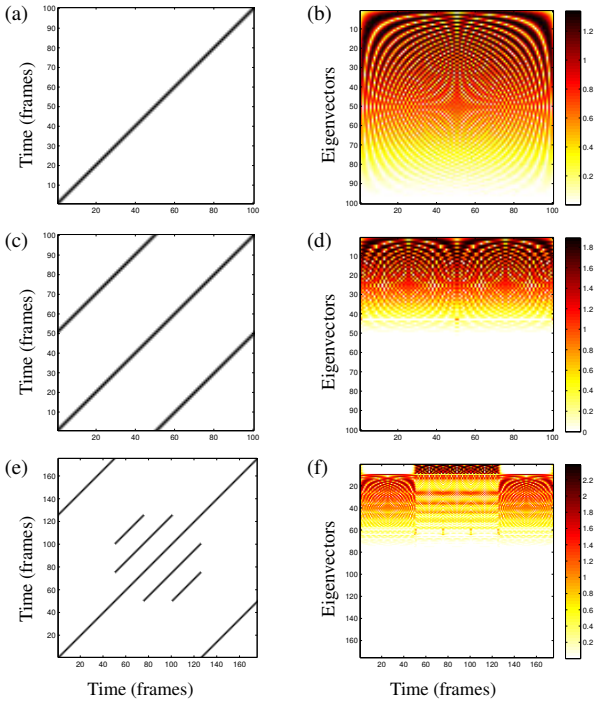


Figure 2: Various SSMs with path structures (**left**) and corresponding eigenvectors (**right**). The figures shows the eigenvectors in some transposed (row-wise) and weighted (multiplied by the corresponding eigenvalue) form. Furthermore, the eigenvectors are sorted according to decreasing eigenvalues. **(a)/(b)** SSM I_K^ϵ. **(c)/(d)** SSM reflecting the musical form A_1A_2. **(e)/(f)** SSM reflecting the musical form $A_1B_1B_2B_3A_2$.

2.2 Eigenvalue Decomposition

When comparing the elements of a feature sequence in a pairwise fashion using a symmetric similarity measure results in a symmetric SSM $\mathcal{S}$. This property may be lost by applying enhancement and image processing techniques as used in Section 2.1. However, one can restore the symmetry by considering $\frac{1}{2}(\mathcal{S}+\mathcal{S}^\top)$ instead of $\mathcal{S}$, where $\mathcal{S}^\top$ denotes the transposed matrix of $\mathcal{S}$. Doing so, we may assume in the following that $\mathcal{S}$ is a symmetric matrix of dimension $N \times N$ for some $N \in \mathbb{N}$.

Next, we apply an eigenvalue decomposition of the symmetric matrix $\mathcal{S}$ and investigate the properties of the resulting eigenvectors. Principle component analysis tells us that there exists a real-valued diagonal matrix $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ and an orthogonal matrix E such that $\mathcal{S} = EDE^\top$, where the n^{th} column e_n of E is an eigenvector of $\mathcal{S}$ with eigenvalue λ_n, i. e., $\mathcal{S}e_n = \lambda_n e_n$.

In our scenario, we assume that the matrix $\mathcal{S}$ consists of path-like structures, where a prototype of a path of length K may be modeled by the matrix [3]

$$I_K^\epsilon := \begin{pmatrix} & & & \epsilon & 1 \\ & & \epsilon & 1 & \epsilon \\ & \iddots & \iddots & \iddots & \\ \epsilon & 1 & \epsilon & & \\ 1 & \epsilon & & & \end{pmatrix} \in \{0, \epsilon, 1\}^{K \times K}.$$

[3] Opposed to existing conventions, we enumerate in this papers the rows of the matrix from bottom to top with the aim to better match the visualizations of the SSMs in the figures.

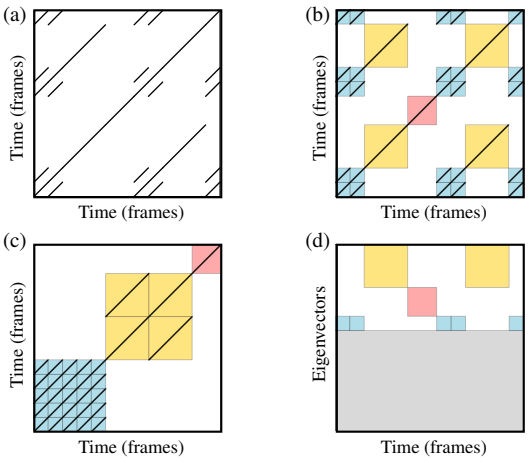


Figure 3: **(a)** SSM $\mathcal{S}$ with a path structure corresponding to the musical form $A_1A_2B_1C_1A_3A_4B_2A_5$. **(b)** SSM with highlighted block structure. **(c)** SSM in some normalized form after applying some permutation. **(d)** Illustration of the support of the suitable sorted and transposed eigenvectors.

In this matrix the non-specified entries have the value 0, see also Figure 2a. Intuitively, each path consists of a diagonal matrix with large entries on the main diagonal (represented by the value 1 in I_K^ϵ) and with decreasing entries on the diagonals above and below the main diagonal (represented by the value ϵ in I_K^ϵ). Such paths typically arise when applying path enhancement strategies based on smoothing techniques.

In [6], an explicit eigenvalue decomposition of the matrix I_K^ϵ is described. The eigenvalues are

$$\lambda_k = 1 + \epsilon \cdot 2\cos\tfrac{k\pi}{K+1}, \qquad 1 \le k \le K, \tag{1}$$

with corresponding eigenvectors

$$e_k = \left(\sin\tfrac{k\pi}{K+1}, \sin\tfrac{2k\pi}{K+1} \ \ldots, \sin\tfrac{Kk\pi}{K+1}\right)^\top. \tag{2}$$

In particular, note that the entries of the eigenvectors are non-zero, see also Figure 2b for an illustration. This property, as we will see, becomes crucial for converting paths into blocks.

The matrix I_K^ϵ constitutes the basic building block for SSMs with more general path structures. Instead of a mathematically rigorous treatment, which is beyond the scope of this paper, we explain the general case by means of an illustrative example. Let $\mathcal{S}$ be the SSM shown in Figure 3a, which has the path structure corresponding to the musical form $A_1A_2B_1C_1A_3A_4B_2A_5$. The desired block structure is indicated in Figure 3b. By applying a suitable permutation matrix $\mathcal{P}$, it can be shown that $\mathcal{S}$ can be converted into a matrix $\mathcal{S}' := \mathcal{P}\mathcal{S}\mathcal{P}^{-1} = \mathcal{A} \oplus \mathcal{B} \oplus \mathcal{C}$, which is the direct sum of three matrices $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ corresponding to the musical parts A, B, and C (inclusive repetitions), respectively. This is illustrated by Figure 3c. As for the eigenvalue decomposition, it can be shown that the eigenvalues of $\mathcal{S}'$ are given by the eigenvalues of the matrices $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. Furthermore, the eigenvectors of $\mathcal{S}'$ are obtained by suitably extending the eigenvectors of the matrices $\mathcal{A}$,

$\mathcal{B}$, and $\mathcal{C}$ by zero entries. As a result, the supports (non-zero entries) of eigenvectors coming from different summands $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ in $\mathcal{S}'$ are disjoint. This fact is illustrated by Figure 3d. Furthermore, the repetitions induce substructures in the summands $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. Each such substructure can be expressed by a suitable Kronecker product with an all-one matrix and a matrix of the form I_K^ϵ. For example, the matrix $\mathcal{B}$ corresponds to two repeating parts and can be expressed by

$$\mathbb{1}^{R\times R} \otimes I_K^\epsilon = \begin{pmatrix} I_K^\epsilon & I_K^\epsilon \\ I_K^\epsilon & I_K^\epsilon \end{pmatrix}$$

with $R = 2$ being the number of repetitions, $\mathbb{1}^{R\times R}$ being the all-one $R \times R$ matrix, and $\otimes$ being the Kronecker product. Note that the rank of $\mathcal{B}$ is K, so that in the case of $R = 2$ half of the eigenvalues are zero (therefore, the corresponding eigenvectors are not uniquely determined). Also, the permutation $\mathcal{P}$ is not known in practice. In our pipeline, we multiply the normed eigenvectors with their corresponding eigenvalue and arrange the modified eigenvectors according to their length in decreasing order. (The eigenvectors to eigenvalue 0 are irrelevant here.)

To build up some more intuition, let us consider the examples shown in Figure 2. The case of I_K^ϵ and its eigenvalue decomposition is illustrated by (a)/(b) of Figure 2, whereas the case of two repeating segments ($\mathbb{1}^{2\times 2} \otimes I_K^\epsilon$) is shown in (c)/(d). A third example corresponding to the musical form $A_1B_1B_2B_3A_2$ is shown in (e)/(f) of Figure 2. Here, the disjointness property of the supports for the eigenvalues that belong to different parts is visible. Note that the permutation matrix $\mathcal{P}$ is not known and that the eigenvectors are not ordered according to the musical parts they belong to. Furthermore, in practical applications the path structures may be noisy and distorted so that the discussed properties of the eigenvectors are not strictly fulfilled.

2.3 Deriving SSMs with Block Structures

Let $N \in \mathbb{N}$ denote the dimension of the eigenvectors, which also coincides with the number of frames. As indicated by Figure 1c, we form a matrix by defining its rows to be the transposed eigenvectors weighted and sorted by the corresponding eigenvalues. We denote this $N \times N$ matrix by $\mathcal{E}$. As discussed above, the path structure of the SSM is reflected by the support properties of the rows. In theory (assuming an ideal path structure as discussed before), two rows either have the same support (when corresponding to the same repeating musical part) or have disjoint supports (when corresponding to repeating, but different musical parts). Furthermore, the support of an eigenvector reveals all frames that belong to repeating segments of the same musical part (e. g., the frames of all A-part segments).

Motivated by this observation, we consider the columns $\mathcal{E}(n)$ of $\mathcal{E}$ as features, $n \in [1 : N]$. This yields a feature sequence $\mathcal{E}(1), \ldots, \mathcal{E}(N)$, which, in turn, can be used to define a self-similarity matrix $\mathcal{S}(\mathcal{E})$. The properties of the eigenvalues imply that two features $\mathcal{E}(i)$ and $\mathcal{E}(j)$ are similar if the frames i and j belong to repeating segments of the same musical part (or to frames of non-repeating segments), and dissimilar otherwise. As a consequence, the matrix $\mathcal{S}(\mathcal{E})$ has the desired block structure.

To make this procedure applicable for real data, we post-process the matrix $\mathcal{E}$ prior to forming the self-similarity matrix. To this end, we first replace each entry e of $\mathcal{E}$ by $\log(10|e| + 1)$ to prevent overrating the most-repeated segment. Then we apply a standard k-means clustering procedure [4] to rearrange the eigenvectors (rows of $\mathcal{E}$) so that vectors corresponding to similar structures are adjacent. Then we smooth the rearranged matrix in both directions, horizontally as well as vertically, see Figure 1d. Here, the horizontal smoothing balances out the values of the non-zero entries in the eigenvectors, whereas the vertical smoothing introduces robustness to local distortions. [5] Denoting the smoothed matrix by $\mathcal{E}'$, we compute the self-similarity matrix as above to obtain $\mathcal{S}^{\mathrm{Block}} = \mathcal{S}(\mathcal{E}')$. This matrix constitutes our final result, see Figure 1e.

3. EXPERIMENTS

To show how our conversion approach behaves on real data, we now discuss a number of explicit examples (Section 3.2) and report on some quantitative experiments (Section 3.3). Note that optimizing and investigating the specific role of the various parameters is not in the scope of this paper. Rather than numerically improving a specific structure analysis result, our main goal is to highlight the conceptual novelty of our approach. In particular, we demonstrate that procedures that are designed for homogeneity-based structure analysis (as the one described in Section 3.1) can now be applied for repetition-based structure analysis thanks to our conversion procedure.

3.1 Structure Analysis Procedure

As a typical example approach, we consider the homogeneity-based structure analysis procedure as described in [5], where a given self-similarity matrix is decomposed into a prototype matrix and an activation matrix using non-negative matrix factorization (NMF). Looking at maximizing entries in the activation matrix yields a frame-wise classification of the columns of the SSM, which in turn can be used to assign a class label to each frame. A segment is then defined as a maximal run of consecutive frames having the same class label, see [5] for more details and Figure 1f for an example.

In our experiments, we used an NMF-variant with additional sparseness constraints [7] setting the sparseness parameter to $4 \cdot \mathrm{mean}(\mathcal{S}^{\mathrm{Block}})$ and the rank parameter to 6 (assuming at most six different musical parts). The procedure was then applied to the matrix $\mathcal{S}^{\mathrm{Block}}$.

[4] In our implementation, we used 6 clusters. Our experiments showed that any number between 5 and 20 led to similar results.

[5] In our experiments, we used Gaussian smoothing using an adaptive window size vertically and 7 frames horizontally. Again these values are not crucial here.

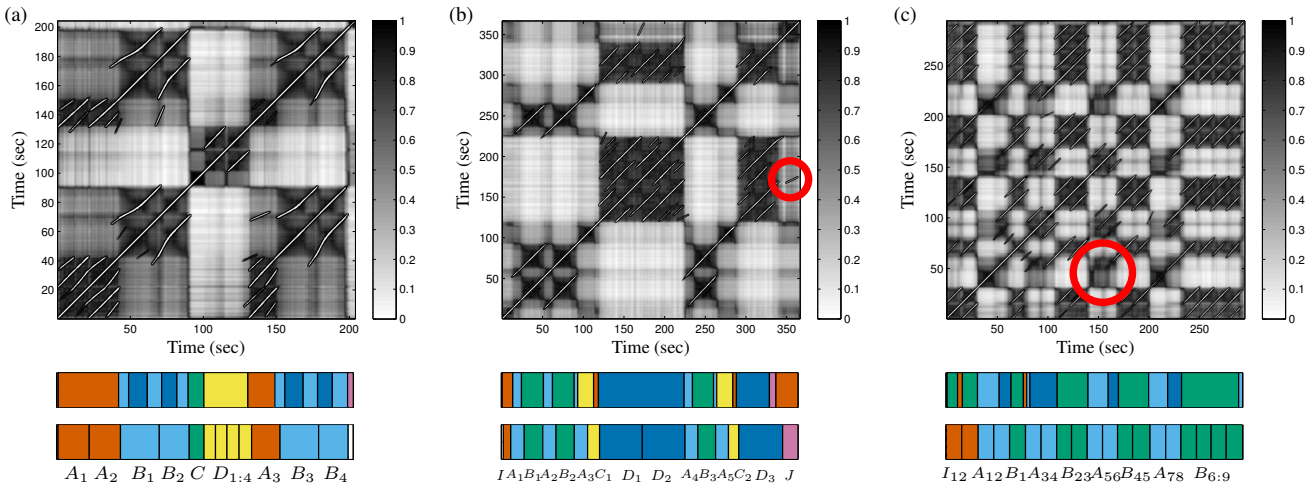


Figure 4: Results for three different audio recordings. The figure shows the computed block matrix overlaid with the path structure of the input matrix (top), the computed structure analysis results (middle) and the manually generated structure annotations (bottom). **(a)** Hungarian Dance No. 5 by Johannes Brahms. **(b)** March No. 1 from Op. 39 (Pomp and Circumstance) by Edward Elgar. **(c)** The song "The winner takes it all" by ABBA.

3.2 Qualitative Evaluation

We now discuss some specific examples to show the potential and the limitations of our conversion procedure. We start with our running example shown in Figure 1, which is a recording of the Waltz No. 2 from the Suite for Variety Orchestra by Dmitri Shostakovich. Using the path matrix $\mathcal{S}$ as shown in Figure 1b as input, our conversion procedure outputs the block matrix $\mathcal{S}^{\text{Block}}$ shown in Figure 1e. As the figure illustrates, path structures of repeating segments have been correctly converted into blocks. For example, the four repetitions of the combined AB-part are clearly visible as path structure in Figure 1b and as block structure in Figure 1e. Furthermore, Figure 1f shows the computed structure annotation obtained from $\mathcal{S}^{\text{Block}}$, whereas Figure 1g shows a manually generated structure annotation. Indeed, the homogeneity-based clustering approach applied to $\mathcal{S}^{\text{Block}}$ produced a reasonable repetition-based structure analysis result. Only subsequent repeating parts such as the two D-parts D_1D_2 (which are clearly reflected by paths in Figure 1b) have not been resolved by our frame-based labeling approach. Also, note that, because of significant musical variations in harmony and melody, the two repeating C-parts are neither reflected by paths nor by blocks.

Next, we consider the three examples of Figure 4. For each example, the computed block matrix $\mathcal{S}^{\text{Block}}$ overlaid with the original path structure inputted to our conversion procedure is shown. Also the structure annotations obtained from $\mathcal{S}^{\text{Block}}$ as well as the manually generated "ground truth" annotations (for comparison) are shown. The first example shown in Figure 4a is a recording of the Hungarian Dance No. 5 by Johannes Brahms. The A-part as well as the B-part segments are well reflected in the block structure despite of some distortions and inconsistencies in the path structure. Also, tempo differences between B-part segments (B_2 and B_4 are played faster than B_1 and B_3) still led to meaningful block structures. As in the previous Shostakovich example, the subsequent repeating A-part and D-part segments were not subdivided as a result of the purely frame-based labeling procedure. Also note that even in the path representation only the repetitions D_1D_2 and D_3D_4 were captured, but not the finer grained subdivision (because of the chosen temporal resolution induced by the parameter setting). Finally, the B-part segments were further subdivided by our structure analysis procedure, illustrating an over-segmentation as typical for automated structure analysis methods [9].

The example shown in Figure 4b is based on a recording of the March No. 1 from Op. 39 by Edward Elgar. As before, one can say that overall the computed block structure correspond well to the inputted path structure. The erroneously extracted small path fragment indicated by the red circle has no major influence on the computed block structure as well as on the final structure. This indicates that our conversion procedure is, at least to some degree, robust to local distortions and noise. In general, our procedure tends to yield better results when the inputted path structure is sparse, thus requiring a denoising/smoothing and thresholding step to enhance the path structure as is also done in most repetition-based structure analysis approaches [1, 14].

The final example is a recording of the song "The winner takes it all" by ABBA, see Figure 4c. With this example, we want to indicate that missing path relations as marked by the red circle may be "recovered" in the block structure. Since the eigenvalue decomposition is a *global* analysis of the entire matrix $\mathcal{S}$, local deviations and missing relations are balanced out, thus enforcing some kind of transitivity on the block level.

3.3 Quantitative Evaluation

Finally, we quantitatively evaluated and compared our overall structure analysis procedure based on two well-known datasets. First, we used the Beatles dataset with

Dataset	Method	pairwise			boundary ($3s$)		
		F[%]	P[%]	R[%]	F[%]	P[%]	R[%]
BeatlesTUT	**proposed**	68.0	71.4	68.8	61.4	58.0	69.5
	[5]	60.8	61.5	64.6	N/A	N/A	N/A
	[13]	59.9	72.9	54.6	N/A	N/A	N/A
	SMGA (worst)	65.8	70.9	65.9	69.6	68.1	72.9
	SMGA (best)	71.8	65.1	80.0	75.3	73.4	79.1
Mazurka49-Rub	**proposed**	72.3	70.1	78.7	60.6	66.3	60.5
Mazurka49-Coh	**proposed**	70.0	69.3	74.2	62.7	65.3	65.9
Mazurka49-Eza	**proposed**	71.4	69.0	77.4	64.4	70.7	64.1
Mazurka2792	SMGA (worst)	68.1	75.2	65.2	65.9	70.3	65.3
	SMGA (best)	71.9	75.8	71.6	69.2	72.4	69.5

Table 1: Evaluation results for various procedures, evaluation measures, and datasets, see text for a detailed explanation.

the TUT annotations [6] described in [13]. Second, we used three complete recordings (Rubinstein 1966, Cohen, Ezaki) taken from the 2792 recordings of the Mazurka dataset [7] with manually generated structure annotations. Using standard precision (P), recall (R) and F-measure (F) for labeled pairs of frames as well as for segment boundaries (with 3 seconds tolerance), we compared our approach to [5, 13] as well as to the best performing MIREX2012 method [8] denoted by SMGA which is based on an extension of [17]. For SMGA, the results are reported for two different parameter settings corresponding to best and the worst performing setting, respectively.

Table 1 shows the results. Note that we have applied a similar NMF-based structuring algorithm as in [5], however applied to our converted matrix $\mathcal{S}^{\mathrm{Block}}$. This leads to substantial improvements compared to [5] on the Beatles dataset considering pairwise P/R/F-values. Also compared to the SMGA results, we are at least in the same range. As for the segment boundaries, however, we are worse. This is by no surprise since our approach is a purely frame-based procedure, whereas SMGA is based on a segment boundary detection step. Similar results hold for the Mazurka dataset, where SMGA has been evaluated on all 2792 recordings, which include the three versions used in our experiments. Our procedure yields for all three pianists pairwise P/R/F-values that are in the same range as the ones reported for SMGA. Again we want to emphasize that the quantitative results are not in the focus of this paper, but should only indicate the overall behavior of our conversion procedure.

4. CONCLUSIONS

In this paper, we introduced a novel method for converting SSMs with path structures into SSMs with block structure based on eigenvalue decompositions. As main technical contribution, we discussed how certain path structures translate into characteristic properties of the eigenvectors. Furthermore, as an application of our conversion, we showed how a homogeneity-based structure analysis procedure can be applied to the converted path matrix to facilitate repetition-based structure analysis. We hope that our contribution is interesting not only from a conceptual point of view, but may also open up novel ways for fusing different segmentation principles at an early stage of a structure processing pipeline. In particular, it seems promising to directly combine block-like SSMs (reflecting homogeneous musical properties) with converted path-like SSMs (reflecting repetitive musical properties), which can then be handled using the same algorithmic pipeline.

Acknowledgments: This work has been supported by the German Research Foundation (DFG CL 64/8-1, DFG MU 2682/5-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer IIS.

[6] http://www.cs.tut.fi/sgn/arg/paulus/structure.html

[7] http://www.mazurka.org.uk

[8] http://nema.lis.illinois.edu/nema_out/mirex2012/results/struct/mrx09/

5. REFERENCES

[1] Roger B. Dannenberg and Masataka Goto. Music structure analysis from acoustic signals. In David Havelock, Sonoko Kuwano, and Michael Vorländer, editors, *Handbook of Signal Processing in Acoustics*, volume 1, pages 305–331. Springer, New York, NY, USA, 2008.

[2] Shlomo Dubnov and Ted Apel. Audio segmentation by singular value clustering. Proc. ICMC, 2004.

[3] Emilia Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, UPF Barcelona, 2006.

[4] Florian Kaiser, Marina Georgia Arvanitidou, and Thomas Sikora. Audio similarity matrices enhancement in an image processing framework. Proc. CBMI, Madrid, Spain, 2011.

[5] Florian Kaiser and Thomas Sikora. Music structure discovery in popular music using non-negative matrix factorization. Proc. ISMIR, pages 429–434, Utrecht, The Netherlands, 2010.

[6] Jerry L. Kazdan. A Tridiagonal Matrix, http://hans.math.upenn.edu/~kazdan/AMCS602/tridiag-short.pdf, Retrieved 11.03.2013.

[7] Jingu Kim and Haesun Park. Toward faster nonnegative matrix factorization: A new algorithm and comparisons. Proc. ICDM, pages 353–362, Pisa, Italy, 2008.

[8] Mark Levy and Mark Sandler. Structural segmentation of musical audio by constrained clustering. IEEE TASLP, 16(2):318–326, 2008.

[9] Hanna Lukashevich. Towards quantitative measures of evaluating song segmentation. Proc. ISMIR, pages 375–380, Philadelphia, USA, 2008.

[10] Meinard Müller and Sebastian Ewert. Towards timbre-invariant audio features for harmony-based music. IEEE TASLP, 18(3):649–662, 2010.

[11] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. Proc. ISMIR, pages 215–220, Miami, FL, USA, 2011.

[12] Meinard Müller and Frank Kurth. Towards structural analysis of audio recordings in the presence of musical variations. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007.

[13] Jouni Paulus and Anssi P. Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. IEEE TASLP, 17(6):1159–1170, 2009.

[14] Jouni Paulus, Meinard Müller, and Anssi P. Klapuri. Audio-based music structure analysis. Proc. ISMIR, pages 625–636, Utrecht, The Netherlands, 2010.

[15] Geoffroy Peeters. Deriving musical structure from signal analysis for music audio summary generation: "sequence" and "state" approach. Proc. CMMR, Vol. 2771, LNCS, pages 143–166. Springer, 2004.

[16] Geoffroy Peeters. Music structure dicovery: Measurig the "stateness" of times. ISMIR: Late Breaking Session, 2011.

[17] Joan Serrà, Meinard Müller, Peter Grosche, and Josep Lluis Arcos. Unsupervised detection of music boundaries by time series structure features. Proc. AAAI International Conference on Artificial Intelligence, Toronto, Canada, 2012.

[18] 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. Proc. ISMIR, pages 555–560, Miami, FL, USA, 2011.

THE AUDIO EFFECTS ONTOLOGY

Thomas Wilmering, György Fazekas, Mark B. Sandler
Centre for Digital Music (C4DM)
Queen Mary University of London
thomas.wilmering@eecs.qmul.ac.uk

ABSTRACT

In this paper we present the Audio Effects Ontology for the ontological representation of audio effects in music production workflows. Designed as an extension to the Studio Ontology, its aim is to provide a framework for the detailed description and sharing of information about audio effects, their implementations, and how they are applied in real-world production scenarios. The ontology enables capturing and structuring data about the use of audio effects and thus facilitates reproducibility of audio effect application, as well as the detailed analysis of music production practices. Furthermore, the ontology may inform the creation of metadata standards for adaptive audio effects that map high-level semantic descriptors to control parameter values. The ontology is using Semantic Web technologies that enable knowledge representation and sharing, and is based on modular ontology design methodologies. It is evaluated by examining how it fulfils requirements in a number of production and retrieval use cases.

1. INTRODUCTION

The development of tools and services for the realisation of the Semantic Web has been a very active field of research in recent years, with a strong focus on linking existing data. In the field of music information management, Semantic Web technologies may facilitate searching and browsing, and help to reveal relationships with data from other domains. At the same time, many algorithms have been developed to extract low and high-level features, which enable the user to analyse music and audio in detail. The use of semantics in the process of music production however is still a relatively new field of research. With computer systems and music processing applications becoming increasingly powerful and complex in their underlying structure, semantics can help musicians and producers in decision processes, and provide more natural interactions with the systems.

Herrera and Serra [9] stressed the potential of semantic sound descriptors for the development of new audio applications in their work using MPEG-7 descriptors. They asserted that "there are [...] sound content-based processing applications waiting to be developed once we have a robust set of descriptors and structures for putting them into relation and for expressing semantic concerns about sound." We argue that Semantic Web technologies, such as Semantic Web ontologies and RDF are a superior choice for the representation of metadata in audio production, because they allow for a more flexible and extensible representation of this heterogeneous information domain. Beside, as de-facto standards of the future Web, they allow for sharing and linking structured information across different domains. Ontology-driven knowledge management in music production has also been discussed in [2, 7]. Within this field, the main focus of our study is the representation of information about audio effects.

Audio effects play an integral part in modern music production. They modify an input signal and may be applied in order to enhance the perceived quality of a sound or to make more drastic changes to it in the composition process. Employing music information retrieval (MIR) and Semantic Web technologies specifically for the control of audio effects has the potential of representing a significant step in their evolution. This work therefore has a good potential to address a phenomenon described by Voorvelt [17]: "in the context of popular music production, the equipment in use generally trails the latest technological developments." Detailed descriptions of the use of audio effects in a music production project can additionally facilitate the reproduction of workflows, and add an additional layer of depth to MIR. For instance, the ontology can help answering queries such as: *Which effects have been used in a music production project and what are the parameter settings? Which effect implementations are available that suit the needs for a specific workflow?*

The Audio Effects Ontology presented in this paper is designed as an extension to the Music Ontology [14] and Studio Ontology [7], as well as our previous work detailed in [18]. First, we briefly discuss the Semantic Web Technologies underlying this work. Then we introduce the Audio Effects Ontology and provide an overview of its design and purpose. We discuss some applications in the domain of music production and music information retrieval, and finally outline directions of future work.

2. SEMANTIC WEB TECHNOLOGIES

The Semantic Web aims to bring intelligence to the Web by allowing machines to reason about Web content. With

the proliferation of audio content on the Web, representing information about audio and its production is just as important as processing and linking text documents. The first step towards this goal is to represent information in a machine interpretable format.

A stack of technologies have been proposed for building the Semantic Web. The Resource Description Framework (RDF) is a data model for describing statements using *subject, predicate, object* triples. These form an RDF graph when combined. When the elements of RDF statements are identified by uniform resource identifiers (URI), we obtain an interlinked, globally distributed "database", the Web of Linked Data. However, to enable querying or reasoning over linked data, we need languages that describe the shared meaning of RDF graphs, in other words, represent knowledge about the entities described in data sets.

2.1 Ontologies and knowledge representation

Ontology languages like the RDF Schema language and the Web Ontology Language (OWL) [1] allow for characterising entities in terms of their relationships. They describe a shared conceptualisation of a world [8] comprised of individuals, classes and relations, with formal semantics that allow automated reasoning over RDF data expressed using an ontology. RDF Schema allows for defining simple hierarchies of classes and properties with a set of constraints over their use, but without adequate logical grounding. OWL refines this model by adding tools for representing domain knowledge more precisely. For instance, we can characterise properties in terms of transitivity, symmetry or reflexivity. OWL constructs directly correspond to Description Logics (DLs), a family of logic-based knowledge representation languages which in turn are based first-order logic [10].

3. THE STUDIO ONTOLOGY

The Studio Ontology is an OWL ontology for capturing the nuances of record production by providing an explicit, application and situation independent conceptualisation of the studio environment [7]. It is presented as a modular framework of ontologies, which in turn are built on the Music Ontology framework [14] and its components. It uses its core elements that allow for the representation of time-based events (Event and Timeline ontologies), and the workflow of music production in an editorial context subsumed under broader terms defined by the Functional Requirements for Bibliographic Records (FRBR) [13].

The Music Ontology allows for describing the music production workflow from composition to delivery, however, it lacks some concepts to do so in sufficient detail. The Studio Ontology provides some of the necessary extensions that form the foundation for a comprehensive representation of audio effects, and their application in music production. Here, we outline only those of its features and components which make it suitable as basis for our work.

- **Foundational components**: The Studio Ontology allows for characterising and describing the application of technological artefacts (devices) in music production. The Device Ontology provides a fundamental device and device decomposition model, and entities for representing device states, such as variable device parameters at different levels of granularity.
- **Complex device descriptions**: The ontology provides a model for describing complex devices such as signal processing tools and their interconnections. It includes a four-layered abstract model of these devices resembling the FRBR model.
- **Core components**: The ontology provides for describing recording studios on the editorial level (e.g. personnel and available equipment). It also provides for describing signal processing workflows in the studio using a parallel event and signal flow utilising music production tools.
- **Domain specific extensions**: The Studio Ontology supports the provision of domain specific extensions. It also provides some extensions for describing audio recording (e.g. microphones), mixing, editing and a core model for describing audio effects.

The application of audio effects to signals can be described using the concept studio:Transform defined by the Studio Ontology. This concept represents an event that takes a signal as a factor, and produces a transformed signal. This concept may be subsumed in more specific effect ontologies. The Studio Ontology sets aside the problem of defining specific audio effects, their classifications, parameters and their application specific descriptions. The Audio Effects Ontology fills this gap.

4. THE AUDIO EFFECTS ONTOLOGY

The aim of the Audio Effects Ontology is the representation of knowledge concerning audio effect implementations and their application in the music production studio. For instance, music software such as digital audio workstations (DAW) and digital effects implementations, may support the audio engineer by producing and reusing knowledge that is represented using the concepts and properties defined by the ontology.

4.1 The Core Ontology

The core parts of the Audio Effects Ontology define concepts and properties for the description of audio effect implementations and how they are applied within the production process. This facilitates the incorporation of data about the application of audio effects in online catalogues or content-based music recommendation systems, and allows for using an effects database in Semantic Web applications at large. For instance, this facilitates the retrieval of songs characterised by certain effects or effect types used in a production, and the reproduction of workflows.

Rather than signal processing devices, audio effects in our proposed ontology are conceptualised as physical and acoustical phenomena, that are represented on the same conceptual layer as the abstract *Work* entity in the FRBR

model [13] of intellectual works. An effect is represented by the OWL class *afx:Fx*. Furthermore, the *Fx* class can be linked to effect types (see Section 4.2), thus adding meaning to the audio effect that may not be given solely by an implementation's given name. A separate class serves the purpose of describing signal processing devices, such as software implementations or hardware effect units.

The description of audio transformations in music production is another purpose of the ontology. To enable this functionality, the Audio Effects Ontology defines concepts for describing the application of effects to a signal. These concepts integrate seamlessly with the Studio Ontology that already provides a class for transformations (see Figure 1).

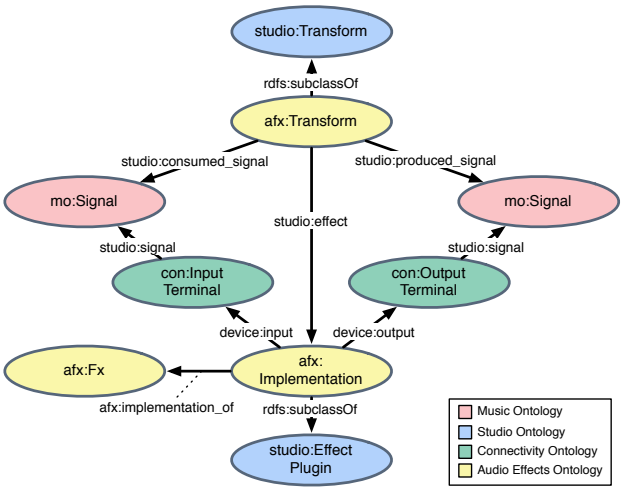


Figure 1. Transformation of an audio signal described using the Studio Ontology and Audio Effects Ontology.

A *transform* can be linked to an effect implementation using the *studio:effect* property.Using this mechanism we can express details about the effect implementation involved in a transformation and the device state at the time of the transformation. The ontology provides additional concepts for the description of implementation attributes, such as the plugin format, operating system, parameters and parameter settings (see Table 1). Instead of being conceptualised as static individuals of the respective classes, operating systems and plugin formats are conceptualised as subclasses, which enables a more detailed description of the execution context, for instance, by specifying a particular version number of an effect plugin. This implementation class may also act as the connection to the *afx:Fx* class specifying the effect type with the property *afx:implementation_of*. Associating events on an audio signal timeline — using appropriate Music Ontology terms — to a particular transform and its parameters, it is possible to state where a certain effect has been applied during the course of a track.

Finally, it is a common practice to automate effect parameters in music production, i.e. the parameters of a given effect may change over time. Modern DAWs are able to store the automation data for this purpose. In order to represent changing effect parameter values the Audio Effects Ontology provides the class *afx:State* which represents a similar concept as the *device:State* class, proposed in the context of *consolidated reification* in [6]. It conceptualises variable attributes and relationships of an audio effect. The *afx:State* class is a subclass of *event:Event* in the Event Ontology [1]. This allows describing a region on a signal timeline, during which a certain parameter setting is true. The region is defined by an *entry* and *exit* point. These are subclasses of *tl:Instant* of the Timeline Ontology [2].

4.2 Audio Effects Classification

In order to produce metadata describing the workflow and the elements of an audio production, it is beneficial to describe which specific audio effect implementation has been used for a given transformation, and also to specify the effect type. This facilitates the comparison of workflows independently from the tools that have been used in the studio. The conceptualisation of effect types for instance enables the search for similarities in the use of audio effects in a database of music production data. While the classification of audio effects has several applications in music production, the heterogeneity of possible taxonomies, as well as the many viable points of view for organising effects present research problems that are not easily solved. Creating extensible ontologies provide a possible solution to this problem. Musicians and music producers have a large number of digital audio effects at their disposal, while over 70 types of effects have been identified in academic research [21]. There are different approaches to the task of audio effects classification depending on a variety of factors. For instance, we may group effects by their perceptual attributes or classify them by their underlying signal processing implementations. The best classification depends on the intended use. A developer for example would probably want to emphasise signal processing techniques, whereas a musician would prefer to classify effects by their perceptual qualities. An example of inter-disciplinary effect classification has been proposed in [15], as part of an effort to facilitate communication and collaborations between DSP programmers, sound engineers, composers, performers and musicologists. To address this issue, we incorporated several linked classification systems subsumed under the concept *afx:Fx* in our ontology. These are based on different criteria, including technical aspects as well as perceptual attributes. As a result, an MIR system using the ontology may answer questions such as: *Which audio effects affect the timing of the audio material? Which productions used delay-based audio effects?*

4.3 Effect Parameters

Recognising the fact that not all audio effect implementations adhere to parameter naming conventions, we extend our ontology with the Parameter Ontology module. We conceptualise effect parameters in such a way that we can assign a parameter type to a parameter that is linked to an audio effect implementation. We distinguish between two types of parameters: numerical parameters and indexed pa-

[1] http://motools.sf.net/event/event.html
[2] http://motools.sf.net/timeline/timeline.html

concept	property	range	some subclasses/individuals
Product name	dc:title	*literal* (xsd:string)	-
FX format	available_as	Format	Vst, Au, Lv2, Rtas
Operating system	os	Os	Windows, MacOS, Linux
FX type	implementation_of	Fx	class1:Chorus, class2:Bandpass
FX technique	technique	Technique	SchroederMoorer, PhaseVocoder
FX parameters	parameter	Parameter	NumParameter, IndexedParameter
FX preset	preset	Preset	-
Audio inputs	audio_inputs	*literal* (xsd:int)	-

Table 1. Some of the concepts and properties for the description of a digital audio effect implementation.

rameters. The former is set by numerical values, while the latter consists of a list of string values. For instance, a parameter may be used to specify a filter type or a waveshape for an oscillator. We may want to query for effects that have a specific type of parameter (e.g. the delay time or attack time). Consistent parameter names, which are not necessarily given, are a prerequisite for efficient comparison. The Parameter Ontology solves this by providing concepts for parameter types. Instead of simply labelling a parameter with a literal stating its given name, linking parameters to conceptualised parameter types facilitates retrieving this information independently from the actual given parameter names. Furthermore, by specifying the unit (for the delay time this may be milliseconds or seconds) we can compare and transfer settings across different implementations. To achieve this we use concepts of the Quantities, Units, Dimensions and Data Types (QUDT) ontology [3]. Figure 2 shows a description for a delay time parameter.

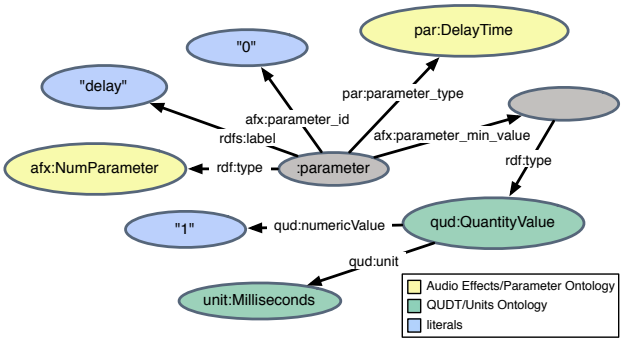


Figure 2. Partial description for a delay time parameter as it appears in an effect implementation.

4.4 Provenance

Provenance information describes entities, activities, and people involved in producing data. It enables software agents to track changes to data, thus ensuring a level of transparency and trust by providing information about the sources of data items. For instance, provenance information can consist of a statement about who created a particular resource. A detailed review of provenance ontologies, both general and discipline-specific, is provided in [5].

Digital audio effects alter audio data and consequentially the audio features associated with it. The studio ontology already provides mechanisms for describing the author, for instance an audio engineer, that was involved in the creation of a music production. Moreover, relating a *transform* with an effect implementation documents which software device has been used in the process. We introduce additional provenance properties in the Audio Effects Ontology for describing timed audio features that have been produced or altered by the application of an audio effect. Since the Studio and Audio Effects ontologies are developed in the context of future intelligent audio workstations that produce detailed metadata about the audio material and workflows in music production, the inclusion of the provenance properties facilitates adding effect-specific metadata to annotations of audio signals. For describing provenance information in the DAFX ontology, we introduce subproperties subsuming properties of the The Open Provenance Model Vocabulary (OPMV), that is based on the Open Provenance Model (OPM) [12]. In OPM, an *artefact* is defined as an "immutable piece of state" which may refer to an actual physical object or a digital representation. A *process* is the action that creates artefacts, be it by acting on an existing artefact or by being caused by one. *Agent* describes an entity involved in a process by enabling or influencing its execution. Edges denote causal dependencies between its source (the effect) and its destination (the cause). OPM can be used in combination with terms from the Dublin Core specification which on its own we found to be insufficient for our requirements.

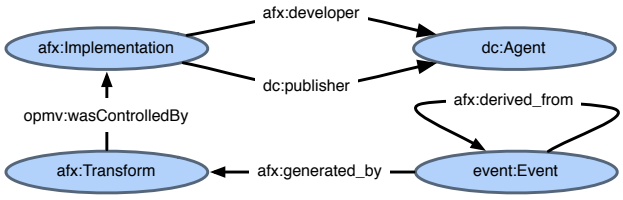


Figure 3. Provenance properties in the Audio Effects Ontology.

Using these properties of the Audio Effects Ontology (Figure 3), we can represent provenance information about audio features. For instance, an echo effect produces additional note onsets, since it adds the delayed signal to the original. We can express that such an event (artefact) has been *generated by* a given transformaion (process), and that the transformation *was controlled by* an audio effect implementation, the echo effect (agent). We can express that the existence of an audio feature produced by the transformation is dependent on a previously existing feature. For instance, a delayed note onset event may be *derived from* an onset in the original audio material prior to trans-

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

formation.

5. APPLICATIONS

5.1 Creation of Music Production Studio Databases for Information Retrieval

Designed as an extension of the Studio Ontology, knowledge represented with the Audio Effects Ontology can be seamlessly integrated into a database using the Studio Ontology framework. This information may include details about audio effects, such as the type of effects, their characteristics and parameter configuration of specific instances of effects applied in a music production project. Including detailed information about the application of audio effects may facilitate retrieval for various purposes for music production. This also facilitates reproducibility of workflows concerning the application of audio effects. Furthermore, by employing techniques similar to those applied in music recommendation systems based on Semantic Web technologies [4] [11], the Audio Effects Ontology provides a framework for the retrieval of audio effects given a set of specified criteria. These criteria can be technical aspects, as well editorial information such as the developers or vendors involved in audio effect implementations.

5.2 Publishing effect data on the Semantic Web

The Audio Effects Ontology is capable of refining the retrieval of songs based on production procedures as proposed in the context of the Studio Ontology framework. Data about audio effects may be published as Linked Data resources. This allows for the creation of an audio effects database on the Semantic Web, and facilitates the incorporation of data about the application of audio effects in music productions in online catalogues and content-based music recommendation systems. For example, this enables the retrieval of songs characterised by certain types of effects, or the actual effect used in a track. Assuming a large enough database of production data, this also allows for musicological research with regard to trends in the application of audio effects. Furthermore, plugin presets may be shared on the Semantic Web and retrieved by users and agents helping students or the work of professional engineers.

5.3 Adaptive Audio Effects

Previous implementations of adaptive audio effects [16] that map high-level features stored in a database to control parameters either use proprietary non-standardised formats, or MPEG-7 descriptors for the representation of audio features (e.g. [3]). However, the majority of metadata standards only specify the syntax of documents, while the semantics remain implicit and hardcoded in procedural software. Using Semantic Web technologies on the other hand, provide a uniform way of encoding and linking information. The Audio Effects Ontology provides the means for integrating adaptive audio effects seamlessly into a music production system that supports RDF based knowledge representation and retrieval. Examples of this new class of audio effects are given in [19, 20].

6. CASE STUDY: RETRIEVING AUDIO EFFECT SETTINGS BY QUERYING METADATA

Audio effect play a crucial role in creating the right "sound" of a track in most contemporary musical genres. Reproducing the application of effects including their exact parameter settings is therefore a very important use case.

The Audio Effects Ontology covers the necessary concepts to represent the information about an effect transformation in such a way that it is possible to query for a given effect and its parameter settings at a given temporal location relative to the audio signal. In the following we show an example of how we can query data represented with the Audio Effects Ontology to retrieve information about the application of audio effects in a music production project, where production metadata is available using our ontology framework. We may want to investigate an audio effect found in an annotated music production where one minute into the song an echo effect has been applied to the guitar. Using the appropriate SPARQL [4] queries we are able to retrieve the necessary information in order to identify comparable effect plugins present in our studio setup. First, we query the project database for the parameter types, settings and units of an echo effect applied on the specified track at the specified time instant (Listing 1). We assume the timeline for the guitar track as *:guitarTimeline* starting at the beginning of the project.

```
SELECT ?parameter_type ?value ?unit
WHERE {
 ?transform a afx:Transform ;
  studio:effect [ a afx:Implementation ;
    implementation_of ?class1:Echo ;
    afx:state ?state ] .
 ?event a event:Event ;
  event:time [ a tl; instant ;
    tl:timeline :guitarTimeline ;
    tl:at "60.0s"^^xsd:duration ] .
 ?state afx:entry ?event ;
  afx:parameter [ a afx:Parameter ;
    par:parameter_type ?parameter_type ;
    qud:value [ qudt:numericValue ?value ]
        ;
    afx:unit ?unit ] . }
```

Listing 1. Query retrieving effect settings.

The parameter unit specifications can form the basis of the conversion of settings between implementations having parameters of the same type with different units. This may be useful in case the implementations used originally are not available, and we wish to approximate the transformations with effects at our disposal in our studio. In a second step we query the database describing our studio facility for existing echo effect implementations having parameters of the same type (Listing 2). The query retrieves all the echo effect implementations available in our studio setup that have parameters of the same type as the one used in the production project in question. Since we also know the respective units for the parameter values it is possible to transfer the settings for the retrieved effect implementations. The information can act as a starting point for the

[4] SPARQL is a recursive acronym for SPARQL Protocol and RDF Query Language.

```
SELECT ?publisher ?fx_name ?time_name
 ?time_unit ?parameter_name ?parameter_unit
WHERE {
 :ourStudio studio:equipment ?device ;
 ?device a afx:Implementation ;
  dc:publisher [ fc:name ?publisher ] ;
  dc:title ?fx_name ;
  implementation_of [ class1:Echo ] ;
  has_parameter [ a afx:Parameter ;
   afx:parameter_type par:DelayTime ;
   rdfs:label ?time_name ;
   afx:unit ?time_unit ] ,
   [ a afx:Parameter ;
   afx:parameter_type par:LowpassFilter ;
   rdfs:label ?lowpass_name ;
   afx:unit ?lowpass_unit ] . }
```

Listing 2. Query retrieving effect implementations.

approximation of transformations performed by one implementation of a given effect type with another implementation of the same effect family.

7. CONCLUSIONS AND FUTURE WORK

We presented a Semantic Web Ontology covering the domain of audio effects and their implementations that is designed as an extension to the Studio Ontology framework. Using the ontology it is possible to create detailed metadata about the application of effects in music production projects and to classify and describe audio effect implementations. The applications of the Audio Effects Ontology range from adaptive audio effects using high-level semantic metadata, content-aware music production tools, and searchable audio effect databases. We have shown that querying RDF databases storing information about projects, studio equipment and available effect implementations enables access to detailed information about workflows, and facilitates their reproduction. Moreover, it allows for the analysis and comparison of musical works with regards to the use of audio effects.

Future work includes the development of software applications that support the Studio Ontology framework, such as content-aware audio production tools that automatically retrieve information and annotate multitrack projects automatically.

8. REFERENCES

[1] G. Antoniou and F. van Harmelen. Web ontology language: OWL. *in S. Staab and R. Studer (eds.), Handbook on Ontologies, International Handbooks on Information Systems,* Springer-Verlag Berlin Heidelberg, pages 91–110, 2009.

[2] K. Barkati, A. Bonardi, A. Vincent, and F. Rousseaux. GAMELAN: A knowledge management approach for digital audio production workflow. In E. Mercier-Laurent, editor, *Proceedings of Artificial intelligence for Knowledge Management Workshop (AI4KM) of ECAI*, 2012.

[3] O. Celma, E. Gómez, J. Janer, F. Gouyon, P. Herrera, and D. Garcia. Tools for content-based retrieval and transformation of audio using mpeg-7: the spoffline and the mdtools. *presented at the AES 25th International Conference, London, UK*, June 2004.

[4] Ò. Celma and X. Serra. FOAFing the music: Bridging the semantic gap in music recommendation. *Lecture Notes in Computer Science*, 4273, 2006.

[5] L. Ding, J. Bao, J. Michaelis, J. Zhao, and D. L. McGuinness. Reflections on provenance ontology encodings. *Proceedings of the 3rd International Provenance and Annotation Workshop (IPAW), Troy, New York, USA*, June 2010.

[6] G. Fazekas. *Semantic Audio Analysis - Utilities and Applications*. PhD thesis, Queen Mary University of London, 2012.

[7] G. Fazekas and M. B. Sandler. The studio ontology framework. *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, 2011.

[8] M. R. Genesereth and N. J. Nilsson. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, San Mateo, CA, USA., 1987.

[9] P. Herrera and X. Serra. Audio descriptors and descriptor schemes in the context of mpeg-7. *Proceedings of the International Computer Music Conference, Beijing, China*, 1999.

[10] I. Horrocks. Ontologies and the Semantic Web. *Communications of the ACM*, Vol. 51(12):pp. 58–67., 2008.

[11] K. Jacobson, G. Fazekas, Y. Raimond, and M. Smethurst. Music and the web of linked data. *Tutorial presented at the 10th International Society for Music Information Retrieval Conference (ISMIR), Kobe, Japan*, 2009.

[12] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, and J. V. den Bussche. The open provenance model core specification (v1.1). *Future Generation Computer Systems*, Preprint, July 2010.

[13] M.-F. Plassard, editor. *Functional Requirements For Bibliographic Records : final report / IFLA Study Group on the Functional Requirements for Bibliographic Records*, volume 19. K.G. Saur, 1998.

[14] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson. The music ontology. *Proceedings of the International Conference on Music Information Retrieval*, 2007.

[15] V. Verfaille, C. Guastavino, and C. Traube. An interdisciplinary approach to audio effect classification. *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-06), Montreal, Canada*, September 2006.

[16] V. Verfaille, U. Zölzer, and D. Arfib. Adaptive digital audio effects (A-DAFx): A new class of sound transformations. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5), 2006.

[17] M. Voorvelt. New sounds, old technology. *Organised Sound*, 5(2):67–73, 2000.

[18] T. Wilmering, G. Fazekas, and M. B. Sandler. Towards ontological representations of digital audio effects. *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx-11), Paris, France*, 2011.

[19] T. Wilmering, G. Fazekas, and M. B. Sandler. High level semantic metadata for the control of multitrack adaptive audio effects. *presented at the 133rd Convention of the AES, San Francisco, USA*, 2012.

[20] T. Wilmering and M. Sandler. RDFx: Audio effects utilising musical metadata. *presented at the 4th IEEE International Conference on Semantic Computing (IEEE ICSC 2010), Pittsburgh, PA, USA*, 2010.

[21] U. Zölzer. *DAFX - Digital Audio Effects*. J. Wiley & Sons, 2nd edition, 2011.

EXPLORATION OF MUSIC EMOTION RECOGNITION BASED ON MIDI

Yi Lin, Xiaoou Chen and Deshun Yang
Institute of Computer Science & Technology, Peking University
lin.yi.tz@gmail.com
{chenxiaoou, yangdeshun}@pku.edu.cn

ABSTRACT

Audio and lyric features are commonly considered in the research of music emotion recognition, whereas MIDI features are rarely used. Some research revealed that among the features employed in music emotion recognition, lyric has the best performance on valence, MIDI takes the second place, and audio is the worst. However, lyric cannot be found in some music types, such as instrumental music. In this case, MIDI features can be considered as a choice for music emotion recognition on valence dimension.

In this presented work, we systematically explored the effect and value of using MIDI features for music emotion recognition. Emotion recognition was treated as a regression problem in this paper. We also discussed the emotion regression performance of three aspects of music in terms of edited MIDI: chorus, melody, and accompaniment. We found that the MIDI features performed better than audio features on valence. And under the realistic conditions, converted MIDI performed better than edited MIDI on valence. We found that melody was more important to valence regression than accompaniment, which was in contrary to arousal. We also found that the chorus part of an edited MIDI might contain as sufficient information as the entire edited MIDI for valence regression.

1. INTRODUCTION AND RELATED WORKS

Music is a natural carrier to express and convey emotion. Some emotion models have been developed to describe emotion state. Russell's two-dimensional valence-arousal (V-A) model [4] consisted of two independent dimension of valence and arousal. Valence stands for appraisal of polarity and arousal stands for the intensity of emotion. Mehrabian [10] extended this approach and developed a three-dimensional pleasure-arousal-dominance (PAD) model, where dimension P distinguishes the positive-negative quality of emotion, dimension A refers to the intensity of physical activity and mental alertness, and dimension D refers to the degree of control. In this paper, we focused on Russell's V-A model, especially the valence (V) dimension, which corresponds to pleasure (P) dimension in PAD model. Several types of feature have been developed to represent a piece of music. Audio and lyric features are commonly considered in the research of music emotion recognition, whereas MIDI features are rarely used [1]. Emotion recognition can be viewed as a multiclass-multilabel classification or regression problem [1]. In this paper emotion recognition was treated as a regression problem.

This paper focused on music emotion recognition with MIDI features, which can be extracted directly from MIDI files. Unlike audio data, MIDI is a kind of the electronic score. Symbolic representations of music (such as key, pitch, tempo, etc.), which are high-level musicological symbolic features reflecting music concepts, can be easily extracted and calculated from MIDI by toolkits such as jSymbolic [7]. Therefore, MIDI features may be more effective on music emotion regression.

Oliveira and Cardoso [3] constructed a dataset of 96 western tonal music (film music) pieces, which lasted between 20 seconds to 1 minute. These pieces were in MIDI format and each piece might express only one type of affective content. 80 listeners were asked to label these musical pieces with affective labels on valence and arousal, respectively. Both musicological symbolic features (e.g., tempo, note duration, note density, etc) and acoustical features (such as MFCCs) were extracted. After feature selection, they performed SVM regression and received correlation coefficient of 81.21% on valence and 84.14% on arousal from 8-fold cross validation experiment. Then Oliveira and Cardoso [3] selected the most important features that were identified separately during feature selection results for valence and arousal and performed the 8-fold cross validation of SVM regression again. The most important features selected were all symbolic features. The performance evaluated in terms of correlation coefficient reached 71.5% on valence and 79.14% on arousal. Oliveira and Cardoso's work demonstrated that MIDI is effective on music emotion recognition.

Oliveira and Cardoso's work [3] only focused on MIDI, whilst Guan et al. [2] compared music emotion recognition performance among audio, lyric, and MIDI features. They presented an AdaBoost approach on 1687 Chinese songs. A wave file and a lyric file were collected for each song and a MIDI files was converted from the

wave file. Audio, lyric, and MIDI features were extracted separately. Guan et al. [2] applied their AdaBoost.RM approach on a multi-modal feature set in which audio, lyric and MIDI features were combined together. The performance of the experiment was 74.2% on valence in terms of correlation coefficient. They also applied regression to audio, lyric, and MIDI features, respectively. The results showed that the regressor built using lyric features yielded the best performance on valence; the performance was 62.3% in terms of correlation coefficient, whereas the regressor built using audio features had the poorest performance of 47.3% and the regressor built using MIDI features was in between, 54.1%.

Based on findings from the related works, there were two aspects of views towards MIDI. Oliveira and Cardoso [3] regarded MIDI as an object on which the emotion value is labeled. Although they received a good result on emotion regression, it was very difficult to find a satisfactorily-composed MIDI file for a song. A satisfactorily composed MIDI file should be a MIDI that sounds exactly the same to the original song when listened to. Instead Guan et al. [2] regarded MIDI as an intermediate representation of music audio data. They firstly converted audio files into MIDI files and then extracted MIDI features from the MIDI files. This was a much easier way for obtaining example MIDI files, as there were large amount of audio files available on the Internet.

In our work, we took Guan et al.'s [2] view and regarded MIDI as an intermediate representation of music audio data. We provided a systematic exploration of the effect and value of using MIDI features for music emotion recognition. Two types of MIDI files were. One type of MIDI files was converted from audio files, e.g., mp3, wav, etc. MIDI files obtained in this way were referred as converted MIDI. The other type of MIDI files was composed by musicians and composers via a score editing tool such as Guitar pro [5]. MIDI files obtained in this way were referred as edited MIDI. Oliveira and Cardoso [3] only considered edited MIDI and Guan et al. [2] only considered converted MIDI, whereas we considered both in our work.

Our work consisted of two parts. In part one we compared the music emotion regression performance of audio, lyric, and MIDI features. We carried out emotion regression with audio, lyric, and MIDI features separately rather than combined them together. We also compared the music emotion regression performance between converted MIDI and edited MIDI; this work was not included in the paper of Guan et al [2]. Moreover, we played the 653 edited MIDI files back acoustically, recorded that, and converted that back to MIDI automatically to investigate what helped to predict emotion. In part two we investigated the edited MIDI from three aspects of music: chorus, melody, and accompaniment.

The paper is organized as follows. Section 2 presents the datasets and features. Section 3 reports the experiments and results. Section 4 provides the analysis of the experiment results. Section 5 concludes the key findings.

2. DATASETS AND FEATURES

2.1 Datasets

We used Guan et al.'s 2500-Chinese-song list, where each song was annotated with a PAD label [2]. Datasets were constructed according to the 2500-Chinese-song list. All these Chinese songs were composed based on Western 12-tone equal temperament.

In our work, 7 datasets were constructed as described in the following section (including four sets of edited MIDI files, one set of audio files, one set of lyric files and one set of converted MIDI files). Among the four types of data (edited MIDI, converted MIDI, audio, and lyric), the edited MIDI data was the most difficult to be collected among the four kinds of data: edited MIDI, converted MIDI, audio, and lyric. Therefore we firstly constructed our edited MIDI datasets to determine how many songs could be included in our work. Then we constructed audio dataset and lyric dataset. Finally we constructed converted MIDI dataset in which converted MIDIs came from audios.

2.1.1 Edited MIDI

We applied two ways for collecting edited MIDIs. One way was to download edited MIDI files from Internet that were composed by musicians and composers. In our work, the notes of Edited MIDI files that collected in this way are entered directly into a sequencer program or notation software rather than being recorded directly from playing on a MIDI instrument. The other way was to download scores from Internet that were written by musicians and composers and then translated the scores into MIDI files via a score editing tool called Guitar pro [5].

The final edited MIDI dataset contained 653 MIDI pieces, which lasted between 15 seconds to 7 minutes. This dataset was a subset of the 2500 Chinese songs. Each MIDI in the dataset contains melody, accompaniment, and at least two different timbres, so that the MIDI could sound as similar as possible to the original songs when listened to. If multiple MIDI versions of a song were available, the one that was the closest to the original song when listened to would be retained. This dataset was referred as edit-MIDI.

We split each of the 653 MIDI pieces into two parts: melody and accompaniment. This work was carried out by manually extracting the tracks corresponding to melody and accompaniment, respectively, from one MIDI to create two new MIDIs. One represented melody and the other represented accompaniment. Thus two more datasets were constructed. The dataset, which contained 653 MIDI pieces corresponding to the melody part, was referred as edit-MIDI-melody; the other dataset, which

contained 653 MIDI pieces corresponding to the accompaniment part, was referred as edit-MIDI-accom.

Short versions of the 653 MIDIs in edit-MIDI are also collected. Each short MIDI was the chorus part of the corresponding music. Most of these pieces were used as cell phone ringtones, lasting between 15 seconds to 40 seconds. This dataset is referred as edit-MIDI-chorus.

To conclude, we built four datasets for edited MIDI: edit-MIDI, edit-MIDI-melody, edit-MIDI-accom, and edit-MIDI-chorus. It worth to mention that it has been very difficult to find for a piece of music a satisfactorily-composed edited MIDI that sounds exactly the same as the original music. The reasons are two-fold: on one hand, not all music had a MIDI; on the other hand, the composer might not be willing to share the MIDI file.

2.1.2 Audio and Lyric

For the 653 songs corresponding to the edited MIDI, we downloaded their wav audio files and lyric files, which constituted the audio dataset and lyric dataset, respectively.

2.1.3 Converted MIDI

Converted MIDI dataset contained 653 MIDI files converted from the 653 wav files. WIDI Recognition System Pro 4.0 [6] was used to help with the conversion. This dataset was referred as conv-MIDI.

There were several differences between edited MIDI and converted MIDI. Firstly, each converted MIDI pieces was of the same length as the original song, whereas the length of edited MIDI pieces varied. Secondly, there was only one timbre existed in converted MIDI; the timbre was set to be Instrumental Grand in WIDI. Edited MIDI, however, contained at least two different timbres. Finally, the converted MIDI was very different from the edited MIDI of the same song when listened to. The converted MIDI sounded like its pitches were in a mess and it was hard or even unable to distinguish how the melody went. Edited MIDI, however, sounded similar to or even the same as the original song.

2.2 Features

MIDI features were extracted from the MIDI files by jSymbolic [7]. For each MIDI file 112 types of MIDI features were extracted to compose a feature vector of 1022 dimensions. Audio features were extracted from the wave files by jAudio [8]. For each audio file 27 types of audio features were extracted to compose a feature vector of 112 dimensions for each song. After lyric files were pre-processed with traditional NLP tools including stop-words filtering and word segmentation, unigram features were extracted from the lyrics file to compose a feature vector of 13251 dimensions.

3. EXPERIMENTS AND RESULTS

Supervised feature selection was carried out on each of the feature sets to reduce the number of features and to improve the regression results. Correlation-based Feature Subset Selection with BestFirst was applied as its search method in our work [9].

Following results of regression experiments were obtained using 5-fold cross validation of SMO regression with RBFKernel [9]. SMO regression [11] implements support vector machine for regression and in our work we used Radial Basis Function (RBF) as its kernel function. The performance of regression was measured in terms of correlation coefficient (CF).

Russell's two-dimensional valence-arousal (V-A) model [4] was employed to measure music emotion.

3.1 Comparison among Lyric, Audio, Edited MIDI, and Converted MIDI

Firstly, regression was applied on edit-MIDI, conv-MIDI, audio, and lyric datasets separately to compare the MIDI features with commonly used audio and lyric features. The results are showed in Table 1.

Table 1 shows the regression performance of lyric, audio, conv-MIDI, and edit-MIDI on valence and arousal. It worth noting that conv-MIDI was found to perform better than audio and worse than lyric on valence. We also found that the performance of edit-MIDI was much worse than conv-MIDI.

Dataset	V	A
Lyric	78.81%	66.52%
Audio	54.45%	76.5%
Conv-MIDI	57.09%	74.96%
Edit-MIDI	46.42%	53.37%

Table 1. Performance of lyric, audio, conv-MIDI, and edit-MIDI.

To analyze the difference between edited MIDI and converted MIDI, we examined the remaining features on valence after feature selection. For conv-MIDI, there were 10 types of features remaining that consisted of 60 features. For edit-MIDI, there were 37 types of features remaining that consisted of 315 features. Among these types of remaining features, there were 7 types of features that were common to both converted MIDI and edited MIDI. The remaining feature types on valence are shown in Table 2.

In Table 2, row 1 listed the remaining feature types that belonged to converted MIDI only; row 3 listed the remaining feature types that belonged to edited MIDI only; row 2 listed the remaining feature types that belonged to both converted MIDI and edited MIDI. In row 1, it can be seen that there were only 3 types of features (e.g.

in row 1: Duration, Combined Strength of Two Strongest Rhythmic Pulses, and Rhythmic Variability) that were included in features of converted MIDI, but not in features of edited MIDI. In order to investigate the importance of these three types of features, we carried out the regression again without these 3 types of features on conv-MIDI. The performance of the experiment dropped 3.01% (from 57.09% to 54.07%) in terms of CF on valence.

conv-MIDI only	Duration Combined Strength of Two Strongest Rhythmic Pulses Rhythmic Variability
conv-MIDI & edit-MIDI	Chromatic Motion Strength of Strongest Rhythmic Pulse Variability of Note Duration Basic Pitch Histogram Beat Histogram Melodic Interval Histogram Time Prevalence of Pitched Instruments
edit-MIDI only	Amount of Apreggiation Average Melodic Interval Brass Fraction Changes of Meter Dominant Spread Glissando Prevalence Most Common Melodic Interval Prevalence Most Common Pitch Class Prevalence Note Density Number of Common Pitches Quality Fifths Pitch Histogram Melodic Interval Histogram Note Prevalence of Pitched Instruments ……

Table 2. The remaining feature groups on valence after feature selection.

3.2 Conversions of the Edited MIDI

In order to investigate whether it was the process of conversion from the original audio to MIDI that helped predicting emotion, we went through a two-step experiment. Firstly we played the 653 edited MIDI files back acoustically and recorded the sounds (MIDI-WAV dataset); and secondly, the files were converted back to MIDI automatically (MIDI-WAV-MIDI dataset). We then examined the changes of the regression performance on valence. The results are showed in Table 3.

Dataset	V
Edit-MIDI	46.42%
MIDI-WAV	26.04%
MIDI-WAV-MIDI	43.83%

Table 3. The performance of tow conversions of the edited MIDI.

Table 3 shows how the valence regression performed on the three datasets obtained from the two conversions. The performance was measured in terms of CF. Results in Table 3 revealed that the performance of MIDI-WAV-MIDI is lower than that of Edit-MIDI by 2.59% (from 46.42% to 43.83%).

3.3 Melody and Accompaniment of Edited MIDIs

The edited MIDI sounded similar to, or even the same as the original song. In our work each edited MIDI file was split into two parts: melody and accompaniment. This allowed us to measure the performance of these two parts on music emotion regression separately. However, we were not able to experiment them on converted MIDI, because melody or accompaniment could not be extracted from converted MIDI.

The experiment results are showed in Table 4.

Dataset	V	A
Edit-MIDI-melody	46.26%	44.8%
Edit-MIDI-accom	39.51%	48.94%

Table 4. Performance of melody and accompaniment of edited MIDI.

Table 4 shows the performance of melody and accompaniment of edited MIDI in terms of CF. Melody was found to perform better than accompaniment on valence regression, which was in contrary to arousal.

3.4 The Chorus of Edited MIDI

In most cases the chorus of a song is the emphatic part that reflects music concept. The chorus may express only one type of affective content.

We collected edited MIDI files containing only the chorus part of the corresponding songs and applied emotion regression on them. The results are showed in Figure 1.

Figure 1 shows the performance of the chorus dataset with different dataset size. Meanwhile we compared the performance of the chorus dataset with the performance of the entire MIDIs dataset (i.e. Edit-MIDI dataset). The number on the vertical axis refers to the number of MIDI files and the percentage on the horizontal axis refers to the performance of regression on valence measured in CF. The black bars refer to the performance of the chorus dataset and the gray bars refer to the performance of the corresponding entire MIDIs dataset. For each dataset size, we carried out the experiments 3 times by randomly choosing data examples.

Figure 1 revealed that the performance on the chorus dataset was very close to that on the entire MIDIs dataset.

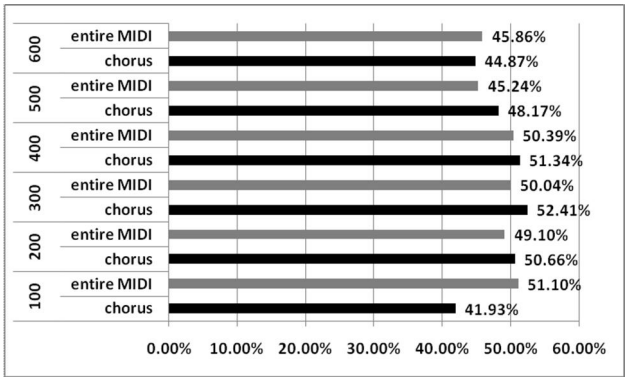


Figure 1. The performance of the chorus.

4. RESULTS ANALYSIS

The result showed in row 1 to 4 of Table 1 aligns with the work of Guan et al [2]. Lyric performed the best, converted MIDI was the second and audio performed the worst for valence regression. By converting audio to MIDI and then using MIDI features extracted from converted MIDI files, converted MIDI performed better than the audio by 2.36% on valence.

The result showed in row 3 to 4 of Table 1 indicated that the edited MIDI performs worse than the converted MIDI by 10.67% on valence.

To analyze the difference between edited MIDI and converted MIDI, we investigated the remaining features on valence after feature selection, which was shown in Table 2. The performance of the experiment dropped 3.01% (from 57.09% to 54.07%) in terms of CF on valence. This result indicated that, three types of features (Duration, Combined Strength of Two Strongest Rhythmic Pulses, and Rhythmic Variability) have stronger ability to express and distinguish emotion on valence.

Table 3 shows how the performance varied with the two conversions processes. The results indicated that the MIDI files converted from audio might not perform as well as the original edited MIDIs. From Table 1 we found that the conv-MIDI performed better than the edit-MIDI; the reason for this result might be that the source audio of conv-MIDI was much better than the source audio of MIDI-WAV-MIDI rather than that the process of conversion from the audio to MIDI that helped predicting emotion. The source audio of conv-MIDI was the original audio that people listened to, while the source audio of MIDI-WAV-MIDI was the audio files synthesized from the edited MIDI files, which were not exactly the same as the originally performed audio when listened to. If a large amount of satisfactorily-composed edited MIDIs were available, the regression performance of edited MIDIs might well be better than that of converted MIDIs. However, it was not always practical and feasible to obtain such perfect dataset. Considering the difficulty in collecting satisfactorily-composed edited MIDIs, converted MIDIs can be considered as a good choice for music emotion regression.

Table 4 shows that for edited MIDI the melody MIDI performed better than the accompaniment MIDI by 6.75% on valence and accompaniment performed better than the melody by 4.14% on arousal; this result indicates that melody is more effective and important to distinguish the positive-negative quality of affective content, and accompaniment is more effective to distinguish intensity of physical activity and mental alertness.

Figure 1 shows the regression performance on the chorus dataset and entire MIDIs dataset with different dataset size. The results showed that the performance on chorus dataset was very close to that on entire MIDIs dataset. Most of the entire MIDIs were sufficiently long to contain more than chorus part of music. On one hand, the result revealed that the use of chorus instead of the entire song did not improve the valence regression in terms of edited MIDI. On the other hand, the result showed that the chorus part of an edited MIDI might have contained as sufficient information as the whole edited MIDI for valence regression.

5. CONCLUSION

In this presented work, much valuable findings were obtained. Firstly, we found that the MIDI features extracted from converted MIDI files performed better than audio features that were extracted from audio files Secondly, we found the edited MIDI performed worse than converted MIDI under the realistic conditions. Therefore, the converted MIDI could be considered as a good choice for music emotion regression rather than the edited MIDI. We also compared and illustrated the differences between them based on features. The results indicated that three types of features (Duration, Combined Strength of Two Strongest Rhythmic Pulses, and Rhythmic Variability) have stronger ability to express and distinguish emotion on valence. Finally, we decomposed the edited MIDI and explored three aspects that were believed to be important to music emotion recognition: melody, accompaniment, and chorus. Two conclusions were drawn from the experimental results. One was that melody was more effective to valence regression and accompaniment to arousal; the other one was that the chorus of an edited MIDI may have contained as sufficient information as the whole edited MIDI for valence regression.

6. ACKNOWLEDGMENT

This work is supported by the Natural Science Foundation of China (No.61170167) and Beijing Natural Science Foundation (4112028).

7. REFERENCES

[1] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbull: "Music emotion recognition: A state of the art review, " *11th International Society for Music Information Retrieval Conference*, 2010

[2] Di Guan, Xiaoou Chen, and Deshun Yang: "Music Emotion Regression based on Multi-modal Features," *9th International Symposium on Computer Music Modeling and Recognition*, 2012.

[3] A. P. Oliveira, and Amılcar Cardoso: "Modeling affective content of music: a knowledge base approach," *Sound and Music Computing Conference*, 2008.

[4] J. A. Russell, "A circumspect model of affect," *Journal of Psychology and Social Psychology*, vol. 39, no. 6, p. 1161, 1980

[5] http://www.guitar-pro.com

[6] http://www.widisoft.com

[7] C. McKay, and I. Fujinaga: "jSymbolic: A feature extractor for MIDI files," *Proceedings of the International Computer Music Conference*, 2005

[8] D. McEnnis, C. McKay, and I. Fujinaga: "jAudio: A Feature Extraction Library," *Proceedings of the International Conference on Music Information Recognition*, 2005

[9] Weka: Data Mining Software in Java, http://www.cs.waikato.ac.nz/ml/weka

[10] Mehrabian, A.: "Framework for A Comprehensive Description and Measurement of Motional States," *Genetic, Social, and General Psychology Monographs*, vol. 121, pp. 33—361, 1995

[11] S.K. Shevade, S.S. Keerthi, C. Bhattacharyya, and K.R.K. Murthy: "Improvements to the SMO Algorithm for SVM Regression," *IEEE Transactions on Neural Networks*, 1999

RHYTHMIC PATTERN MODELING FOR BEAT AND DOWNBEAT TRACKING IN MUSICAL AUDIO

Florian Krebs, Sebastian Böck, and Gerhard Widmer
Department of Computational Perception
Johannes Kepler University, Linz, Austria
florian.krebs@jku.at

ABSTRACT

Rhythmic patterns are an important structural element in music. This paper investigates the use of rhythmic pattern modeling to infer metrical structure in musical audio recordings. We present a Hidden Markov Model (HMM) based system that simultaneously extracts beats, downbeats, tempo, meter, and rhythmic patterns. Our model builds upon the basic structure proposed by Whiteley et. al [20], which we further modified by introducing a new observation model: rhythmic patterns are learned directly from data, which makes the model adaptable to the rhythmical structure of any kind of music. For learning rhythmic patterns and evaluating beat and downbeat tracking, 697 ballroom dance pieces were annotated with beat and measure information. The results showed that explicitly modeling rhythmic patterns of dance styles drastically reduces octave errors (detection of half or double tempo) and substantially improves downbeat tracking.

1. INTRODUCTION

From its very beginnings, music has been built on temporal structure to which humans can synchronize via musical instruments and dance. The most prominent layer of this temporal structure (which most people tap their feet to) contains the approximately equally spaced *beats*. These beats can, in turn, be grouped into *measures*, segments with a constant number of beats; the first beat in each measure, which usually carries the strongest accent within the measure, is called the *downbeat*. The automatic analysis of this temporal structure in a music piece has been an active research field since the 1970s and is of prime importance for many applications such as music transcription, automatic accompaniment, expressive performance analysis, music similarity estimation, and music segmentation. However, many problems within the automatic analysis of metrical structure remain unsolved. In particular, complex rhythmic phenomena such as syncopations, triplets, and swing make it difficult to find the correct phase and period of downbeats and beats, especially for systems that rely on the assumption that beats usually occur at onset times. Considering all these rhythmic peculiarities, a general model no longer suffices.

One way to overcome this problem is to incorporate higher-level musical knowledge into the system. For example, Hockman et al. [12] proposed a genre-specific beat tracking system designed specifically for the genres hardcore, jungle, and drum and bass. Another way to make the model more specific is to model explicitly one or several *rhythmic patterns*. These rhythmic patterns describe the distribution of note onsets within a predefined time interval, e.g., one bar. For example, Goto [9] extracts bar-length drum patterns from audio signals and matches them to eight pre-stored patterns typically used in popular music. Klapuri et al. [14] proposed a HMM representing a three-level metrical grid consisting of tatum, tactus, and measure. Two rhythmic patterns were employed to obtain an observation probability for the phase of the measure pulse. The system of Whiteley et al. [20] jointly models tempo, meter, and rhythmic patterns in a Bayesian framework. Simple observation models were proposed for symbolic and audio data, but were not evaluated on polyphonic audio signals.

Although rhythmic patterns are used in some systems, no systematic study exists that investigates the importance of rhythmic patterns for analyzing the metrical structure. Apart from the approach presented in [17], which learns a single rhythmic template from data, rhythmic patterns to be used for beat tracking have so far only been designed by hand and hence depend heavily on the intuition of the developer.

This paper investigates the role of rhythmic patterns in analyzing the metrical structure in musical audio signals. We propose a new observation model for the HMM-based system described in [20], whose parameters are learned from real audio data and can therefore be adapted easily to represent any rhythmic style.

2. RHYTHMIC PATTERNS

Although rhythmic patterns could be defined at any level of the metrical structure, we restrict the definition of rhythmic patterns to the length of a single measure.

2.1 Data

As stated in Section 1, strong deviations from a straight on-beat rhythm constitute potential problems for automatic rhythmic description systems. While pop and rock music is commonly concentrated on the beat, Afro-Cuban rhythms frequently contain syncopations, for instance in the *clave* pattern – the structural core of many Afro-Cuban rhythms. Therefore, Latin music represents a serious challenge to beat and downbeat tracking systems.

The ballroom dataset [1] contains eight different dance styles (Cha cha, Jive, Quickstep, Rumba, Samba, Tango, Viennese Waltz, and (slow) Waltz) and has been used by several authors, for example, for genre recognition [6, 18]. It consists of 697 [2] 30 seconds-long audio excerpts (sampled at 11.025 kHz) and has tempo and dance style annotations. The dataset contains two different meters (3/4 and 4/4) and all pieces have constant meter. The tempo distributions of the dance styles are displayed in Fig. 4.

We have annotated both beat and downbeat times manually. In cases of disagreement on the metrical level we relied on the existing tempo and meter annotations. The annotations can be downloaded from `https://github.com/CPJKU/BallroomAnnotations`.

2.2 Representation of rhythmic patterns

Patterns such as those shown in Fig. 1 are learned in the process of inducing the likelihood function for the model (cf. Section 3.3.3), where we use the dance style labels of the training songs as indicators of different rhythmic patterns. To model dependencies between instruments in our pattern representations, we split the audio signal into two frequency bands and compute an onset feature for each of the bands individually as described in Section 3.3. To illustrate the rhythmic characteristics of different dance styles, we show the eight learned representations of rhythmic patterns in Fig. 1. Each pattern is represented by a distribution of onset feature values along a bar in two frequency bands.

For example, the *Jive* pattern displays strong accents on the second and fourth beat, a phenomenon usually referred to as *backbeat*. In addition, the typical *swing* style is clearly visible in the high-frequency band. The *Rumba* pattern contains a strong accent of the bass on the 4th and 7th eighth note, which is a common bass pattern in Afro-Cuban music and referred to as *anticipated bass* [15]. One of the characteristics of *Samba* is the shuffled bass line, a pattern originally played with the *Surdo*, a large Brazilian bass drum. The pattern features bass notes on the 1st, 4th, 5th, 9th, 12th, and 13th sixteenth note of the bar. *Waltz*, finally, is a triple meter rhythm. While the bass notes are located mainly on the downbeat, high-frequency note onsets are also located at the quarter and eighth note level of the measure.

[1] The data was extracted from `www.ballroomdancers.com`.

[2] One of the 698 original files was found duplicated and was removed.

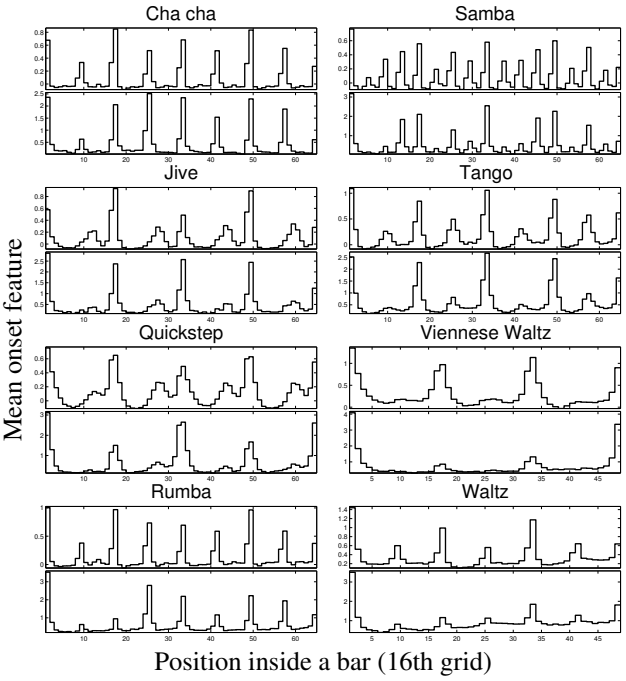


Figure 1. Illustration of learned rhythmic patterns. Two frequency bands are shown (Low/High from bottom to top).

3. METHOD

In this section, we describe the *dynamic Bayesian network* (DBN) [16] we use to analyze the metrical structure. We assume that a time series of *observed* data $\mathbf{y}_{1:K} = \{\mathbf{y}_1, ..., \mathbf{y}_K\}$ is generated by a set of unknown, *hidden* variables $\mathbf{x}_{1:K} = \{\mathbf{x}_1, ..., \mathbf{x}_K\}$, where K is the length of an audio excerpt in frames. In a DBN, the joint distribution $P(\mathbf{y}_{1:K}, \mathbf{x}_{1:K})$ factorizes as

$$P(\mathbf{y}_{1:K}, \mathbf{x}_{1:K}) = P(\mathbf{x}_1) \prod_{k=2}^{K} P(\mathbf{x}_k|\mathbf{x}_{k-1}) P(\mathbf{y}_k|\mathbf{x}_k) \quad (1)$$

where $P(\mathbf{x}_1)$ 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*.

The proposed model is similar to the model proposed by Whiteley et. al [20] with the following modifications:

- We assume conditional dependence between the tempo and the rhythmic pattern (cf., Section 3.2), which is a valid assumption for ballroom music as shown in Fig. 4.
- As the original observation model was mainly intended for percussive sounds, we replace it by a Gaussian Mixture Model (GMM) as described in Section 3.3.

3.1 Hidden variables

The *dynamic bar pointer model* [20] defines the state of a hypothetical bar pointer at time $t_k = k \cdot \Delta$, with $k \in \{1, 2, ..., K\}$ and Δ the audio frame length, by the following discrete hidden variables:

1. Position inside a bar $m_k \in \{1, 2, ..., M\}$, where $m_k = 1$ indicates the beginning and $m_k = M$ the end of a bar;

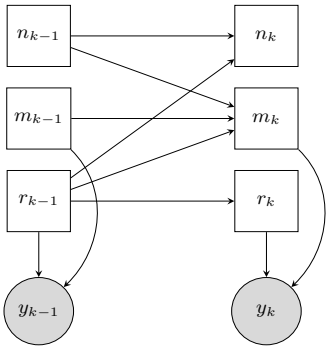


Figure 2. Dynamic Bayesian network; circles denote continuous variables and rectangles discrete variables. The gray nodes are observed, and the white nodes represent the hidden variables.

2. Tempo $n_k \in \{1, 2, ..., N\}$ (unit $\frac{\text{bar positions}}{\text{audio frame}}$), where N denotes the number of tempo states;

3. Rhythmic pattern $r_k \in \{r_1, r_2, ..., r_R\}$, where R denotes the number of rhythmic patterns.

For the experiments reported in this paper, we chose $\Delta = 20$ ms, $M = 1216$, $N = 26$, and R (the number of rhythmic patterns) was 2 or 8 as described in Section 4.2. Furthermore, each rhythmic pattern is assigned to a meter $\theta(r_k) \in \{3/4, 4/4\}$, which is important to determine the measure boundaries in Eq. 4. The conditional independence relations between these variables are shown in Fig. 2.

As noted in [16], any discrete state DBN can be converted into a regular HMM by merging all hidden variables of one time slice into a 'meta-variable' $\mathbf{x}_k$, whose state space is the Cartesian product of the single variables:

$$\mathbf{x}_k = [m_k, n_k, r_k]. \tag{2}$$

3.2 Transition model

Due to the conditional independence relations shown in Fig. 2, the transition model factorizes as

$$\begin{aligned} P(\mathbf{x}_k|\mathbf{x}_{k-1}) = \quad & P(m_k|m_{k-1}, n_{k-1}, r_{k-1}) \times \\ & \times P(n_k|n_{k-1}, r_{k-1}) \times P(r_k|r_{k-1}) \end{aligned} \tag{3}$$

where the three factors are defined as follows:

- $P(m_k|m_{k-1}, n_{k-1}, r_{k-1})$
  At time frame k the bar pointer moves from position m_{k-1} to m_k as defined by

  $$m_k = [(m_{k-1} + n_{k-1} - 1) \text{mod} (N_m \cdot \theta(r_{k-1}))] + 1. \tag{4}$$

  Whenever the bar pointer crosses a bar border it is reset to 1 (as modeled by the modulo operator).

- $P(n_k|n_{k-1}, r_{k-1})$
  If the tempo n_{k-1} is inside the allowed tempo range $\{n_{min}(r_{k-1}), ..., n_{max}(r_{k-1})\}$, there are three possible transitions: the bar pointer remains at the same tempo, accelerates, or decelerates:
  if $n_{min}(r_{k-1}) \leq n_{k-1} \geq n_{max}(r_{k-1})$,

  $$P(n_k|n_{k-1}) = \begin{cases} 1 - p_n, & n_k = n_{k-1}; \\ \frac{p_n}{2}, & n_k = n_{k-1} + 1; \\ \frac{p_n}{2}, & n_k = n_{k-1} - 1. \end{cases} \tag{5}$$

  Transitions to tempi outside the allowed range are assigned a zero probability. p_n is the probability of a change in tempo per audio frame, and the step-size of a tempo change per audio frame was set to one bar position per audio frame.

- $P(r_k|r_{k-1})$
  For this work, we assume a musical piece to have a characteristic rhythmic pattern that remains constant throughout the song; thus we obtain

  $$r_{k+1} = r_k. \tag{6}$$

3.3 Observation model

For simplicity, we omit the frame indices k in this section. The observation model $P(\mathbf{y}|\mathbf{x})$ reduces to $P(\mathbf{y}|m, r)$ due to the independence assumptions shown in Fig. 2.

3.3.1 Observation features

Since the perception of beats depends heavily on the perception of played musical notes, we believe that a good onset feature is also a good beat tracking feature. Therefore, we use a variant of the *LogFiltSpecFlux* onset feature, which performed well in recent comparisons of onset detection functions [1] and is summarized in the top part of Fig. 3. We believe that the bass instruments play an important role in defining rhythmic patterns, hence we compute onsets in low-frequencies (< 250 Hz) and high-frequencies (> 250 Hz) separately. In Section 5.1 we investigate the importance of using the two-dimensional onset feature over a one-dimensional one. Finally, we subtract the moving average computed over a window of one second and normalize the features of each excerpt to zero mean and unity variance.

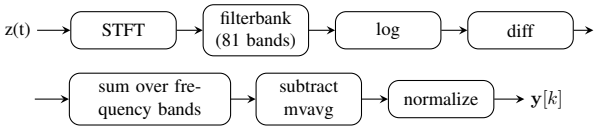


Figure 3. Computing the onset feature $\mathbf{y}$[k] from the audio signal $z(t)$

3.3.2 State tying

We assume the observation probabilities to be constant within a 16th note grid. All states within this grid are tied and thus share the same parameters, which yields 64 (4/4 meter) and 48 (3/4 meter) different observation probabilities per bar and rhythmic pattern.

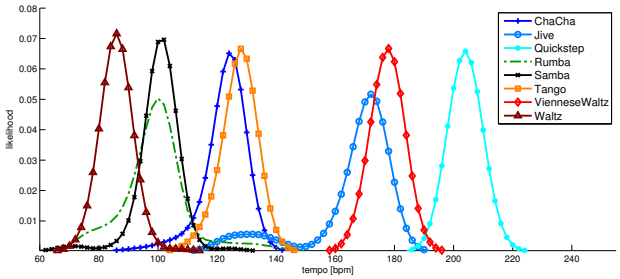


Figure 4. Tempo distributions of the ballroom dataset dance styles. The displayed distributions are obtained by (Gaussian) kernel density estimation for each dance style separately.

3.3.3 Likelihood function

To learn a representation of $P(\mathbf{y}|m,r)$, we split the training dataset into pieces of one bar length, starting at the downbeat. For each bar position within the 16th grid and each rhythmic pattern, we collect all corresponding feature values and fit a GMM. We achieved the best results on our test set with a GMM of $I = 2$ components. Hence, the observation probability is modeled by

$$P(\mathbf{y}|m,r) = \sum_{i=1}^{I} w_{m,r,i} \cdot \mathcal{N}(\mathbf{y}; \mu_{m,r,i}, \Sigma_{m,r,i}), \qquad (7)$$

where $\mu_{m,r,i}$ is the mean vector, $\Sigma_{m,r,i}$ is the covariance matrix, and $w_{m,r,i}$ is the mixture weight of component i of the GMM. Since, in learning the likelihood function $P(\mathbf{y}|m,r)$, a GMM is fitted to the audio features for every rhythmic pattern (i.e., dance style) label r, the resulting GMMs can be interpreted directly as representations of rhythmic patterns. Fig. 1 shows the mean values of the features per frequency band and bar position for the GMMs corresponding to the eight rhythmic patterns $r \in \{$Cha cha, Jive, Quickstep, Rumba, Samba, Tango, Viennese Waltz, Waltz$\}$.

3.4 Initial state distribution

The bar position and the rhythmic patterns are assumed to be distributed uniformly, whereas the tempo state probabilities are modeled by fitting a GMM [3] to the tempo distribution of each ballroom style shown in Fig. 4.

3.5 Inference

We are looking for the state sequence $\mathbf{x}^*_{1:K}$ with the highest posterior probability $p(\mathbf{x}_{1:K}|\mathbf{y}_{1:K})$:

$$\mathbf{x}^*_{1:K} = \arg\max_{x_{1:K}} p(\mathbf{x}_{1:K}|\mathbf{y}_{1:K}). \qquad (8)$$

We solve Eq. 8 using the Viterbi algorithm [19]. Once $\mathbf{x}^*_{1:K}$ is computed, the set of beat and downbeat times are obtained by interpolating $m^*_{1:K}$ at the corresponding bar positions.

[3] The number of components was set to two (PS2), and four (PS8)

4. EXPERIMENTAL SETUP

We use different settings and reference methods to evaluate the relevance of rhythmic pattern modeling for the beat and downbeat tracking performance.

4.1 Evaluation measures

A variety of measures for evaluating beat tracking performance is available (see [3] for an overview). We chose to report continuity-based measures for beat and downbeat tracking as in [4, 5, 14]:

- CMLc (Correct Metrical Level with continuity required) assesses the longest segment of correct beats at the correct metrical level.
- CMLt (Correct Metrical Level with no continuity required) assesses the total number of correct beats at the correct metrical level.
- AMLc (Allowed Metrical Level with continuity required) assesses the longest segment of correct beats, considering several metrical levels and offbeats.
- AMLt (Allowed Metrical Level with no continuity required) assesses the total number of correct beats, considering several metrical levels and offbeats.

Due to lack of space, we present only the mean values per measure across all files of the dataset. Please visit `http://www.cp.jku.at/people/krebs/ISMIR2013.html` for detailed results and other metrics.

4.2 Systems compared

To evaluate the use of modeling multiple rhythmic patterns, we report results for the following variants of the proposed system (PS): PS2 uses two rhythmic patterns (one for each meter), PS8 uses eight rhythmic patterns (one for each genre), PS8.genre has the ground truth genre, and PS2.meter has the ground truth meter as additional input features.

In order to compare the system to the state-of-the-art, we add results of six reference beat tracking algorithms: Ellis [7], Davies [4], Degara [5], Böck [2], Ircambeat [17], and Klapuri [14]. The latter two also compute downbeat times.

4.3 Parameter training

For all variants of the proposed system PSx, the results were computed by a leave-one-out approach, where we trained the model on all songs except the one to be tested. The Böck system has been trained on the data specified in [2], the SMC [13], and the Hainsworth dataset [10]. The beat templates used by Ircambeat in [17] have been trained using their own annotated PopRock dataset. The other methods do not require any training.

4.4 Statistical tests

In Section 5.1 we use an analysis of variance test (ANOVA) and in Section 5.2 a multiple comparison test [11] to find

System	CMLc	CMLt	AMLc	AMLt
PS2.1d	**62.2**	**65.8**	**87.6**	**93.1**
PS2.2d	**66.7**	**70.1**	**88.5**	**93.2**
PS8.1d	**76.6**	**79.7**	**87.7**	**92.1**
PS8.2d	**79.5**	**83.0**	**87.6**	**91.6**
PS2	66.7	70.1	**88.5**	**93.2**
PS8	**79.5**	**83.0**	87.6	**91.6**
Ellis [7]	26.7	30.9	65.2	80.2
Davies [4]	57.9	59.2	87.9	89.8
Degara [5]	64.6	66.9	85.3	89.5
Ircambeat [17]	58.1	60.3	86.1	89.6
Böck [2]	65.7	67.7	**92.0**	**94.4**
Klapuri [14]	55.2	57.0	84.9	87.3
PS2.meter	68.0	71.7	88.7	93.7
PS8.genre	89.9	93.7	90.9	94.8

Table 1. Beat tracking performance on the ballroom dataset. Results printed in bold are statistically equivalent to the best result.

statistically significant differences among the mean performances of the different systems. A significance level of 0.05 was used to declare performance differences as statistically relevant.

5. RESULTS AND DISCUSSION

5.1 Dimensionality of the observation feature

As described in Section 3.3.1, the onset feature is computed for one (PS*x*.1d) or two (PS*x*.2d) frequency bands separately. The top parts of Table 1 and Table 2 show the effect of the dimensionality of the feature vector on the beat and downbeat tracking results respectively.

For beat tracking, analyzing the onset function in two separate frequency bands seems to help finding the correct metrical level, as indicated by higher CML measures in Table 1. Even though the improvement is not significant, this effect was observed for both PS2 and PS8.

For downbeat tracking, we have found a significant improvement for all measures if two bands are used instead of a single one, as evident from Table 2. This seems plausible, as the bass plays a major role in defining a rhythmic pattern (see Section 2.2) and helps to resolve the ambiguity between the different beat positions within a bar.

Using three or more onset frequency bands did not improve the performance further in our experiments. In the following sections we will only report the results for the two-dimensional onset feature (PS*x*.2d) and simply denote it as PS*x*.

5.2 Relevance of rhythmic pattern modeling

In this section, we evaluate the relevance of rhythmic pattern modeling by comparing the beat and downbeat tracking performance of the proposed systems to six reference systems.

System	CMLc	CMLt	AMLc	AMLt
PS2.1d	46.9	47.1	70.5	71.1
PS2.2d	**55.5**	**55.7**	**76.2**	**76.5**
PS8.1d	65.4	65.8	80.9	81.8
PS8.2d	**71.1**	**71.5**	**85.3**	**85.9**
PS2	55.5	55.7	76.2	76.5
PS8	**71.1**	**71.5**	**85.3**	**85.9**
Ircambeat [17]	36.5	37.4	57.4	59.4
Klapuri [14]	39.6	40.1	68.1	68.9
PS2.meter	62.1	62.4	84.2	84.6
PS8.genre	82.8	83.1	92.6	92.9

Table 2. Downbeat tracking performance on the ballroom dataset. Results printed in bold are statistically equivalent to the best result.

5.2.1 Beat tracking

The beat tracking results of the reference methods are displayed together with PS2 (=PS2.2d) and PS8 (=PS8.2d) in the middle part of Table 1. Although there is no single system that performs best in all of the measures, we can still determine a best system for the CML measures and one for the AML measures separately.

For the CML measures (which require the correct metrical level), PS8 clearly outperforms all other systems. If the correct dance style is supplied as in PS8.genre, the performance increases even more. Apparently, the dance style provides sufficient rhythmic information to resolve tempo ambiguities.

For the AML measures (which do not require the correct metrical level), we found no advantage of using the proposed methods over most of the reference methods. The system proposed by Böck, which has been trained on Pop/ Rock music, outperforms all other systems, even though the difference to PS2 (for AMLc and AMLt) and PS8 (for AMLt) is not significant.

Hence, if the correct metrical level is unimportant or even ambiguous, a general model like Böck or any other reference system might be preferable to the more complex PS8. On the contrary, in applications where the correct metrical level matters (e.g., a system that detects beats and downbeats for automatic ballroom dance instructions [8]), PS8 is the best system to chose.

Knowing the meter a priori (PS2.meter) was not found to increase the performance significantly compared to PS2. It appeared that meter was identified mostly correct by PS2 (in 89% of the songs) and that for the remaining 11% songs both of the rhythmic patterns fitted equally well.

5.2.2 Downbeat tracking

Table 2 lists the results for downbeat tracking. As shown, PS8 outperforms all other systems significantly in all metrics. In cases where the dance style is known a priori (PS8.genre), the downbeat performance increases even more. The same was observed for PS2 if the meter was known (PS2.meter). This leads to the assumption that downbeat

tracking (as well as beat tracking with PS8) would improve even more by including meter or genre detection methods. For instance, Pohle et al. [18] report a dance style classification rate of 89% on the same dataset, whereas PS8 detected the correct dance style in only 75% of the cases.

The poor performance of Ircambeat and Klapuri's system is probably caused by the fact that both systems were developed for music comprising a completely different metrical structure than present in ballroom data. In addition, Klapuri's system explicitly assumes 4/4 meter (only true for 522 songs) and relies on the high-frequency content of the signal (that is drastically reduced using a sampling rate of 11.025 kHz) to determine the measure boundaries.

6. CONCLUSION AND FUTURE WORK

In this study, we investigated the influence of explicit modeling of rhythmic patterns on the beat and downbeat tracking performance in musical audio signals. For this purpose we have proposed a new observation model for the system proposed in [20] representing rhythmical patterns in two frequency bands.

Our experiments indicated that computing an onset feature for at least two different frequency bands increases the downbeat tracking performance significantly compared to a single feature covering the whole frequency range.

In a comparison with six reference systems, explicitly modeling dance styles as rhythmic patterns was shown to reduce octave errors (detecting half or double tempo) in beat tracking. Besides, downbeat tracking was improved substantially compared to a variant that only models meter and two reference systems.

Obviously, ballroom music is well structured in terms of rhythmic patterns and tempo distribution. If all the findings reported in this paper also apply to music genres other than ballroom music has yet to be investigated.

In this work, the rhythmic patterns were determined by dance style labels. In future work, we want to use unsupervised clustering methods to extract meaningful rhythmic patterns from the audio features directly.

7. ACKNOWLEDGMENTS

We are thankful to Simon Dixon for providing access to the first bar annotations of the ballroom dataset and to Norberto Degara and the reviewers for inspiring inputs. This work was supported by the Austrian Science Fund (FWF) project Z159 and the European Union Seventh Framework Programme FP7 / 2007-2013 through the PHENICX project (grant agreement no. 601166).

8. REFERENCES

[1] S. Böck, F. Krebs, and M. Schedl. Evaluating the online capabilities of onset detection methods. In *Proceedings of the 14th International Conference on Music Information Retrieval (ISMIR)*, Porto, 2012.

[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] M. 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.

[4] M. Davies and M. Plumbley. Context-dependent beat tracking of musical audio. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1009–1020, 2007.

[5] N. Degara, E. Argones Rua, A. Pena, S. Torres-Guijarro, M. Davies, and M. Plumbley. Reliability-informed beat tracking of musical signals. *Audio, Speech, and Language Processing, IEEE Transactions on*, (99):1–1, 2011.

[6] S. Dixon, F. Gouyon, and G. Widmer. Towards characterisation of music via rhythmic patterns. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, 2004.

[7] D. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[8] F. Eyben, B. Schuller, S. Reiter, and G. Rigoll. Wearable assistance for the ballroom-dance hobbyist-holistic rhythm analysis and dance-style classification. In *Proceedings of the 8th IEEE International Conference on Multimedia and Expo (ICME)*, Beijing, 2007.

[9] M. Goto. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159–171, 2001.

[10] S. Hainsworth and M. Macleod. Particle filtering applied to musical tempo tracking. *EURASIP Journal on Applied Signal Processing*, 2004:2385–2395, 2004.

[11] Y. Hochberg and A. Tamhane. *Multiple comparison procedures*. John Wiley & Sons, Inc., 1987.

[12] J. Hockman, M. Davies, and I. Fujinaga. One in the jungle: Downbeat detection in hardcore, jungle, and drum and bass. In *Proceedings of the 13th International Society for Music Information Retrieval (ISMIR)*, Porto, 2012.

[13] A. Holzapfel, M. Davies, J. Zapata, J. Oliveira, and F. Gouyon. Selective sampling for beat tracking evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(9):2539–2548, 2012.

[14] 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.

[15] P. Manuel. The anticipated bass in cuban popular music. *Latin American music review*, 6(2):249–261, 1985.

[16] K. Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

[17] 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.

[18] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Proceedings of the 10th International Society for Music Information Retrieval (ISMIR)*, Kobe, 2009.

[19] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[20] N. Whiteley, A. Cemgil, and S. Godsill. Bayesian modelling of temporal structure in musical audio. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, Victoria, 2006.

LARGE-SCALE COVER SONG IDENTIFICATION USING CHORD PROFILES

Maksim Khadkevich
Fondazione Bruno Kessler-irst,
via Sommarive 18, Povo 38050, Italy
hadkevich@gmail.com

Maurizio Omologo
Fondazione Bruno Kessler-irst,
via Sommarive 18, Povo 38050, Italy
omologo@fbk.eu

ABSTRACT

This paper focuses on cover song identification among datasets potentially containing millions of songs. A compact representation of music contents plays an important role in large-scale analysis and retrieval. The proposed approach is based on high-level summarization of musical songs using chord profiles. Search is performed in two steps. In the first step, the Locality Sensitive Hashing (LHS) method is used to retrieve songs with similar chord profiles. On the resulting list of songs a second processing step is applied to progressively refine the ranking. Experiments conducted on both the Million Song Dataset (MSD) and a subset of the Second Hand Songs (SHS) dataset showed the effectiveness of the proposed solution, which provides state-of-the-art results.

1. INTRODUCTION

Recent advances in digital media have allowed for extensive wide-spread growth of musical collections. We entered an era of content-based multimedia search engines, boosting the demand for advanced audio analysis tools and applications. Cover song identification based on the analysis of audio contents is a challenging problem, caused by the fact that different renditions of a song can differ in tempo, instrumentation, key, or genre. Given this, audio spectral contents of two covers can vary significantly from one another. During the last decade, the problem of cover song identification has been of a great interest to scientists working in the Music Information Retrieval (MIR) research area [1–3]. Identifying cover songs can help detect copyright infringements and correctly handle music license management.

In this paper, we propose an approach to large-scale cover song identification using chord progressions and chord profiles. A chord progression is extracted from audio or from chroma features provided with the Million Song Dataset (MSD) [4]. For the most part, the approaches proposed in the literature are based on the alignment of local features, which is typically performed by Dynamic Time Warping (DTW) [1], or string alignment [5], and require a significant amount of computational resources. To build a system that can operate on a large scale, we propose the use of chord profiles for indexing and fast retrieval. A chord profile of a song is a compact representation that summarizes the rate of occurrence of each chord. To solve the problem of cover song identification, we propose the use of a well-established approach for fast retrieval in multi-dimensional spaces, which is Locality-Sensitive Hashing (LHS). A more accurate comparison is then performed between the top k results, based on the alignment of chord progressions.

1.1 Previous work

Most of the cover song identification systems reported in the literature work on small datasets (up to several thousands of songs) and involve pair-wise comparison, when a query song is matched against all the songs in the database [1, 2]. This makes them impractical for large collections, containing millions of items. A good overview of existing approaches is given in [3].

Recent works on large-scale datasets are based on indexing and fast retrieval. So far, several indexing schemas have been introduced which allow for fast searching over large databases. Bertin-Mahieux and Ellis [6] proposed using "chroma jump codes". They extract beat-synchronized chromagram, discover chroma bins with high energy, and construct "jump codes" between such pairs. For retrieval, "jump codes" extracted from a queried song are searched in the database and results are ranked according to the number of matching pairs. In the approach of Bertin-Mahieux and Ellis [7], 2-D Fast Fourier Transform (FFT) of overlapping chromagram segments corresponding to 75 beats are averaged with subsequent application of Principal Component Analysis (PCA). Distances between vectors of PCA components are used to generate ranked lists.

Several approaches have been proposed based on chord sequence alignment. Lee [8] used chord sequences for cover song identification. His approach is based on the extraction of chords by means of a HMM-based recognizer trained on data generated from MIDI. The songs are ranked according to a DTW-based pairwise similarity on key-transposed sequences. Bello [5] extended this approach, systematically evaluating key shifting, the cost of gap insertions and chord swaps in string alignment. Martin et al. [9] adapted tools from computational biology to align

chord sequences. However, their system could not perform well on a subset of the MSD dataset, probably due to some simplifications in the chord representations; instead of taking into account chord types, they extract only chord roots. These works show that chord progressions can be successfully used as high-level features for the problem of cover song identification.

The previous works listed above operate either on frame-level features such as chroma [1, 2] or chords [5, 8], or by extracting very compact descriptors of the whole song [6, 7]. The former approaches are more accurate but do not scale to large databases, since brute-force sequence matching is computationally expensive. The latter approaches allow for scalability, but usually are limited in terms of performance. In this paper, the problem of cover song identification is approached trying to find a compromise between fast retrieval and exhaustive matching.

1.2 Organization of the paper

The datasets and the evaluation methodology are presented in Section 2. Section 3 briefly describes the proposed system architecture, including front-end processing and indexing schema. Section 4 is devoted to experimental results. Some conclusions and future work are finally presented in Section 5.

2. DATASET AND EVALUATION METHODOLOGY

The lack of large datasets for cover song identification is a problem, which has always been an obstacle to develop systems suitable for commercial use, i.e., able to run queries on a large database comprising millions of songs. The Million Songs Dataset (MSD) is the first attempt to create a large dataset for different MIR tasks. The MSD dataset contains features and meta-data for one million songs. All the features are extracted using the EchoNest [1] API. Unfortunately, the distribution of the MSD dataset does not contain chords. We used chroma segments and beats to estimate chord sequences from MSD features using the template-matching approach [10].

The Second Hand Songs (SHS) [2] dataset consists of meta-data for around 250000 songs. For about 50000 songs a link to YouTube [3] video is available.

An intersection of the SHS and the MSD datasets was used to build the SHS-MSD dataset. 18196 songs from the SHS dataset are chosen to form the SHS-MSD dataset. It was proposed to divide the SHS-MSD dataset into training (12960 tracks) and test (5854 tracks) parts [6]. They will be referred to as SHS-training and SHS-test, respectively.

Due to the fact that data provided with the MSD dataset does not contain waveforms, and extracted chroma features are probably not the best solution to produce chord recognition rate comparable to state-of-the-art systems, we collected our own dataset. It will be referred to as SHS-WAV dataset. We used the SHS website to download 24282 videos, from which audio tracks were extracted. All the tracks from the SHS-WAV dataset are divided in 5650 cover groups. The largest group contains 128 covers, which is "Summertime" by George Gershwin. To our knowledge, this is the largest collection of audio waveforms suitable for the evaluation of large-scale cover song identification systems. The list of the songs for both the datasets and the corresponding extracted chords are publicly available [4].

We follow the same evaluation methodology as proposed in [7]. As with many other information retrieval systems that produce a list of ranked items as a result, we adopt Mean Average Precision (MAP) as the main evaluation metric. We also report Average Rank (AR), but this metric is less informative and can be misleading when used alone. AR is mostly influenced by the most difficult covers, while differences in the top of the rank are neglected [7]. We also present distribution statistics of the ranked songs for the whole MSD dataset with a particular emphasis on the top ten results.

3. PROPOSED SYSTEM ARCHITECTURE

Chords and melody are considered to be essential characteristics of a song. They are the two attributes that describe tonal and harmonic properties of a musical piece and allow us to identify a song among many others, regardless of tempo, instrumentation, or genre. The proposed cover song identification system is based on the use of chord progressions and chord profiles. Chords are considered to be a high-level descriptor. Despite possible local changes in the different renditions of a song (e.g., a major chord is replaced by a minor one) the general characteristics of the chord progressions embedded in it are typically preserved from one cover to another. Therefore, in this paper we explore different ways of exploiting chord progressions and chord profiles to build a robust and large-scale cover song identification system. The block diagram of the proposed system presented in Figure 1. It comprises the following structural components: high-level feature extraction, indexing and retrieval.

3.1 Chord progression extraction

Chord progressions and chord profiles are the two high-level features used in the proposed system. The extraction of beat-synchronous chord progression is the first step. Two different datasets are used to evaluate the proposed approach, one containing audio waveforms and the other containing only features and metadata. Correspondingly, two different chord extraction techniques are adopted as discussed below.

In the case of SHS-WAV dataset, for the extraction of beats and chords from audio waveform we use Vamp [5] plugins *Chordino* and *BarBeatTracker* that show state-of-the-

[1] http://www.echonest.com
[2] http://www.secondhandsongs.com/
[3] http://www.youtube.com/
[4] https://github.com/FBK-SHINE/CoverSongData
[5] http://www.vamp.org

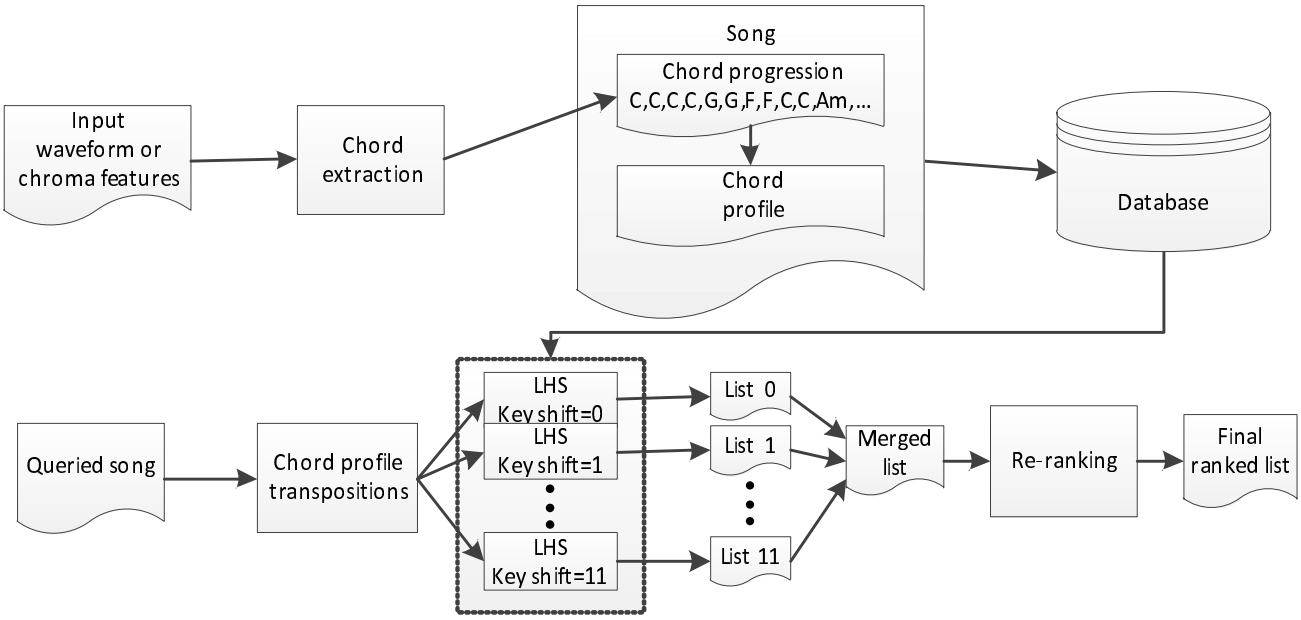


Figure 1: Block diagram of the proposed system

art results. The beat structure is used to obtain a tempo-independent sequence of chords. Once chords and beat structure are extracted, the chords are split into beat segments so that each beat segment contains one chord. If a chord transition occurs inside a beat segment, the chord segment that has the longest intersection with the current beat segment is used to derive the chord label. The chord dictionary comprises two chord types, which are *major* and *minor*.

To derive chord progressions from the MSD dataset, beat-synchronous chroma features are first extracted. We follow the approach of Bertin-Mahieux and Ellis [6], where chroma vectors are averaged across beat segments. In the second step, we apply the template matching technique proposed in [10]. Template matching for chord recognition is based on the idea of introducing a set of templates for each chord type. The template configurations are derived heuristically. We define a binary mask as a 12-dimensional chord template in which the pitch classes that correspond to constituent notes of the given chord are set to one, while the other components are set to zero. A binary template T is defined as

$$T = [Z_C, Z_{C\sharp}, Z_D, Z_{D\sharp}, Z_E, ..., Z_{A\sharp}, Z_B] \qquad (1)$$

where Z_p denotes the mask value that corresponds to the pitch class p. For example, binary masks for C major and D minor chords would take the following form:

```
T(C:maj)=[1,0,0,0,1,0,0,1,0,0,0,0]
T(D:min)=[0,0,1,0,0,1,0,0,0,1,0,0]
```

The template that produces the highest cosine similarity between chroma vectors is used to generate a chord label for the given beat segment. The cosine similarity between vectors a and b is defined as

$$S_c(a,b) = \frac{a \cdot b}{\|a\| \, \|b\|} \qquad (2)$$

where $\|\cdot\|$ denotes Euclidean distance.

The resulting sequence of beat-aligned chords is subsequently used as a compact representation of the harmonic structure of the song. However, the comparison of chord progressions is usually done by sequence alignment, which is a computationally expensive operation and cannot be used on a large scale. Therefore, we propose to further compress the extracted high-level features. This can be done by discarding the temporal information and compacting all the related contents in a chord profile. A chord profile is a 24-dimensional vector, in which each dimension corresponds to the rate of occurrence of a chord. Let n_i be number of beat segments containing chord i, where $i \in 1..24$. Then, the i-th component of the chord profile vector c is calculated as $c_i = \frac{n_i}{N}$, where N is the total number of beat segments.

Chord profiles and chord progressions extracted for each song of a given dataset are stored in a database. In the retrieval stage, high-level features extracted from a queried song are used to derive a ranked list of possible covers from that database, as described in the following section.

3.2 Retrieval

The proposed cover song identification system relies on a two-step retrieval schema. Given a large database of chord profiles and a queried song, we address the problem of finding the nearest neighbors. Finding the nearest neighbors of an element in large databases is a well-known problem addressed in many areas of information retrieval. For low-dimensional data, nearest neighbor search can be performed by partitioning the search space using, for example, k-d trees as mentioned in [11]. Data with high number of dimensions cause the so-called "curse of dimensionality", when the distance between neighboring points tend to be large [12]. Locality Sensitive Hashing is a probabilistic approach to reduce dimensionality by hashing features

so that items that are close to each other fall in the same bucket with high probability. Casey and Slaney [13] used LHS for fast shingle retrieval from a comparatively large database. Casey et al. [14] extended their work, performing analysis of optimal parameters and giving examples of LHS application for different MIR tasks. Yu et al. [15] proposed an adapted two-level LHS scheme to tackle the problem of retrieving multi-variant audio tracks. In their work, a study on trade-off between identification accuracy and efficiency is presented.

For small datasets containing several thousands of songs, a straightforward approach suggests computing the distances between a queried song and all the items in the dataset. When working with larger databases, containing several millions of songs, a significant increase in speed can be achieved by using LHS. Following the approach of Casey and Slaney [13], in the first step we use LHS to retrieve the nearest neighbours. L_1 distance is adopted as distance metric. Given two chord profiles a and b, the distance between them is defined as

$$||a - b||_1 = \sum_{i=1}^{24} |a_i - b_i| \quad (3)$$

Note that L_2 distance was used in our early experiments, with definitely worse results. However, this topic should be matter of further investigation.

Due to the fact that a cover of a given song can be performed in a different key, we should introduce a mechanism to make the distance between chord profiles key-invariant. This can be achieved by performing a circular permutation of a queried song chord profile 12 times, taking into account all the possible key transpositions. For each transposition, we retrieve a list of candidates. In the final part of the first stage, all the lists are merged and all the retrieved songs are ranked according to (3).

In the second step, the top k results are re-ranked by computing edit (or Levenshtein) distances between chord progressions. The edit distance is the number of insertions, deletions and substitutions to transform one sequence into another. Thus, a more accurate matching is performed, which takes temporal information into account. Time complexity of computing edit distance is $O(nm)$, where n and m are chord progression lengths of the songs under comparison. Choosing k depends on the balance between speed and precision. In our experimental setup, we set $k = 2000$, which means that the top 2000 results from the merged rank list obtained in the first stage are re-ranked according to edit distance between chord progressions. In this way, the output is refined taking into account temporal alignment between a queried song and the top k items from the merged ranked list.

4. EXPERIMENTAL RESULTS

4.1 Evaluating MSD features

The errors produced by the chord extraction propagate through successive processing steps and eventually have an impact on cover song identification performance. As a result, it was important to understand if the features provided by the EchoNest API could be used for an accurate chord estimation. As opposed to frame-based [10] or beat-synchronous [1] approaches to extracting chroma features, the EchoNest API has its own segmentation algorithm which is independent of beat positions. The chroma vectors are averaged across segments that can contain several beats. On the other hand, some beats can contain several such segments. Another limitation of these chroma features is the absence of bass information. It has been shown that using a lower frequency content, as extra-feature, leads to a significant improvement in performance [16, 17]. Given this, the chroma vectors delivered with EchoNest API might not be the best features for automatic chord extraction.

In fact, in our preliminary experiments we applied the chroma feature extraction available with the EchoNest API on the MIREX 2011 corpus to evaluate chord recognition performance. The corpus consists of 220 songs of Beatles, Queen, Zweieck and Carol King. The template-based approach described in Section 3.1 was used to generate chord labels.

The experiment led to a chord recognition rate of 55.7%, compared to 77.8% obtained using the frame-level template matching based on time-frequency reassigned chroma features proposed in [18]. In the latter case, a 24-dimensional chroma vector was used, where the first and the second 12 dimensions corresponded to bass and treble contents, respectively. This suggests that using alternative chroma feature sets to represent a song can lead to an improvement in chord recognition rate, and as a consequence in cover song identification performance.

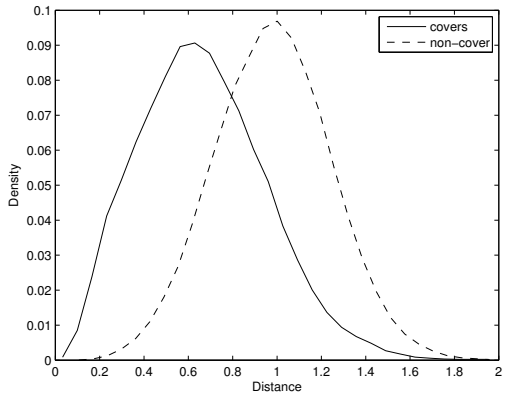
In the following experiments, we compare the performance of the proposed system when applied to the MSD dataset and to the collected SHS-WAV dataset.

4.2 Chroma profiles

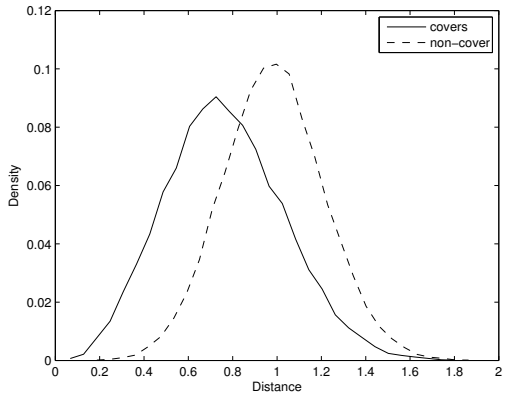
The first set of experiments aimed to collect statistics of chord profile distances between covers and non-covers. The results for the SHS-training and the SHS-WAV datasets are presented in Figure 2.

To generate statistics on covers, we calculated distances between chord profiles corresponding to all cover pairs in the given dataset. As for non-covers, for each song we randomly choose a non-cover song. In order to take into account possible key shifts, the distance between two chord profiles is defined as the minimum distance among those obtained for all the 12 possible circular permutations.

As shown in Figure 2a, for SHS-WAV dataset, Gaussian-like distributions are obtained with mean values and standard deviations of (0.66, 0.28) and (0.97, 0.26), for cover and non-cover pairs, respectively. A similar behavior is obtained with SHS-training dataset, for which mean values and standard deviations are (0.76 0.26) and (0.98, 0.22), respectively.



(a) SHS-WAV dataset



(b) SHS-training dataset

Figure 2: Probability density function of distance between cover and non-cover pairs for SHS-WAV and SHS-training datasets

4.3 Cover song identification results

Table 1 presents the results on the three datasets. We compare our system with "chroma jumps" [6] and 2D-FFT [7]. On all the datasets, the proposed approach showed the best results. The results reported for "SHS-training" were obtained when using the training subset of MSD-SHS dataset (12960 tracks). The results reported for "SHS-WAV" were obtained when using waveforms extracted from YouTube videos (24282 tracks). The "Full MSD" labeled columns refer to the most important experimental setup, which indicates the scalability of the approach and shows the performance of the proposed features on the full MSD dataset. Covers from SHS-test dataset (5854 tracks) were used for querying.

It is interesting to see that the proposed system performed significantly better on SHS-WAV dataset, if compared to SHS-training dataset. This fact confirms that the chroma features provided with MSD dataset are not the best solution for chord extraction. It is likely that the very low performance at this moment obtained on the full MSD dataset can be significantly improved by processing the original waveforms.

Figures 3 and 4 show the distribution of ranks for the experiments on the full MSD dataset. 10% of the covers were ranked in the top 1000, and around 25% of them in the top 10000. For 255 queries, a cover was placed in the first position, while 716 covers appeared in the top-ten ranked list.

Table 1: Experimental results on three datasets

System	Average Rank	MAP
SHS-training		
proposed approach	958.2	0.10753
2DFTM (200 PC) [7]	3,005.1	0.09475
2DFTM (50 PC) [7]	2,939.8	0.07759
SHS-WAV		
proposed approach	1,378.4	0.2062
Full MSD		
proposed approach	114,951	0.03709
2DFTM (200 PC) [7]	180,304	0.02954
2DFTM (50 PC) [7]	173,117	0.01999
jcodes 2 [6]	308,370	0.00213

Table 2: Runtime and memory footprints

Dataset	Size (songs)	sec/query	memory
SHS-WAV	24282	1.46	346M
SHS-training	12960	1.08	198M
full MSD	1 million	7.56	3690M

Average querying runtime for each dataset is presented in Table 2. All the experiments were conducted on a modern laptop with 8GB of RAM installed and CPU Intel i7-2760QM running at 2.4 GHz. Due to the compactness of the chord progressions and chord profiles, it is possible to put all the features extracted from full MSD dataset into a hash table and store it in RAM. Memory footprint was around 3.5 GB. In terms of speed, average runtime per query without any parallelization appeared to be 7.56 seconds when searching in the full MSD dataset.

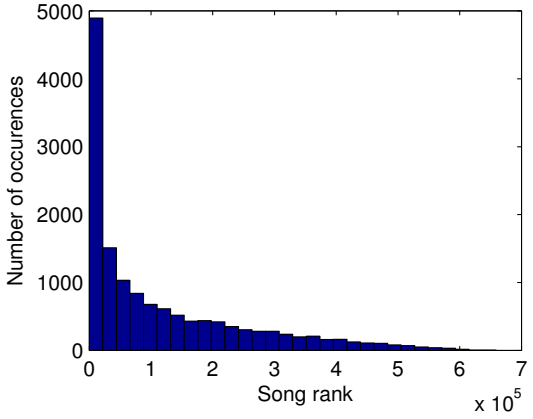


Figure 3: Rank distribution for the results on full MSD dataset

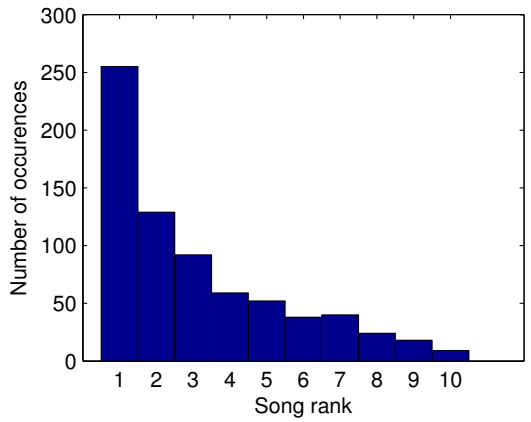


Figure 4: Top ten rank distribution for the results on full MSD dataset

5. CONCLUSION

In this paper, we proposed a new system for scalable cover song identification. The experimental results showed that chord profiles can be used as an extremely compact high-level feature that summarizes harmonic properties of a song. The proposed two-step approach improves the systems on which we compared performance and suggests room for an improvement. More sophisticated distances than Levenshtein could be used for sequence alignment, such as NeedlemanWunsch or Smith-Waterman, which were used in [5] and [2], respectively. More efficient ways of introducing key-invariance can be investigated. Instead of using 12 circular permutation to make a query, an alternative feature vector representation can be utilized. Experimental results obtained on different datasets showed that switching from precomputed feature data provided by the EchoNest API to state-of-the-art high-level feature extractors may further improve the performance.

6. REFERENCES

[1] D. P. W. Ellis and G. E. Poliner, "Identifying 'cover songs' with chroma features and dynamic programming beat tracking," in *Proc. ICASSP*, vol. 4, April 2007, pp. IV–1429–IV–1432.

[2] J. Serrà and E. Gómez, "Audio cover song identification based on tonal sequence alignment," in *Proc. ICASSP*. Las Vegas, USA: IEEE, 2008, pp. 61–64.

[3] J. Serrà, "Identification of versions of the same musical composition by processing audio descriptions," Ph.D. dissertation, Universitat Pompeu Fabra, Barcelona, 2011.

[4] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proc. ISMIR*, Miami, USA, 2011, pp. 591 – 596.

[5] J. P. Bello, "Audio-based cover song retrieval using approximate chord sequences: testing shifts, gaps, swaps and beats," in *Proc. ISMIR*, Vienna, Austria, 2007, pp. 239–244.

[6] T. Bertin-Mahieux and D. P. W. Ellis, "Large-scale cover song recognition using hashed chroma landmarks," in *Proc. WASPAA*. New York, USA: IEEE, 2011, pp. 117 – 120.

[7] T. Bertin-Mahieux and D. P. W. Ellis, "Large-scale cover song recognition using the 2d fourier transform magnitude," in *Proc. ISMIR*, Porto, Portugal, 2012, pp. 241 – 246.

[8] K. Lee, "Identifying cover songs from audio using harmonic representation," in *MIREX task on Audio Cover Song Identification*, 2006.

[9] B. Martin, D. G. Brown, P. Hanna, and P. Ferraro, "Blast for audio sequences alignment: A fast scalable cover identification tool," in *Proc. ISMIR*, Porto, Portugal, 2012, pp. 529 – 534.

[10] L. Oudre, Y. Grenier, and C. Févotte, "Template-based chord recognition : Influence of the chord types," in *Proc. ISMIR*, Kobe, Japan, 2009, pp. 153–158.

[11] M. Slaney, Y. Lifshits, and J. He, "Optimal parameters for locality-sensitive hashing," *Proceedings of the IEEE*, vol. 100, no. 9, pp. 2604–2623, 2012.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[13] M. Casey and M. Slaney, "Fast recognition of remixed music audio," in *Proc. ICASSP*, vol. 4, Honolulu, Hawaii, USA, 2007, pp. IV–1425–IV–1428.

[14] M. Casey, C. Rhodes, and M. Slaney, "Analysis of minimum distances in high-dimensional musical spaces," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 5, pp. 1015–1028, July 2008.

[15] Y. Yu, M. Crucianu, V. Oria, and L. Chen, "Local summarization and multi-level lsh for retrieving multi-variant audio tracks," in *Proc. ACM international conference on Multimedia*, Beijing, China, 2009, pp. 341–350.

[16] M. Mauch and S. Dixon, "Approximate note transcription for the improved identification of difficult chords," in *Proc. ISMIR*, Utrecht, Netherlands, 2010, pp. 135–140.

[17] M. Khadkevich and M. Omologo, "Time-frequency reassigned features for automatic chord recognition," in *Proc. ICASSP*, Prague, Czech Republic, 2011, pp. 181 – 184.

[18] M. Khadkevich and M. Omologo, "Reassigned spectrum-based feature extraction for gmm-based automatic chord recognition," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2013, no. 1, pp. 1–12, 2013.

AUTOMATICALLY IDENTIFYING VOCAL EXPRESSIONS FOR MUSIC TRANSCRIPTION

Sai Sumanth Miryala Microsoft Research India mssumanth99@gmail.com
Kalika Bali Microsoft Research India kalikab@microsoft.com
Ranjita Bhagwan Microsoft Research India bhagwan@microsoft.com
Monojit Choudhury Microsoft Research India monojitc@microsoft.com

ABSTRACT

Music transcription has many uses ranging from music information retrieval to better education tools. An important component of automated transcription is the identification and labeling of different kinds of vocal expressions such as vibrato, glides, and riffs. In Indian Classical Music such expressions are particularly important since a *raga* is often established and identified by the correct use of these expressions. It is not only important to classify *what* the expression is, but also *when* it starts and ends in a vocal rendition. Some examples of such expressions that are key to Indian music are *Meend* (vocal glides) and *Andolan* (very slow vibrato).

In this paper, we present an algorithm for the automatic transcription and expression identification of vocal renditions with specific application to North Indian Classical Music. Using expert human annotation as the ground truth, we evaluate this algorithm and compare it with two machine-learning approaches. Our results show that we correctly identify the expressions and transcribe vocal music with 85% accuracy. As a part of this effort, we have created a corpus of 35 voice recordings, of which 12 recordings are annotated by experts. The corpus is available for download [1] .

1. INTRODUCTION

Vocal expressions, such as glides, licks, and vibrato are an intrinsic part of vocal music of any genre. The use of suitable vocal expressions establishes the characteristic mood of a song and enhances its emotional appeal. In western classical music, the appropriate use of vibrato and tremolo while singing can drastically change the appeal of a given piece.

Similarly, in North Indian Classical Music (NICM), not only do vocal expressions enhance or characterize a song's mood, they also establish the correctness of a *raga*'s [2] rendition. For instance, the nature of the glide between two notes is a key difference between two *ragas*, *Bhupali* and *Shuddha Kalyan*, that are based on the same melodic scale (see Figure 1). Whether the glide from the third to the fifth note is quick or slow differentiates them (see Table 1 for note definitions). Also, whether a note is steady or oscillating can also depend on the *raga* (see Figure 2).

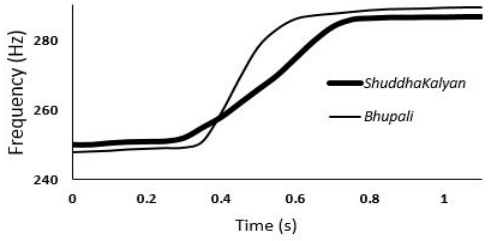


Figure 1. The glide from the third to the fifth note for *raga Bhupali* is quick, but in *Shuddha Kalyan*, it is slower and perceivably touches the fourth's microtones.

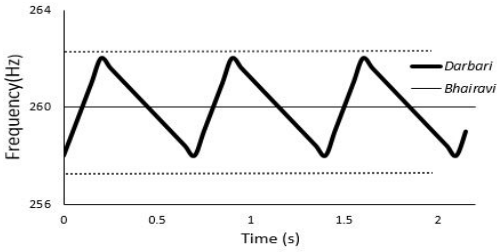


Figure 2. The third note is steady in *raga Bhairavi*, but oscillates across the note's microtones in *raga Darbari*

Hence, automatically identifying and labeling such vocal expressions is very important for accurate music transcription. In addition, identifying vocal expressions is a basic requirement towards building tools for music education. Such an educational tool can process a student's rendition, identify the vocal expression and provide feedback, visual or auditory, on the correctness.

In this paper, we propose an algorithm for automatically identifying vocal expressions and can therefore be a building-block for various music transcription and education tools. The proposed algorithm (a) estimates the pitch curve, (b) identifies the singing voice frames, (c) processes the pitch envelope to obtain a canonical representation and

[1] http://research.microsoft.com/apps/pubs/?id=198396

[2] A *raga* is based on an ascending and descending scale, but is characterized using many other features and evades a formal definition. See [2] for a detailed exposition.

(d) uses *templates* to identify each expression and finally create a transcription of the audio signal.

We concentrate on two main expressions predominantly used in NICM: *Meend* or a slow glide between notes, and *Andolan* or slow microtonal oscillations within the same note. We limit our discussion in this paper to these because they are two of the most important vocal expressions that determine the correctness of a *raga*. While we do not claim that our specific algorithm can capture every kind of vocal expression across all genres, we believe that the overall approach we have proposed can be used to identify different styles of vocal expressions across different genres.

The ground-truth is created by manual annotation of the recordings by experts. We compare the results of our work with two different machine learning techniques: decision trees and conditional random fields. Our findings show that we can achieve up to 85% accuracy across all vocal expressions identified, an improvement of 7% over a machine learning-based competitive baseline.

2. RELATED WORK

The automatic transcription of polyphonic music needs to deal with source separation and note detection in the primary source. Rao et al. [12] proposed a system for extracting vocal melody from a polyphonic music signal in NICM using a spectral harmonic-matching pitch detection algorithm. The voice signal extracted using these methods can then be input to our algorithm for identifying expressions. However this method does not apply to identifying fine vocal expressions.

Typically, audio alignment for note detection, specifically onset, steady period and offset of a note, employ signal processing methods like Dynamic Time Warping (DTW) in conjunction with graphical models like Hidden Markov Models (HMM). Devaney et al. [4] used an HMM model with acoustic features like power and aperiodicity along with DTW priors to align as well as identify the transient as well as steady portions of a note. Our technique does not use onset detection for reasons outlined in Section 3.2.

Classification of *ragas* based on the concept of Pitch Class Distribution Dyads (PCDD) [3] uses spectrum based pitch extraction and onset-detection to create Pitch Class Distribution (PCD) and PCDD. A classifier is then used on the PCD and PCDDs to identify *raga* labels. Ross et al. [13] detects melodic motifs to identify repetition of phrases in a *raga* rendition. While all these methods are successful to a certain extent, they do not take into account the vocal expressions that may be specific to the composition or introduced by the singer for a more rich rendition, or in the music education scenarios, mistakes by a learner.

The biggest challenge for an automatic transcription of NICM is the absence of a written score that makes any top-down processing using the score as a knowledge source practically impossible. A number of approaches for the transcription of Western music [9] have made use of the availability of a score as one of the knowledge sources in their models. Klapuri [6], has an extensive discussion on the use of a "musicological model" as a part of a transcription system. While the system developed in [6] primarily uses the sinusoidal properties of the polyphonic music signal, the concluding discussion clearly points to the use of an existing score for improvement. In a later version of the polyphonic music transcription [7], they make use of a reference database for note event modeling, as well as a musicological model for the transcription system. The work reported in this paper makes no assumptions about the availability of a musical score or a knowledge model and aims to identify and label the different vocal expressions from the signal using signal processing and machine learning techniques.

Sa	*Re*	*Ga*	*Ma*	*Pa*	*Dha*	*Ni*
Do	*Re*	*Mi*	*Fa*	*So*	*La*	*Ti*
1^{st}	2^{nd}	3^{rd}	4^{rd}	5^{rd}	6^{rd}	7^{rd}

Table 1. Relative note names in Indian (top) and Western (bottom) traditions .

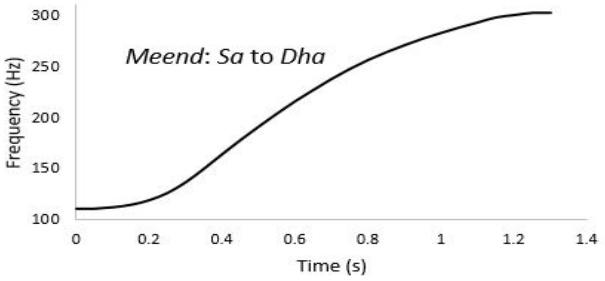


Figure 3. Pitch Envelope of a *meend* from *Sa* to *Dha* in *raga Darbari*, *Sa* is 220Hz.

3. BACKGROUND AND DEFINITIONS

NICM follows a relative note system, where all the notes are sung with respect to the *tonic* which is called *Sa* (same as the *Do*). The frequency of the tonic depends on the singer. Table 1 shows the Indian names for the seven notes corresponding to the western *Do-Re-Mi*. In this paper, we focus on the *Alap*, which is a meterless form of singing that typically starts any NICM rendition. The *alap* captures the essence of the *raga* being rendered. *Alap* is usually sung with no accompaniment except for a background drone called the *Tanpura*, which provides the singer a reference to the tonic.

3.1 Vocal Expressions

As mentioned earlier, the NICM genre uses various characteristic vocal expressions for enhancing as well as establishing a raga rendition In our work, we concentrate specifically on the *meend* and the *andolan*.

Meend is a smooth glide from one note to another, as shown in Figure 3, where the singer moves from *Sa* to *Dha*. This is clearly distinct from a steady note, as shown in Figure 4 (top). A very short glide is often termed as a *sparsh*. For the purpose of this work, we will use the term glide to refer to both of these.

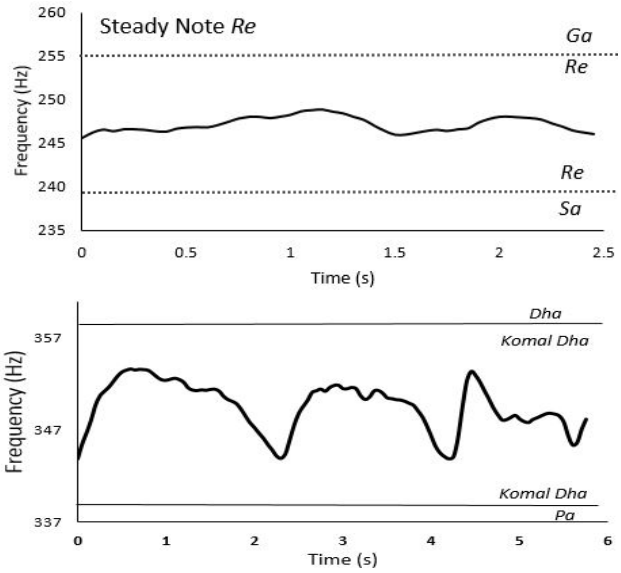


Figure 4. Pitch Envelope of a steady *Re* (top) and *andolan* around *komal Dha* (bottom) in *raga Darbari*.

Andolan is a gentle swing that oscillates between the lower and the higher microtones of a certain note. Fig. 4 (bottom) shows the pitch curve of an *andolan* sung around *komal Dha* or the minor sixth.

3.2 Problem Definition

Given an audio recording of vocal music in NICM, we want to label it with a set of annotations that clearly mark the steady notes as well as the vocal expressions, viz. *meend*, *sparsh* and *andolan*. In the example in Figure 5. From 0 to 0.6s, a steady *Sa* (or the first note) is sung, followed by a *meend* to the *Re* or the major second, until time 1.2s. This is followed by a steady rendition of *Re* until time 1.6s. After a *meend* to *komal Ga* or the minor third, the singer sings an *andolan* around *komal Ga*. This extends from 2.2s to 3.9s. Given a vocal rendition in NICM, our objective is to output the time-series of such annotations.

3.3 Challenges

The task of identifying these vocal expressions and transcribing NICM faces two primary challenges. First, there is no written score available. Indian classical music is an improvisational art-form, and textual representation of musical pieces, if they exist are very rough and used only as a tentative guide. There are no equivalent notations for vocal expressions like trills, vibrato, or tremolo, that exist quite extensively in western classical music.

Second, in Indian classical music notes generally do not correspond to clear onsets. Hence, conventional transcription methods that rely on onset detection cannot be used. Onset detection algorithms depend on detecting transient regions in the signal, including sudden bursts in energy or changes in the spectrum of the signal etc. These methods fail whenever there is a smooth glide between two notes. In this work, we present a transcription scheme, which relies solely on the pitch envelope.

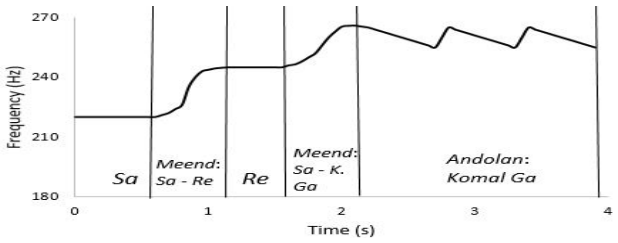


Figure 5. Annotation of an audio recording, using a mock pitch envelope.

4. TRANSCRIPTION AND EXPRESSION IDENTIFICATION

The first step towards transcription is the estimation of fundamental frequency. We chose a difference function based method for this purpose. This method was preferred over the frequency-domain methods, because real-time performances are important for music education tools. No significant improvement was observed by using the spectral methods. Any accurate pitch detection method maybe used for this step.

The second step is to detect audio segments with vocal singing as against segments with only the background drone. From the background drone, we detect the tonic, or the absolute frequency of the base-note for the singer. The third step is to obtain a canonical representation of the pitch curve, for which we use a line fitting algorithm to the curve. The final step is to perform vocal expression identification using template representations for each expression. These steps are explained in the following sections.

4.1 Pitch Estimation

In this work the terms pitch and fundamental frequency are used interchangeably. For each frame, the difference function is evaluated by subtracting the original signal with delayed versions of itself. The fundamental frequency corresponds to the absolute minimum of the difference function. The other local minima correspond to the harmonics. We fixed the frame size as 50 ms with 50% overlap for this work. A low-pass filter with a cutoff frequency of 700Hz is applied before pitch estimation to remove high frequency content. Variance of the pitch track is chosen as a measure to eliminate octave errors using the Viterbi algorithm. In this work, source separation or multi-band pitch estimation is not necessary as the singing voice masks the *tanpura* sound well.

4.2 Drone and Tonic Detection

The background drone or the *tanpura* is a stringed instrument that provides a continuous pitch reference to a vocal performer. The drone usually consists of four strings, three of them at the tonic of the scale, and one string tuned to the fifth note. To identify the singing voice frames in the recording, we use resonance properties of the drone. Due to the special form of the bridge fixed to a resonant body, *tanpura* shows remarkably different acoustic properties compared to other stringed instruments [11]. The wide

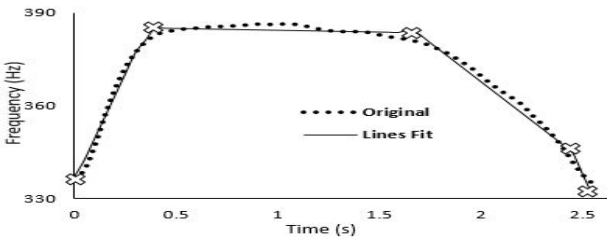


Figure 6. Pitch envelope & Lines fit using the proposed method. Critical points are marked with 'X'

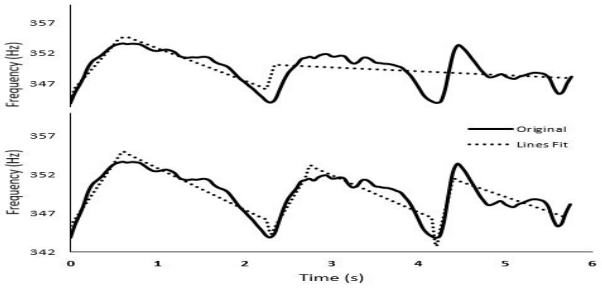


Figure 7. Pitch envelope of the *andolan* note & Lines fit using threshold JND1 (top) & JND1/2 (bottom).

body of the bridge induces a large number of overtones that manifest in the output of the pitch estimation algorithm. In frames that contain voice, these overtones are masked by the voice.

Consequently, the frames with only the drone have higher entropy than the frames that contain voice. We therefore use an entropy based method [5] to differentiate the singing voice frames from the drone. For each audio recording, we dynamically estimate an entropy threshold from the histogram of entropy values. Any frame with lower entropy than the threshold is labeled as a singing voice frame, while the frames with higher entropy are labeled as the *tanpura* frames. Tonic is calculated as the mode of the pitch values in the frames where *tanpura* is prominently audible.

4.3 Line Fitting

The third step in the process, is to obtain a canonical representation of the pitch-curve in terms of straight lines and *critical points* of inflection. We use an augmented version of a previously proposed line-fitting algorithm [1] for this purpose. We outline our algorithm as follows:

Step 1: Identify the local minima and maxima in the pitch curve. This gives us a set of points representing the curve. To achieve better fit along transient regions, the set of points where the singer is changing notes are added to the list. These points are identified by scaling down the pitch values to one octave and mapping the frequencies to notes. Start a sweep from the left of the curve.

Step 2: Start from the first point on the curve, and connect it to the third point using a straight line. From this line, if the second point lies within the distance specified by Just Noticeable Difference (JND) threshold (equation 1), the second point is removed. Then, connect the first point to the fourth point and repeat the JND threshold-based check for the third point. Repeat this process until you find a point that lies outside the JND threshold. This point is the starting point for the next iteration.

Step 3: Starting from the new critical point, repeat Step 2 to find the next critical point. Continue this process until the whole pitch curve is represented by a set of critical points and fit lines between these points, by minimizing the squared error between the pitch curve and the lines fit.

$$JND1(F) = 3.13 - \frac{0.13 * F}{100} \quad (1)$$

Figure 6 shows a sample pitch envelope and the final critical points identified. For each pitch curve, we calculate canonical representations by varying the value of the JND threshold. Small variations in pitch due to singing imperfections are eliminated in the canonical representation. These representations are useful in the identification of vocal expressions of certain types, as we shall describe in the next subsection.

4.4 Identifying Vocal Expressions

Given the canonical representation of the pitch curve, we use *templates* for each kind of vocal expression to recognize and classify them. A template for each expression is a loose representation based on some subset of duration, line lengths, slopes, and number of points. In the next subsections, we describe templates for *andolan* and *meend*, and how we detect these expressions.

4.4.1 Andolan

Using the expert manual annotations and by studying the pitch curves ourselves, we have found that an *andolan* has the following template: six to ten straight lines, with consecutive lines having alternating slope signs. All pitch values that the lines touch *should be within the same or adjacent notes*. This template captures the slow oscillations of an *andolan*, that touch the microtones within a single note. This is as opposed to a vibrato, which manifests itself as a much faster oscillation. However, we could use a similar template for vibrato detection as well.

To match this template, we look for such a pattern across the different canonical representations of the pitch curve that are obtained by decreasing the JND threshold value iteratively a maximum of 3 times. In our work, we have used the thresholds $JND1$, $JND1/2$ and $0.4 * JND1$. The threshold needs to be decreased, because the amplitude of the oscillation can vary from very small to quite large. With large JND thresholds, the canonical representation may not capture all the oscillations, as shown in Figure 7.

However, if the threshold is too low, the oscillatory pattern may be found in steady notes too. So, the threshold should not be decreased too much. If we find such a definite pattern in at least one of the canonical representations, we classify the corresponding segment of the pitch curve as an *andolan*. This matching algorithm is similar to using DTW iteratively to do the template matching.

4.4.2 Glides

The template for a glide is the following: a line between two critical points where the line starts and ends at *different* notes. Any line which satisfies this property is either a *meend* or *sparsh*. If the glide segment is longer than 300 ms, it is labeled a *meend*, else a *sparsh*.

4.4.3 Steady notes

Segments with no expressions are typically represented as horizontal lines, with very low slope values. Hence, we use this simple template to classify segments as steady notes. Steady notes are transcribed using the median of the pitch values between its two end points.

5. EVALUATION & RESULTS

In this Section, we describe our evaluation of the proposed algorithm. First, we describe and characterize the data we collect for evaluation. Next, we compare the accuracy of our technique with that of two machine-learning techniques – the C5.0 Decision Tree Classifier [10], and a Conditional Random Field classifier (CRF) [8] and present the results.

5.1 Data Collection

We have created a corpus of 35 recordings in 8 *ragas* sung by 6 singers of varying expertise which are publicly available for the purposes of Music Information Retrieval research. In this paper, we use 12 recordings of 3 singers singing *alap* in 4 *ragas* for evaluation of our algorithms. We ensured that all the three singers sang identical pieces, with the same set of notes and same vocal expressions in each *raga*. This is to ensure that we have a balanced dataset across different *ragas* and different singers.

We asked two experts, one who has been a professional music teacher for 25 years, and the other a serious music learner for 11 years, to manually annotate the vocal expressions on 12 recordings sung by 3 singers in 4 different *ragas*. We had one expert annotate each file first, and the second expert revised and verified these annotations to ensure no expressions were missed. In case of an ambiguity among the two annotators, the more experienced annotator's labels are used. Each file is approximately 2.5 minutes long, and the sum total of the length of all twelve recordings is approximately 30 minutes. The audio was collected in a recording studio and is therefore comparatively noiseless.

The experts used Praat [14] to annotate these recordings. Praat allows a user to listen to and annotate audio files, while also displaying the pitch envelope. Using Praat textgrids, the experts annotated each file with note and expression boundaries, and they labeled each segment as either *tanpura*, Steady, *andolan*, *meend*, or *sparsh*. Annotating a 3 minute recording took the two experts 120 minutes on average. Therefore, a total of about 24 hours were required for the manual annotation process.

Feature	No. of Features
Pitch	2n+1
First derivative of Pitch	2n
Pitch (warped to one octave)	2n+1
Entropy	2n+1
Amplitude (Normalized)	2n+1

Table 2. Features for the classifiers.

	Proposed	DT	CRF	Improvement(%)
Drone	0.964	0.966	0.956	-0.18
Steady	0.828	0.825	0.828	5.4
Andolan	0.647	0.449	0.448	44.31
Meend	0.72	0.38	0.314	89.49
Sparsh	0.651	0.295	0.344	89.31

Table 3. F1-scores for each class. The last column shows the percentage improvement that our approach shows over the better classifier

5.2 Evaluation Methodology

The annotations of the algorithms are compared with the ground-truth, frame to frame and the overall accuracy is defined as the percentage of frames labeled correctly. We compare the classification of these frames by our algorithm with that of two stock classifiers: the C5.0 decision tree, and CRF. The reason for trying the CRF is to evaluate a classifier which uses a notion of time or sequences, which seems inherent to expressions such as the *andolan*. The decision tree, on the other hand, does not incorporate time. The weights vector for CRF is initialized randomly and each note is considered as a sequence. These methods are evaluated using the leave one out cross-validation method.

The features used for classification are shown in table 2. To provide the note change information, pitch is warped down to one octave and fed to the classifier. We collect these features for the current frame, and a fixed number (n) of frames before and after the current frame. The performance of the classifiers is similar across several values of n. In this section, we report the results for $n = 10$.

5.3 Results

The overall accuracy of our technique is 84.7%, whereas with CRF, it is 77.6% and with C5.0, it is 77%. Hence, our proposed method improves the error produced by the better classifier by 31.7%.

Table 3 shows the F1-scores for each class, for each of the evaluated techniques. All the three approaches identify the Drone segments with about 96% accuracy. Our approach shows a 5.4% improvement over the better of the two classifiers for steady notes. For the more complex vocal expressions, our approach shows much higher improve-

Algorithm	Singer 1	Singer 2	Singer 3
Proposed	15.27	13.63	16.8
DT	21.25	21.96	25.44
CRF	18.69	20.72	21.05

Table 4. singer-wise classification errors (in % of frames)

Algorithm	*Bhairav*	*Darbari*	*Janpuri*	*Lalit*
Proposed	14.49	17.02	20.97	10.65
DT	24.77	29.98	20.69	17.45
CRF	20.43	24.82	22.61	12.75

Table 5. *raga*-wise classification errors (in % of frames)

ment: 44.31% for *andolan*, and about 89% for the glides.

Note that this is in spite of the machine learning methods using approximately 90% of the available frames as training data. Our algorithm, on the other hand, does not use any training data. Moreover, we have no tunable thresholds in the core algorithm. We do use fixed thresholds for certain templates, for instance, to differentiate *meends* from short glides (*sparsh*). However, given the nature of these expressions, we feel this is unavoidable.

However, for all three evaluated techniques, the F1-scores for identifying the vocal expressions are much lower than those for identifying steady notes. For instance, the F1-score for *andolan* using our approach is 0.647, for *meend* it is 0.72, whereas for Steady is 0.828. One source of error is that the boundaries between the glides and steady notes, as annotated by the experts, do not align exactly with the algorithm's labels. Therefore, some frames in these boundary regions, which the expert has annotated as glides are very often mis-labeled by our approach as steady. Another source of error is the mis-labeling of vocal expressions by the annotators. Some of the short vocal glides are hard to perceive and are labeled 'steady' by the annotators. In case of *andolan*, if the range of oscillation is less, the algorithms would identify it as a steady note and sometimes the pitch estimation algorithm does not pick up the microtonal variation accurately enough. Also, the way in which these expressions are achieved sometimes depends on the *raga* and the singer's expertise.

Table 4 shows the classification error by singer. Singer 1 is a professional artiste with 30 years of training, Singer 2 is a music educationist with 15 years of training, and Singer 3 is a music educationist with 40 years of training. Singer 3 uses much smaller microtonal variations in the rendition of *andolans*, some of which are labeled as steady. Hence, the errors are slightly higher for Singer 3 across all three approaches as compared to Singers 1 and 2.

Table 5 shows the classification error by *raga*. Of the four *ragas*, only *Lalit* does not use *andolans*. *Janpuri* and *Darbari*, on the other hand, use significant amounts of this vocal expression. Hence, the error associated with *Lalit* is a lot less(10.65% using our approach), and that associated with *Jaunpuri* (20.97%) and *Darbari*(17.02%) are higher.

6. CONCLUSIONS

We proposed an algorithm for automatic expression identification in vocal music. The idea is to use templates for each expression and match these templates in the pitch curve. We compared the performance of our algorithm with two machine learning methods. Our algorithm is more accurate than the better classifier by about 7%.

In future work, we intend to apply this technique to more vocal expressions across different genres of music. We also intend to use this algorithm as a building-block in various music education and music transcription applications.

7. REFERENCES

[1] Bret Battey: "Bézier spline modeling of pitch-continuous melodic expression and ornamentation," *Computer Music Journal*, Vol. 28(4), pp. 25-39, 2004.

[2] Bhatkhande V. N, Garg P.K.: *Hindustani Sangit Paddhati*, Sakhi Prakashan, 1990.

[3] Chordia P.: "Automatic raag classification of pitch tracked performances using pitch-class and pitch-class dyad distributions," *Proc. of Intl. Computer Music Conf.*, 2006.

[4] Devaney J. et al.: "Improving MIDI-audio alignment with acoustic features," *In Proc. IEEE WASPAA*, 2009.

[5] Jia C., Bo Xu: "An improved entropy-based endpoint detection algorithm," *International Symposium on Chinese Spoken Language Processing*, 2002.

[6] Klapuri A., Ryynänen, M.P.: "Modeling of note events for singing transcription.", *ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, 2004.

[7] Klapuri A., Ryynänen, M.P.: "Transcription of the singing melody in polyphonic music.", *Proc. 7th Intl. Conf. on Music Information Retrieval*, Vol. 15, 2006.

[8] Lafferty J. et al.: "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," *Intl. Conf. on Machine Learning*, 2001.

[9] Martin K.D.: "Automatic transcription of simple polyphonic music: robust front end processing," *in the Third Joint Meeting of the Acoustical Societies of America and Japan*, 1996.

[10] J.R. Quinlan: "Induction of decision trees", *Machine Learning*, Vol. 1.1, pp. 81-106.

[11] Raman C.V.: "On some Indian stringed instruments," *Proceedings of the Indian Association for the Cultivation of Science*, Vol. 7, pp. 29–33, 1921.

[12] Rao V., Rao P.: "Vocal melody extraction in the presence of pitched accompaniment in polyphonic music.", *IEEE Trans. on Audio, Speech, and Language Processing*, Vol 18(8), pp. 2145-2154, 2010.

[13] Ross J.C., Vinutha T.P., Rao P.: "Detecting melodic motifs from audio for Hindustani classical music.", *Proceedings of the 13th Intl. Society for Music Info. Retrieval Conf.*, 2012.

[14] The PRAAT Speech Analysis and Annotation Toolkit: http://www.fon.hum.uva.nl/praat/.

HOOKED: A GAME FOR DISCOVERING WHAT MAKES MUSIC CATCHY

John Ashley Burgoyne* **Dimitrios Bountouridis†** **Jan Van Balen†** **Henkjan Honing***
* Music Cognition Group, University of Amsterdam, the Netherlands
† Department of Information and Computing Sciences, Utrecht University, the Netherlands
{j.a.burgoyne,honing}@uva.nl {d.bountouridis,j.m.h.vanbalen}@uu.nl

ABSTRACT

Although there has been some empirical research on *earworms*, songs that become caught and replayed in one's memory over and over again, there has been surprisingly little empirical research on the more general concept of the musical *hook*, the most salient moment in a piece of music, or the even more general concept of what may make music 'catchy'. Almost by definition, people like catchy music, and thus this question is a natural candidate for approaching with 'gamification'. We present the design of Hooked, a game we are using to study musical catchiness, as well as the theories underlying its design and the results of a pilot study we undertook to check its scientific validity. We found significant differences in time to recall pieces of music across different segments, identified parameters for making recall tasks more or less challenging, and found that players are not as reliable as one might expect at predicting their own recall performance.

1. INTRODUCTION

'Aha! Yes, it's *that* song!' Many music listeners, even casual listeners, have had the pleasant experience of recalling a song to memory after hearing a few seconds of its 'hook'. Likewise, many casual listeners can tell almost immediately upon hearing a new song whether it will be 'catchy'. Despite the prevalence of these musical instincts, musicology (in the broadest sense, encompassing music cognition and MIR) can provide only a limited understanding of why certain pieces music are catchy and what is distinctive about the hooks within these pieces of music. The concepts of the hook and of catchiness are vital to understanding human musical memory, but they also have implications outside of music cognition. Charles Kronengold, a musicologist, has posited that the characteristics of hooks might vary across genres and, *a fortiori*, that different assortments of hook characteristics might constitute a working definition of genre [11]. In MIR, a better understanding of hooks and catchiness would be useful for music recommendation (all else being equal, the catchier of two tunes is probably the better recommendation), measuring musical similarity (as estimating similarity between the hooks of two pieces of music may be closer to human perception than estimating similarity over complete pieces), generating satisfying segmentations of pieces of music (as hooks tend to mark the start of new sections), and to some extent, fingerprinting (as hooks are the fingerprints for the brain's retrieval system).

The boundaries between catchiness, hooks, and some other musical concepts are fuzzy. One related concept that has attracted a certain amount of empirical research is the *earworm*, songs that are so catchy that they become involuntarily stuck in one's mind [3, 7, 19]. Earworms are a much narrower phenomenon than catchiness, too narrow, we believe, for many MIR applications: Few users are looking for playlists comprising nothing but earworms. Another related concept is so-called hit-song science, which aims to predict the popularity of songs based on their musical content [6, 16]. This area of study, in contrast, is broader than our area of inquiry. Although catchiness is certainly correlated with popularity, many popular songs are quite forgettable, and we are most interested in music that remains in listeners' memories for the long term. This level of cognitive information seems to be right for contributing to the widest variety of tasks in MIR [9].

The definition of a hook itself is also fuzzy, and as musicologist Don Traut has observed, 'When we go further and ask not only "What is a hook?", but "What is it about the music that makes this a hook?", the picture gets even more blurry' [17]. From a cognitive point of view, we define a hook to be the most salient, easiest-to-recall fragment of a piece of music [9]; likewise, we define catchiness as long-term musical salience, the degree to which a musical fragment remains memorable after a period of time. By our definitions, every piece of music will have a hook – the catchiest part of the piece, whatever that may be – but some pieces of music clearly have much catchier hooks than others. In principle, a piece of music may also have multiple hooks: two or more fragments of equivalent salience that are nonetheless more salient than all others in the piece. There is agreement in the literature that hooks start at points of considerable structural change, or in other

The Netherlands Organisation for Scientific Research (NWO) funded this study under grant 640.005.004, 'Cognition-Guided Interoperability Between Collections of Musical Heritage (COGITCH)'. In addition to comments from those who tested the prototype, we received helpful suggestions from many colleagues, among them Fleur Bouwer, Aline Honingh, Berit Janssen, Jaap Murre, Johan Oomen, Carlos Vaquero, Remco Veltkamp, Lourens Waldorp, and Frans Wiering. The data are available to download from http://mcg.uva.nl/.

words, at points that we in MIR would consider to be the beginnings of new sections for the purposes of a segmentation algorithm [4, 15]. There is more debate about the duration of hooks. While songwriters will often speak of the hook as the entire chorus, in fact, only a few seconds are necessary for most listeners to recall a catchy song to memory; one study has shown that after only 400 ms, listeners can identify familiar music with a significantly greater frequency than one would expect from chance [12].

We have designed an experiment that we believe will help to quantify the effect of catchiness on musical memory. Because we consider catchiness to be long-term rather than short-term salience, this design posed some important challenges. First, we needed to be able to work with well-known recordings of well-known music in order to capture fragments that have in fact remained in participants' memories for potentially long periods of time. Individual listening histories vary widely, however, and thus this constraint also entailed the ability to use quite a large set of musical stimuli, on the order of 1000 or more. Moreover, listening histories vary with respect not only to what music participants have heard before but also to how well they know particular pieces; as such, in order to obtain reliable statistics, we also needed to be able to support a much larger number of participants than a traditional psychological experiment. Next to becoming a serious alternative to a certain class of lab-based experiments, Internet-based experiments can potentially reach a much larger, more varied and intrinsically motivated participant pool, positively influencing the ecological validity of the results [10]. Furthermore, given that most listeners enjoy catchy music, our question seems naturally suited for 'gaming with a purpose', which has already proven successful for certain tasks in MIR and for machine learning in general [1, 13]. By framing the experiment as a game, we believe we will be able to collect enough data about catchiness to support a robust analysis of recall from musical memory and also to open new possibilities for using content-based MIR to predict musical catchiness.

2. DESIGNING HOOKED

Hooked, as we have named the game, comprises three essential tasks: a recognition task, a verification task, and a prediction task. Each of them responds to a scientific need in what we felt was the most entertaining fashion possible. In this way, we hope to be able recruit the largest number of subjects possible without sacrificing scientific quality.

2.1 Recognition Task

The recognition task is the heart of the game. It stems from the idea that the defining aspect of catchiness its effect on long-term memory. In particular, the easier a fragment of music is to recall after a long period of time, the catchier it should be. Thus, a 'drop-the-needle' style quiz, whereby a piece of music starts playing from a point in the middle and players are asked to recognise it, seemed to be appropriate. As noted above, there is a consensus in the theoretical literature that the hook should start at the beginning of a new structural section (possibly including the beginning of the piece itself), and we extended this idea to limit the number of starting points to a statistically tractable subset: Music will always start playing from the beginning of a structural section. Then the amount of time it takes a player to recognise the piece is a proxy for how easy that section is to recall, or in short, how catchy it is.

Figure 1a illustrates the recognition game as implemented in our current iOS prototype. A piece of music starts playing from the start of a structural section, and players have several seconds to decide whether they know it. While players are listening, points are counting down; the faster players recognises the piece, the more points they can win.

2.2 Verification Task

In a controlled laboratory environment, it might be justifiable to trust subjects to be honest in claiming to have recognised a piece of music. In a competitive game environment, it is not. We needed a task to verify that players have truly recognised the music at the moments they claim so. Most music trivia games, e.g., SongPop, [1] would ask players to identify the title, composer, artist, or year of release, but this type of question would cause serious problems for the scientific goals of Hooked. Many listeners may know a piece of music rather well without knowing its exact title or the name of the performing artist; moreover, even for those users who do know such trivia, the extra cognitive load in recalling it in addition to recognising the music itself would have an unpredictable effect on reaction time.

Ideally, the verification task would be strictly musical, but precisely because we expect players to know the musical material fairly well, finding a strictly musical task was challenging. Playing a new fragment of music and asking the player whether it came from the same song, for example, would likely be far too easy to be a reliable test. Using any kind of audio degradation to make the task harder would likely make it too difficult in cases where the player genuinely did know the song. Using MIR tools to extract melodies, beats, or some other feature would bias the general notion of catchiness unduly toward catchiness as limited to what such an MIR tool can extract.

In the end, we were inspired by the idea that once players have fully recalled a piece of music to memory, they should be able to follow along with the song in their heads for some time even after the music stops playing. Moreover, there is evidence that absolute tempo is part of musical memory, although the error distribution is somewhat skewed in favour of overly fast tempi [14]. In Hooked, as soon as players claim to know a song, playback mutes for a few seconds. During the mute, players are asked to imagine mentally or sing along actively for a few seconds (Figure 1b). When the sound returns, half the time the music returns the correct place (i.e., the mute was genuinely only a mute) and half the time the playback is offset by a few seconds (i.e., an invisible DJ 'scratched the record' during the mute). The player must answer whether the music is in the right place. We believe that over mutes of the duration we are considering

[1] http://www.songpop.fm/

(a) Recognition (b) Verification (c) Prediction

Figure 1: Screenshots from the Hooked prototype. (a) The recognition task is the heart of the game: A song starts from the beginning of an internal musical section, chosen at random, and the player must guess the song as quickly as possible. (b) The sound then mutes for a few seconds while players try to follow along in their heads. When the sound comes back, players must verify that the song is playing back from the correct place. (c) Occasionally, players instead must do the reverse: predict which of two sections is catchiest, in order to store bonus rounds for themselves.

for Hooked, players who truly remember a song should be capable of following along well enough to identify whether the music has returned in the correct place. The primary challenge is finding empirical evidence for the optimal mute time: not so short that one can judge the continuation on the basis of common-sense musical knowledge or timbral characteristics (type 2 error) but also not so long that it would interfere with the speed of imagining that might well be faster than in singing (type 1 error).

2.3 Prediction Task

We have argued here that because the notion of catchiness inherently invokes musical memory, a scientific definition of the term must involve ease of recall. The recognition game seeks to quantify listeners' behaviour on this axis. We would also like to know how well this formal definition corresponds to listeners' informal intuitions for what it catchy and what is not. As such, we decided to include periodic rounds of the game where we turn the recognition task on its head and ask players to choose which of two fragments from the same song is catchier. An image of such a round in our prototype appears in Figure 1c.

As a survey question, this task is pleasant enough, but it was a challenge to integrate it meaningfully into the gameplay. One idea we may explore in the future is adding a social element. For example, we might ask players to try to fool online opponents by predicting which will be the *less* catchy members of each pair and sending those predictions to those opponents for a recognition task; we would then award prediction players the inverse of the number of points their opposing recognition player earns. For the moment, however, we wanted a self-standing game with an intrinsic reward for the prediction task. Our solution was bonus rounds. Each time players complete a prediction task, the chosen fragment is saved in a special buffer for each player. Periodically, the recognition task will enter a bonus round for double points, with a guarantee that the fragment selected comes from the special buffer of prediction fragments. Thus, users who spend time to do a thorough job with prediction tasks can potentially earn many extra points.

3. TESTING SCIENTIFIC SOUNDNESS

We developed a prototype of Hooked on iOS and undertook a pilot study to identify the best values of the free parameters in the design (the maximum time allowed for the recognition task, the length of the mute, and length of the offset used for false returns in the verification task) and to ensure that the scientific assumptions underlying the design were correct. We recruited 26 testers from within our academic networks, 18 men and 8 women, between the ages of 20 and 70. Most participants spent about 45 minutes testing, some at home and some in their offices, some on their own iOS devices and some on ours.

3.1 Musical Material

Although we designed Hooked to accommodate a very large corpus of music, our pilot study required a more constrained set of musical material. We chose 32 songs at random from the 2012 edition of a list of the 'greatest songs of all time' from a popular annual radio programme. In order to avoid licensing problems as the scope of the game expands, we used Spotify's iOS libraries to stream all audio and require a Spotify Premium membership to play. [2] The Echo Nest has a partnership with Spotify that includes a convenient web service for applying the Echo Nest Analyzer to tracks in Spotify's catalogue, [3] and we used this service to obtain estimates of the start times of the major structural sections

[2] http://www.spotify.com/
[3] http://developer.echonest.com/

in each song. For the 9 songs that were ranked highest on the list, we retained all sections but the first and last (which often contain silence); with these songs, we hoped to be able to show that there is indeed significant variation in recognition time across different sections of the same song. For the next 8 highest ranked, we retained a random sample constituting half of the sections, a compromise position. For the remaining 15 songs, we retained only a random pair of sections; with these songs, we hoped primarily to be able to introduce some variety for the participants so that they would have a better sense of how the game would feel with a full-sized corpus. In total, this procedure yielded 160 song sections to use for the recognition task. From among these sections, we selected just one random pair from each of the 32 songs to use for testing the prediction task.

3.2 Method

During a testing session, testers worked through recognition tasks for each of 160 sections in a random order. For the first 80 sections, we asked testers to play as they would in the real world. For the remaining 80 sections, in order to test the limits of the verification task, we asked testers to try to cheat the system by claiming that they recognised every section as soon as possible, ideally before they actually knew the song. We recorded the reaction times and whether the responses were correct for each tester and each section. Throughout a testing session, testers also had a 20 percent chance of being asked to perform a prediction task instead of a recognition task for any given round and a 10 percent chance that a recognition round would be a bonus round.

During test runs, we changed some parameters of the game after every 10 recognition tasks. Overall, there were eight possible configurations of the parameters, which we presented in a random order to each tester during both the first, 'honest' half of the test run and again during the second, 'dishonest' half. Specifically, each of three parameters took one of two distinct values, which we chose based on preliminary testing prior to the pilot. The maximum time allowed for recognition was either 10 s or 15 s; this parameter primarily affects the feel of the gameplay, but it has some scientific consequences in the rare cases where players need more than 10 s to decide whether they recognise a fragment. The mute time was either 2 s or 4 s; this parameter in principle affects the difficulty of the verification task. The offset for false returns in the verification task was either 15 s or −15 s; this parameter likewise affects the difficulty of the verification task. Testers were informed when either the maximum recognition time or mute time changed so that they could comment on their preferences; testers were not informed about changes in the offset time for false returns so as not to give extra information they could have used to cheat the verification task.

4. RESULTS

Due to personal time constraints, not all participants were able to complete the pilot in its entirety: 4 made it less than halfway through and a further 5 made it less than 80 percent through. Nonetheless, because we randomised the presentation of sections and parameter settings for each subject, we have no reason to believe that the missing data should exhibit any systematic bias.

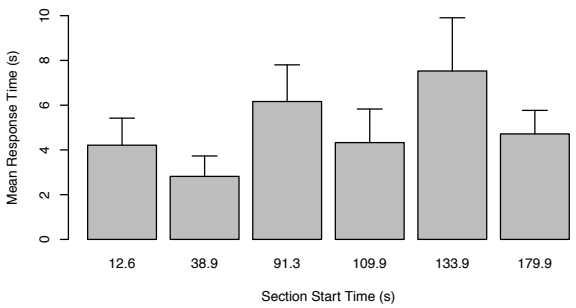


Figure 2: Mean response times from the recognition task on different sections of Adele's 'Rumour Has It'. Error bars reflect standard error. Controlling for multiple comparisons, there are significant differences ($p < .05$) in response times between the bridge (133.9 s) or the verse at 91.3 s, and the initial entry of the vocals (12.6 s) or the pre-chorus at 38.9 s.

For the recognition task, the Box-Cox procedure suggests a log transform on response time, and we assume that response times are log-normally distributed. Regressing thus across all song sections, the average response time for successfully verified claims to know a song is 5.2 s. ANOVA confirms that there are significant differences between the response times for different sections within a song even after accounting for the variation in average response time for different participants: $F(128,964) = 1.55$, MSE = 39.06, $p < .001$. Figure 2 illustrates the variation in response times for Adele's 'Rumour Has It'.[4] After correcting for multiple comparisons with Tukey's test, there are significant differences ($p < .05$) between either of the initial entry of the vocals (12.6 s, 'She, she ain't real') and the pre-chorus at 38.9 s ('Bless your soul, you've got your head in the clouds') and either of the bridge (133.9 s, 'All of these words whispered in my ear') and the second verse (91.3 s, 'Like when we creep out'). The differences in response time are as high as 4 s.

In order to tune the verification task, we needed to determine the best values to use for *maximum recognition time*, the time limit on the recognition task; *mute time*; and the *distractor offset*, the offset to use on the occasions when the sound returns from the mute in the wrong place. More specifically, the distractor offset could be either a *forward offset* of 15 s ahead of where the song should have been playing or a *backward offset* of 15 s before where the song should have been playing. We also needed to ensure that there is a sufficiently large benefit to playing honestly over random guessing. Using the player, maximum recognition time, mute time, the distractor offset, and whether the player was in the 'honest' or 'dishonest' portion of the pilot, we used a stepwise selection procedure on logistic regression models for the probability of answering the validation question correctly. Akaike's Information Criterion (AIC) prefers

[4] `spotify:track:50yHVBbU6M4iIfqBI1bxWx`

Recognition Time	p_1	95% CI	p_2	95% CI
Distractor Offset: −15 s				
10 s	.66	[.59, .73]	.26	[.21, .31]
15 s	.72	[.64, .79]	.19	[.15, .24]
Distractor Offset: +15 s				
10 s	.54	[.47, .61]	.33	[.28, .39]
15 s	.67	[.60, .74]	.33	[.28, .38]

Table 1: Probability of type 1 and type 2 errors for the validation task (i.e., answering the validation question correctly for an unknown song or answering it incorrectly for a known song) under different values of the design parameters. The ideal combination of parameters would minimise both types of error, but some trade-offs will be necessary.

a model including only the player, maximum recognition time, the distractor offset, and the 'honesty' variable with no interactions. A maximum recognition time of 15 s vs. 10 s improved a player's odds of answering the validation question correctly by 31 percent on average (95% CI [5, 62]), a distractor offset of -15 s vs. +15 s improved a player's odds of guessing correctly by 57 percent on average (95% CI [27, 93]), and playing honestly improved a player's odds of guessing correctly by 64 percent on average (95% CI [29, 111]). Table 1 summarises the verification data from the pilot in the more traditional language of type 1 and 2 errors.

In order to analyse the data from the prediction task, we use the fact that after completing a full test run, testers in the pilot had also completed recognition tasks for all fragments offered to them as choices in prediction task. We compared the choices made in prediction tasks to the difference in response times for the same fragments when they appeared in recognition tasks. Although there is a statistically significant relationship ($p = .02$), the effect is small: For each second of difference in response times, the odds of a player choosing the faster-recognised member of a pair during the prediction task increased by only 6 percent (95% CI [1, 12]). Moreover, the variance is quite high. Figure 3a shows the distribution of response-time differences where players chose the first fragment in the prediction task and Figure 3b shows the distribution where they chose the second. Although these distributions are each skewed to the appropriate side, it is clear that players are not necessarily consistent with their behaviour in the recognition task when making predictions.

5. DISCUSSION

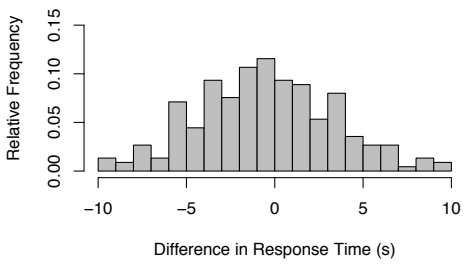
The results of our pilot of the recognition task confirm that different fragments of music, even within the same song, differ measurably in their ability to trigger musical memory. In a context where average response time is just over 5 s, the 4-s effect size is substantial. Moreover, this magnitude of response time sets us comfortably in the realm of musicological theories about hooks: something rather longer than Krumhansl's 400-ms 'plinks' [12] but also rather shorter than a complete refrain chorus, say 5 to 10 s. Historically, MIR has worked rather less with musical fragments of this scale, more often tending to consider audio frames that are shorter even than plinks or attempt to classify complete pieces. Having shown in this pilot study how important these 5-to-10-s fragments are to human musical memory, we would like to suggest that they might be especially profitable when tuning the granularity of algorithms for predicting musical similarity or recommending new music, a claim that is consistent some recent MIR research on segmentation and annotation [2, 5, 8]).

The most important limitation to this result arises from the quality of automatically generated audio segments. If, as musicological theory suggests, hooks are tied to moments of change in the musical texture, any error in the estimation of segment boundaries will propagate throughout the analysis. For a study of this size, it would have been possible to choose the segments by hand, thereby eliminating this source of error, but because our purpose was to test the feasibility of a larger-scale study where it will not be possible to choose segments by hand, we felt it was important to use automatic segmentation for our pilot, too. The analytic techniques available for larger-scale data, most notably the drift-diffusion model [18], will allow us to identify 'lag time' in segments that begin playing a bit too early, but for this study, we have to assume that such lags are noise.

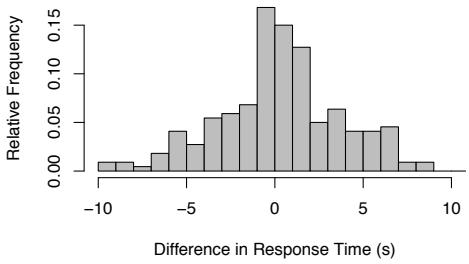
For our verification task, we have arrived at the classical trade-off between type 1 and type 2 errors, perhaps more often encountered in MIR when trying to optimise precision and recall: Because we found no significant interaction between parameter settings and honest play, choosing settings to make the game easier for honest players also will make it easier for cheaters. Conversely, the large benefit to playing honestly – again, a 64 percent improvement in the odds of answering the verification correctly – suggest that we may feel comfortable that players have an incentive to play honestly regardless of the parameter settings and thus can focus on making the game as pleasant for honest players as possible. As such, we intend to allow 15 s for recognition and use the −15-s distractor offset.

We were surprised that the distractor offset had such a strong effect on the players' accuracy, and the idea that distractors from the past are easier to identify as incorrect than distractors from the future is especially intriguing from a cognitive perspective: Is it easier to rewind musical memory than it is to fast-forward? Another possibility, perhaps simpler, is that the forward distractor is more likely to be in the same structural section as the original fragment, whereas because we have chosen our fragments always to start at the beginnings of new sections, the backward distractor will always be in a different one. Assuming that structural sections maintain some degree of timbral consistency, the backward distractor may more often offer timbral clues to the player that something is not right when the sound returns.

The data for the prediction task do not lend strong support our hypothesis that recognition time is a proxy for social intuitions about catchiness. This lack of support is especially surprising given that our concern had originally been more that players would somehow learn to choose fragments that optimised gameplay without touching on



(a) Choosing the First Fragment



(b) Choosing the Second Fragment

Figure 3: Distributions of response-time differences from the recognition task on pairs presented during prediction tasks. There are slight differences in the distribution of differences when the first member of the pair is chosen as opposed to the second, but overall, players do not appear to be consistent with their recognition behaviour when making predictions.

their personal feelings about catchiness; in fact, just the reverse seems to be true. Akin to Williamson and Müllensiefen's work on earworms and Burns's more speculative work, [4, 19], as we roll Hooked out to larger audience and thereby generate a larger database, we plan to find sets audio features that correlate with recognition and prediction performance. The difference between these two sets will help clarify this divergence between listeners' actual long-term musical memories and their expectations of them.

6. REFERENCES

[1] L. van Ahn and L. Dabbish: 'Designing Games with a Purpose', *Communications of the ACM*, Vol. 51, No. 8, pp. 58–67, 2008.

[2] L. Barrington, A. B. Chan, and G. Lanckriet: 'Modeling Music as Dynamic Texture', *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 18, No. 3, pp. 602–12, 2010.

[3] C. P. Beaman and T. I. Williams: 'Earworms ("Stuck Song Syndrome"): Towards a Natural History of Intrusive Thoughts', *British Journal of Psychology*, Vol. 101, No. 4, pp. 637–53, 2010.

[4] G. Burns: 'A Typology of "Hooks" in Popular Records', *Popular Music*, Vol. 6, No. 1, pp. 1–20, 1987.

[5] E. Coviello, A. B. Chan, and G. Lanckriet: 'Time Series Models for Semantic Music Annotation', *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 19, No. 5, pp. 1343–59, 2011.

[6] R. Dhanaraj and B. Logan: 'Automatic Prediction of Hit Songs', *Proc. 6th ISMIR*, pp. 488–91, London, England, 2005.

[7] A. R. Halpern and J. C. Bartlett: 'The Persistence of Musical Memories: A Descriptive Study of Earworms', *Music Perception*, Vol. 28, No. 4, pp. 425–32, 2011.

[8] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck: 'Temporal Pooling and Multiscale Learning for Automatic Annotation and Ranking of Music Audio', *Proc. 12th ISMIR*, pp. 729–34, Miami, FL, 2011.

[9] H. J. Honing: 'Lure(d) into Listening: The Potential of Cognition-Based Music Information Retrieval', *Empirical Musicology Review*, Vol. 5, No. 4, pp. 121–26, 2010.

[10] H. J. Honing and O. Ladinig: 'The Potential of the Internet for Music Perception Research: A Comment on Lab-Based Versus Web-Based Studies', *Empirical Musicology Review*, Vol. 3, No. 1, pp. 4–7, 2008.

[11] C. Kronengold: 'Accidents, Hooks and Theory', *Popular Music*, Vol. 24, No. 3, pp. 381–97, 2005.

[12] C. L. Krumhansl: 'Plink: "Thin Slices" of Music', *Music Perception*, Vol. 27, No. 5, pp. 337–54, 2010.

[13] E. Law, K. West, M. Mandel, M. Bay, and J. S. Downie: 'Evaluation of Algorithms Using Games: The Case of Music Tagging', *Proc. 11th ISMIR*, Utrecht, the Netherlands, 2010.

[14] D. J. Levitin and P. R. Cook: 'Memory for Musical Tempo: Additional Evidence That Auditory Memory Is Absolute', *Perception and Psychophysics*, Vol. 58, No. 6, pp. 927–35, 1996.

[15] P. Mercer-Taylor: 'Two-and-a-Half Centuries in the Life of a Hook', *Popular Music and Society*, Vol. 23, No. 2, pp. 1–15, 1999.

[16] F. Pachet: 'Hit Song Science', *Music Data Mining*, T. Li, M. Ogihara, and G. Tzanetakis, Eds., pp. 305–26, Chapman & Hall/CRC, Boca Raton, FL, 2012.

[17] D. Traut: '"Simply Irresistible": Recurring Accent Patterns as Hooks in Mainstream 1980s Music', *Popular Music*, Vol. 24, No. 1, pp. 57–77, 2005.

[18] J. Vandekerckhove and F. Tuerlinckx: 'Fitting the Ratcliff Diffusion Model to Experimental Data', *Psychonomic Bulletin and Review*, Vol. 14, No. 6, pp. 1011–26, 2007.

[19] V. J. Williamson and D. Müllensiefen: 'Earworms from Three Angles: Situational Antecedents, Personality Predisposition, and the Quest for a Musical Formula', *Proc. 12th ICMPC*, pp. 1124–32, Thessaloniki, Greece, 2012.

HIERARCHICAL CLASSIFICATION OF CARNATIC MUSIC FORMS

Ranjani. H. G.
Dept. of Electrical Communication Engineering,
Indian Institute of Science, Bangalore - 12, India
ranjanihg@ece.iisc.ernet.in

T. V. Sreenivas
Dept. of Electrical Communication Engineering,
Indian Institute of Science, Bangalore - 12, India
tvsree@ece.iisc.ernet.in

ABSTRACT

We address the problem of classifying a given piece of Carnatic art music into one of its several forms recognized pedagogically. We propose a hierarchical approach for classification of these forms as different combinations of rhythm, percussion and repetitive syllabic structures. The proposed 3-level hierarchy is based on various signal processing measures and classifiers. Features derived from short term energy contours, along with formant information are used to obtain discriminative features. The statistics of the features are used to design simple classifiers at each level of the hierarchy. The method is validated on a subset of IIT-M Carnatic concert music database, comprising of more than 20 hours of music. Using 10 s audio clips, we get an average f-ratio performance of 0.62 for the classification of the following six typesof Carnatic art music: */AlApana/, /viruttam/, /thillAna/, /krithi/, /thani-Avarthanam/* and */thAnam/*.

1. INTRODUCTION

Carnatic classical music is an aesthetic art form of South India. A typical Carnatic concert is rich in music content, and has structures at various levels. For example, use of various */rAga/* [1] (melodic framework) and */tALa/* (rhythmic framework) in a concert is well known [1]. Additionally, the concert can be lead-vocal or lead-instrumental in nature [2]. Apart from this, a typical concert comprises of a mix of various forms of Carnatic music, such as, */AlApana/, /thAnam/, /krithi/, /viruttam/, /thani-Avarthanam/, /thillAna/* and */swarakalpana/* [3]. All of these forms fall under any of two broad categories: */manOdharma/* (extempore) or */kalpita/* (those already composed) rendering [1, 4].

In this work, we explore different features from the music signal to classify the given piece of Carnatic vocal concert music in one of the several forms, i.e., */AlApana/*, */thAnam/*, */krithi/*, */viruttam/*, */thani-Avarthanam/* or */thillAna/*. This kind of classification will find application in segmenting a concert into different parts, as many students of music and connoisseurs (*/rasikA/*) want to concentrate on one of the parts for either learning or enjoyment. Similarly, people would like to access these different parts of a concert from a database of Carnatic music, again for specific listening or improvising.

To the best of our knowledge, there is little or no work to automatically classify a given piece of Carnatic music into its various forms or classes. There exists no single feature set that can classify all the classes, owing to the varying nature of the signals. In addition, as can be expected of music, a sub-set of the classes exhibit common overlapping structures, while the rest differ. A hierarchical approach can organize the classes to groups and attribute each such group with a discriminative feature set based on the domain-knowledge. We propose acoustic features to represent and distinguish each class at each hierarchy level.

The paper is organized as below: Details of the forms of music is elaborated in Section 2. In section 3, we present the proposed hierarchical structure. The feature set extracted and the design of classifiers are detailed in Section 4. Section 5 summarizes the experiments along with the results followed by discussion and conclusions in section 6.

2. CARNATIC MUSIC FORMS

Carnatic music has a long, rich and illustrious tradition and hence many forms of the music exist through the */guru-shishya parampara/* [1, 4]. Broadly, in a concert which lasts about 2-3 hours, some of the main forms are presented using construed musical ideas of the performer. These are sung (or played) in the context of different */rAga/* and different */sAhitya/* (lyrics) to create an enjoyable concert for the audience.

We expand on the details of some of the forms in a Carnatic music concert. These are, however, by no means exhaustive. One minute long example excerpts, for each of these forms, can be downloaded from:
http://www.ece.iisc.ernet.in/∼ranjanihg/audio/index.html

2.1 */AlApana/*

/AlApana/ is a purely extempore melodic form of music. The exposition of a */rAga/* (melodic framework of Indian classical music) with no rhythm is called the */AlApana/*. It comprises of a sequence of phrases sung intended to create a mood for the subsequent composition, i.e., */krithi/*. The performer chooses the phrases based on the grammar of

[1] All the Carnatic music terms are written in emphasized text within /./, to indicate phonetic pronounciation

the */rAga/* and elaborates systematically. The duration of an */AlApana/* can vary from 1 - 45 minutes and primarily depends on the taste and mood of the performer and also the subsequent composition. Typical vocal syllables used in */rAga AlApana/* are */ta/, /da/, /na/ and /ri/*.

In vocal concerts, the */AlApana/* comprises of extempore presentation of phrases by the lead artist, which would be pursued by the melodic-instrumental accompanist(s); occassionally this is followed by a solo */AlApana/* performance by the melodic-instrument accompanist(s) too.

2.2 */thAnam/*

A */thAnam/* is also a melodic improvisation form in Carnatic music. In its present day form, */thAnam/* is rendered as a part of */rAgam-thAnam-pallavi/* (or popularly RTP) of a concert. A */thAnam/* blends melody with medium-paced rhythm (*/laya/*) and brings about the intricacies of the */rAga/*. The syllables used are */aa/, /nam/, /tham/, /na/, /thom/, /tha/, /nom/*, such that the audience perceive the word '*/ananthам/*' (endless) and '*/Anandam/*' (happiness). Most often, */thAnam/* is rendered without percussion instruments.

2.3 */viruttam/*

A */viruttam/* is a devotional and metrical verse in one of the (Indian) languages, sung as improvised music. It generally is elaborated in a */rAga/* or a */rAgamAlika/* (where sequence of */rAgas/* used for each verse of a given composition). Though the verse has a metric structure, */viruttam/* is devoid of rhythm in the rendering and hence may not be identifiable in the song.

2.4 */thani-Avarthanam/*

In Carnatic concerts, */thani-Avarthanam/* are percussion solo performances which exhibit creative and technical skill. It, generally, follows the main performance in a concert. In presence of multiple percussionists, alternate performances followed by group is the convention. Through out the performance the tempo of the performance is maintained.

2.5 */thillAna/*

A */thillAna/* is a popular and energetic performance, originated for dance performance, and realized during the end of a music concert. A major part of this rhythmic performance, comprises of a limited set of beat synchronous melodic utterances of meaningless rhythmic syllables. Few example syllables are */dheem/ /ta/ /na/ /dhir/ , /tha/ /ki/ /ta/ /jham/*, and the likes. Another feature of this form is that the final verse of a '*/thillAna/*' consists of lyrics similar to a */krithi/*.

2.6 */krithi/*

A */krithi/* in Carnatic music is a pre-composed piece of music or */kalpita sangeeta/*. It comprises of sub-structures: */pallavi/*, */anupallavi/* and */charana/* which roughly correspond to refrain and verses in Western music. Some compositions also include */chittai swaras/*, which are made of only the Indian solfege syllables i.e.,(*/sa/ /ri/ /ga/ /ma/ /pa/ /da/ /ni/*). [2]

[2] Some more finer aspects to a */krithi/* such as */niraval/*, */varnam/*, */mangalam/*, */gIthe/* exist, which we have currently grouped into one class */krithi/*.

2.7 */swara kalpana/*

A */swara kalpana/* is also an improvised melodic performance with rhythm and the rendition uses only Indian solfege syllables. The performer must conform to the grammar of the chosen raga along with the rules of swarakalpana. This is an extempore performance or */manOdharma sangeeta/*, though for a naive listener, this form is very similar to the */chittai swara/* section of the */krithi/*, which is */kalpita sangeeta/*.

3. PROPOSED APPROACH

From the above discussed forms, it is clear that we have to examine both melodic properties as well as rhythmic aspects, in grouping the different Carnatic music forms. Further, we can exploit certain specific syllable vocalizations which have become well accepted conventions among singers. Because of such high level features of the music form, we cannot find one feature set which will permit us to do a single level multi-class classification. Additionally, features chosen for discriminating between a chosen set of classes, may have no impact on other classes, owing to the presence of simultaneous structures among the different classes. Instead, we resort to different feature set at different levels and propose a hierarchical approach to classify a given piece of Carnatic vocal music into one of the above forms (Section 2).

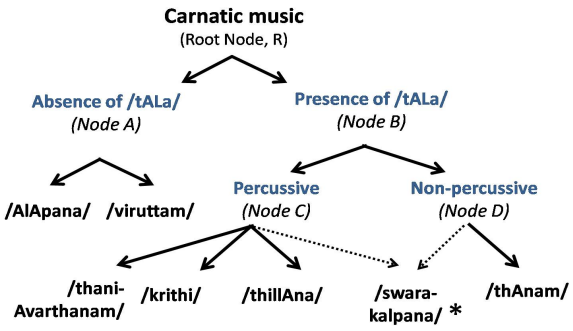


Figure 1. [Color online] Proposed hierarchical classification of Carnatic music forms. Text in blue indicates proposed hierarchy classification.

The classes are hiearchically divided as shown in Figure 1. At the root level (Node R), a grouping based on whether the form has a */tALa/* framework or not, leads */thAnam/*, */krithi/*, */thillAna/*, */swarakalpana/* and */thani-Avarthanam/* in the former (Node B) and */AlApana/* and */viruttam/* in the latter (Node A). The */tALa/*-based music pieces can be further classified based on whether they contain percussion instruments (in a concert) along with melody leading to a divide between */thillAna/*, */krithi/* & */thani-Avarthanam/* and */thAnam/*, seen as Nodes C and D in Figure 1. However, the presence of percussion for */swara-kalpana/* [3] is a choice of the percussion artist and hence can belong to either of the two nodes. At Node C, we can further introduce another branch based on the presence

[3] ,*It is not yet clear as to how to distinguish this from a possible */chittai swara/* and hence, we consider this as a part of */krithi/*, in this work.

of the instrument, and thus separate */thani-Avarthanam/* from the rest, if presence of percussion-only instrument is detected. This can be approached by incorporating instrument or instrument class identification [12]. However, the current approach intends to minimize the depth of the hierarchy so as to reduce propagation of errors [13].

4. FEATURE SETS AND CLASSIFIERS

We consider features relevant to each node in the hierarchy so as to emphasize the discriminative capabilities between the select classes.

4.1 Presence or absence of */tALa/*: (Root Node R)

4.1.1 Rhythmogram

Consider a t^{th} segment of a music signal $x(n)$ as, $s(t,m) = [x(n), n = (t-1)H_E + m], m = (1 : N_E)$, where N_E is the segment length and H_E is the segment hop. Let E_t be the short-time energy of the t^{th} segment of the music-signal, calculated as $E_t = \sum_{m=1}^{N_E} s^2(t,m)$. The window length N_E, corresponding to 32 ms, and a window hop H_E to 10 ms are considered, similar to speech analysis, motivated by the suitable frequency and time resolutions, respectively.

Consider the sequence, $[E_t,\ t = (1 : N)]$, which we further segment as: $E(r,\tau) = [E_t,\ t = (r-1)H_{ac} + \tau],\ \tau = (1 : N_{ac})$, where N_{ac} is the energy segment length and H_{ac} is the energy segment hop. We refer to the short-time energy signal as the 'novelty signal'. The parameters H_{ac} and N_{ac} must be chosen to suit the slowest tempo in the music. We have chosen, N_{ac} to correspond to 10 s and H_{ac} to 50 ms. We, then, compute the time-varying autocorrelation function of $E(r,\tau)$ given as: $R_{EE}(r,k) = \sum_{\tau=1}^{N_{ac}-k} E(r,\tau)E(r,\tau+k)$, and, $k = 0 : (N_{ac}-1)$. This is the autocorrelation of the r^{th} novelty signal segment for the k^{th} lag. Figure 2 shows this novelty-autocorrelation function as it evolves over time.

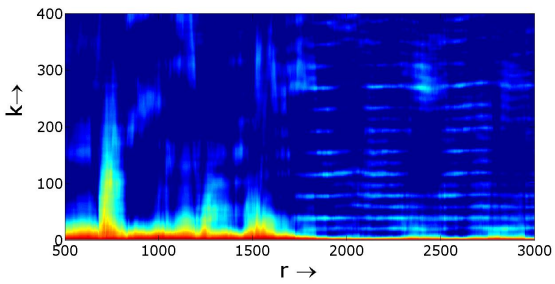


Figure 2. [Color online] Plot of $|R_{EE}(r,k)|$, the novelty-autocorrelation function for an audio clip consisting of */alapana/* followed by */krithi/*, as a 3D plot. Color=$|R_{EE}(,)|$.

The peaks in autocorrelation lags capture the long term periodicity of the energy signal, and are indicative of the metric structure of the */tALa/* [4]. This has been refered to as the rhythmogram [5]. (The method described above differs from [5], in taking autocorrelation of short-time energy in former, while use of autocorrelation of onsets is considered in [5].) The rhythmogram can be made discrete by picking P peaks of the rhythmogram for every r^{th} segment of the energy signal. We refer to this as thresholded binary-rhythmogram, denoted as $\mathcal{R}_b(r,\{l_i\})$ where $\{l_i\}$ indicates the i^{th} peak location for $i = [1 : P]$. */tALa/* structure, if present, can be well observed using this as a feature as shown in Figure 3.

In the Indian music context, tempo is a choice of the singer/ performer apart from being related to the composition. [5] Thus, for Node-R classification, we only need a measure to check the presence of */tALa/*; hence, we can use simple features (unlike in the literature which is mainly for Western music [8, 9] and uses the tempo to aid in classification or segmentation).

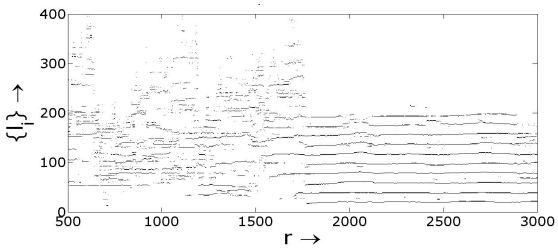


Figure 3. Plot of $\mathcal{R}_b(r,\{l_i\})$ for the audio clip corresponding to Figure 2, for $P = 10$

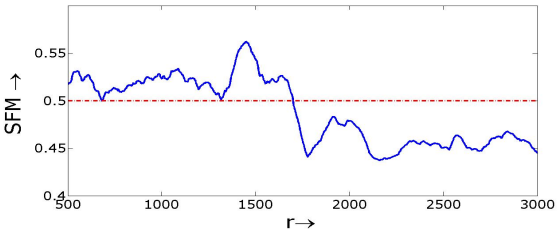


Figure 4. Plot of variation of Spectral Flatness Measure for the data corresponding to Figure 3, for $P = 10$

4.1.2 Classification at Root Node R

We can see from Figure 3 that the peak locations, $\{l_i\}$, show consistency in regions of music corresponding to presence of */tALa/* as against those which do not have */tALa/*. By constructing a histogram of the peak locations, we can expect peaky distributions, $P_r(k)$ for regions containing */tALa/*, while the non-*/tALa/* sections would have wider spread in the histogram. Motivated by this, we use spectral flatness measure (SFM or the GM:AM ratio) of the histogram of $\mathcal{R}_b(r,\{l_i\})$ which can be represented as:

$$P_r(k) = \frac{\text{\# of peaks with } l_i = k, \text{over } r = 1 : R}{\text{Total \# of peaks at all } k}$$

$$SFM(r) = \frac{[\prod_{k=1}^{K} P_r(k)]^{\frac{1}{K}}}{\frac{1}{K}\sum_{k=1}^{K} P_r(k)}, \text{ and, } K \triangleq N_{ac} - 1$$

The evolution of SFM across time for the music clip considered in Figure 3 is shown in Figure 4. We propose a low SFM as a score to indicate the presence of */tALa/*, with 0.5 value denoting boundary between the two classes. The effectiveness of the estimated distribution of peaks can

[4] It can be seen that slower the tempo, the peaks in $|R_{EE}(r,k)|$ will be farther apart.

[5] It can be argued that picking P peaks makes the feature invariant to tempo w.r.t further processing

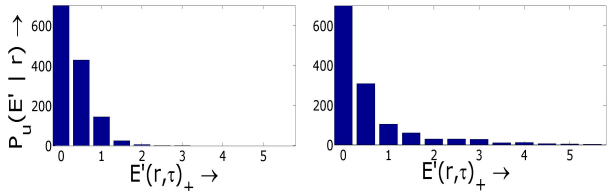
be increased by considering more points from $\mathcal{R}_b(r, \{l_i\})$. Hence, a trade-off between performance and duration, R can be expected. A value corresponding to 10 s is chosen for R.

4.2 Percussive Vs Non-percussive: (Node B)

4.2.1 Onset strength

Percussion instruments give rise to stronger onsets, than melody based instruments, owing to their sharp attack pattern [6, 7]. The method to calculate onsets is briefed as follows: The energy signal, E_t, used for rhythmogram earlier is again considered. The positive first order difference function, $E'(r,\tau)_+ = (E(r,\tau) - E(r,\tau-1))_+$ [6] is computed; depending on the instrument and its corresponding attack pattern, this difference will be large (for percussive) or small (for non-percussive). The positive first order differences can be an estimate of onset strength at the r^{th} frame, or, $E'(r,\ \tau)_+ \in \mathbb{R}^+$.

Let $P_u(E'_+|r)$ denote histogram of $E'(r,\tau)_+$, (onset strength) for a 10 s audio clip containing solo voice (singing with */tALa/*) and solo voice with percussion is shown in Figure 5. It can be seen that these distributions can be modeled using an exponential distribution with a significantly different parameter, λ.



(a) (b)

Figure 5. Histograms $P_u(E' \mid r)$ for (a) soft onsets and (b) hard onsets of */tALa/* class of data from randomly chosen 10 s segments of non-percussive and percussive kind.

Estimating the parameter of the exponential distribution, using the maximum likelihood framework, we get: $\lambda_+(r) = \frac{N_{os}}{\sum_\tau E'(r,\tau)_+}$, where N_{os} is the number of samples in the segment considered. Figure 6 shows a sample plot of variation of $\lambda_+(r)$ over successive r segments, for an audio concert clip reflecting the change from */thAnam/* to */krithi/*. The parameter is calculated for every 0.1 s, for a window size N_{os} corresponding to N_{ac}, ie., 10 s.

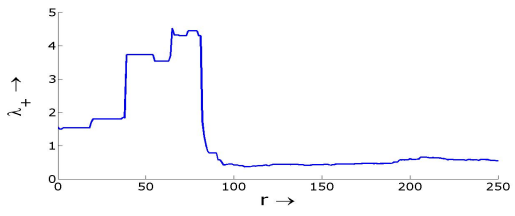


Figure 6. $\lambda_+(r)$ variation as a function of data segment, estimated for an audio clip transiting from */thanAm/* to */krithi/*.

A similar approach is seen in [10], where a Gaussian distribution is used to check the presence of onsets. Our approach differs not only in using exponential distribution, but also in using it for discriminating between soft and hard onsets.

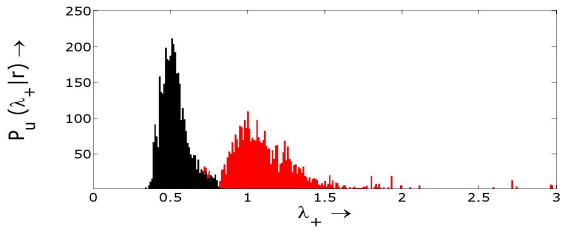


Figure 7. [Color Online] Histograms of, $P_u(\lambda_+ \mid r)$, depicting parameter distributions for 2 audio files representing (red) non-percussive and (black) percussive */tALa/* class.

4.2.2 Classification at Node B

From Section 4.2.1, estimated λ_+ parameter is seen to be effective in discriminating between forms of music (containing */tALa/*) that contain percussive and non-percussive instruments. Characterizing the same, across varied performances, shows that λ_+ distribution for percussive instruments is as shown in black and in red for non-percussive, shown in Figure 7. The deviation from the prototype parameters characterized by mean and variance of the distributions, provides correctness scores of classification.

4.3 */AlApana/* Vs */viruttam/*: (leaf nodes of A)

4.3.1 Formant peaks as features

The differentiator between */AlApana/* and */viruttam/* music pieces is that, the former uses limited syllables compared to the latter. */viruttam/*, which gives more importance to articulation, so that the lyrics are heard clearly. $(F1, F2)$ are estimated using Praat software [11] (Burg's method) [7], using a segment length $N_f = 32\ ms$, and segment hop $H_f = 10\ ms$. Figure 8 depicts a histogram of $(\frac{F2}{F1})$ of both the classes. It can be observed that there is lesser $(\frac{F2}{F1})$ variation in the alapana signal. We can thus use the rate of movement of $(\frac{F2}{F1})$ as a discriminative feature set in the rhythm-less form of Carnatic music.

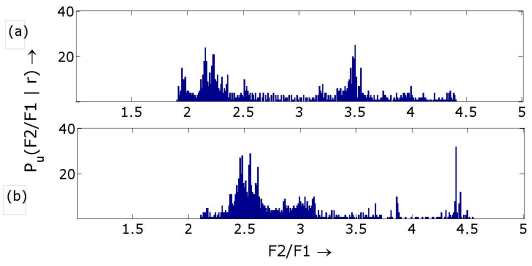


Figure 8. Histogram of $(\frac{F2}{F1})$ for a 10 s of audio containing (a) */viruttam/* (b) */alapana/* for a 10 s second clips from randomly chosen example clips of the classes.

4.3.2 Classification for /AlApana/ and /viruttam/

Section 4.3.1 illustrated the effectiveness of $(F1, F2)$ to discriminate between */AlApana/* and */viruttam/*. It is observed that *AlApana* is more likely to have realizations of

[6] We define a function, $(x)_+ \triangleq \max(0, x)$

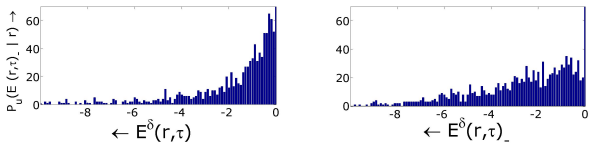
[7] Complexity in estimating reliable formants for high pitched sounds is well known and is beyond the scope of current paper.

/a/, yielding a choice $(\frac{F2}{F1})_a \sim \mathcal{N}(\mu_a, \sigma_a)$, with parameters $(\mu_a, \sigma_a) = (2.5, 0.3)$, with suitable Bayes' decision boundary. Thus, classification is based on maximum likelihood, where the likelihood is calculated for the ratio of mean of the formants extracted from 10 s data.

4.4 */thani-Avarthanam/ Vs /thillAna/ Vs /krithi/* (leaf nodes of C)

4.4.1 Cessation strength as feature

Between */krithi/* and */thillAna/*, from Sections 2.5, 2.6 and audio clips, one can prominently observe that */thillAna/*, inspite of presence of vocals, give a stronger perception of beat cycle than in the case of a */krithi/*. On closer observation, one can attribute it to the regular cessation of energy in */thillAna/*, owing to the syllables used (which mainly comprise of prominent stops as onsets of the syllables). This cessation of energy can be gauged as 'offset'. We observe that the strength of this offset is higher for */thillAna/* than for */krithi/*. Using terminology similar to Section 4.2.1, we can express $E^{\delta}(r,\tau) = (E(r,\tau) - E(r,\tau-\delta))_-$ [8], where $E^{\delta}(r,\tau) \in \mathbb{R}^-$. The parameter δ, is required as the cessation cannot be instantaneous, requiring δ duration to effect a noticable change; or in other words, it is a gradual offset. A sample histogram depicting this is shown in Figure 9. Hence, we can use the exponential parameter, λ_-, as a parameter to differentiate between */krithi/* and */thillAna/*, which consist of sustained energy owing to vocal or instrumental content.



(a) (b)

Figure 9. Histogram $P_u(E^{\delta} \mid r)$ of $E^{\delta}(r,\tau)$ for a 10 s of audio containing (a) */krithi/* (b) */thillAna/*

4.4.2 Dynamic range of the energy contours

The spectral content of */thani-Avarthanam/*, the percussion-only performance, comprises of mainly short lived bursts of energy, or "vertical lines" in the spectrogram. A periodic and large variation in the intensity as a function of time can be observed due to strong onsets and offsets, and no sustaining period, resulting in lower average intensity of the audio signal. Figure 10, depicts the distribution of average intensity in 10 s clips of */thani-Avarthnam/* and */thillAna/*, which can be used as a feature to distinguish these classes.

4.4.3 Classification combining the offset and dynamic range

Similar to Section 4.2.1, estimated λ_- parameter can discriminate between */thani-Avarthanam/* and */thillAna/* against */krithi/*. Discrimination between */thani-Avarthanam/* and */thillAna/* uses mean energy, $\mu(E)$, as a feature.

[8] We define, $(x)_- \triangleq \min(0, x)$

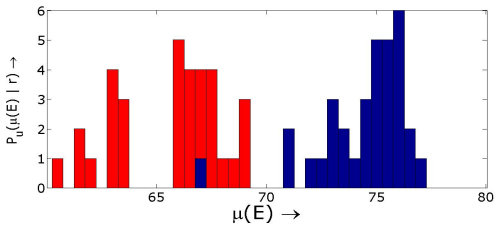


Figure 10. Histogram of average intensity, $\mu(E)$, in a 10 s of audio containing (blue) */thillAna/* and (red) */thani-Avarthanam/*

4.5 */thanam/* (leaf node of D)

For the present, */thanam/* is the only class considered at node D and hence on classifying a particular piece to contain */tALa/* without percussion at Node B, leads to this class in the proposed schema.

5. EXPERIMENTS AND RESULTS

All experiments are carried out on excerpts of vocal Carnatic music concerts. The database used comprises of randomly chosen 100 files from IITM concert database [14], amounting to 12 vocal concerts, by a total of 9 artistes (6 male and 3 female artists), and a total of ∼20 hours duration. The tonic frequencies ranges from 129 to 205 Hz. The ground truth is generated after manually listening to the music pieces. The number of occurences and its duration, of each of the Carnatic form in the database, is detailed in Table 1. Also indicated are the number of non-overlapping 10 s clips of each of the music form, which is used for evaluating the algorithm. The inherent skewness in class distributions is to be noted. Change from one form to another in a Carnatic music concert is customary, and hence, any given 10 s clip can comprise of at most 2 classes. In such cases, the considered ground truth gives weightage to the temporally dominant class.

The results of the classification at each node is shown in Table 2. The performance of the proposed hierarchical model is evaluated at each node of the hierarchy, taking into account the performance of the parent nodes. However, errors at all nodes are given equal weightage, similar to "flat classification" evaluation [13]. The performance of the algorithm for each 10 s audio data is measured using the f-ratio [9]. The f-ratio is only indicative of the predictability of the algorithm to a particular class, and is invariant to tn, which is required in the unbalanced distribution of multi-class. Hence, we consider accuracy, $A = \frac{tp+tn}{tp+tn+fp+fn}$ also as a performance metric, which is not invariant to tn [15].

Though the performance of the proposed approach depends on the size of audio data analysed, we report results for 10 s clip only. An average f-ratio of 0.62 is seen for the proposed hierarchical classification, which is quite promising, given the complexity of the high level structures in the

[9] f-ratio is $F \triangleq \frac{tp}{tp+(\frac{fp+fn}{2})}$ while precision is $P \triangleq \frac{tp}{tp+fp}$. Recall, $R \triangleq \frac{tp}{tp+fn}$, & tp, fp, fn indicating the number of true positive, false positive & false negatives respectively.

music signal being examined. Also, the performance was observed to be not affected in terms of gender of the singer or the tonic variations, (at nodes R, B, C and D) as feature set considered is mostly independent of pitch.

Detailed analysis shows that */tAnam/*, */AlApana/* and */viruttam/* are the most confused classes. This is because the fp (of */tALa/* class) at Node R, trickle down to leaf node D always. This can be expected, as leaf nodes of A always have 'soft' onsets, thus reducing the predictability score of */thAnam/*. Similarly, fn errors (of */tALa/* class, mostly comprising of */tAnam/* class), and have higher syllabic repetitive features similar to that of */AlApana/*. Also, it is seen that */thani-Avarthanam/*, */thillAna/* and */krithi/* display better performance inspite of being a node below in the hierarchy. */krithi/* shows least recall performance which has been observed to be due to assumption of */chittaiswaras/* and */swara-kalpana/* in the */krithi/* class, resulting in misclassification to */thillAna/* class. Also, the performance at each node is sensitive to choice of decision boundary. Hence, it is crucial to have additional discriminative features along with high performance classifiers at each node of the hierarchy. The hierarchical framework is proposed for vocal concerts, and a generalizable framework across vocal and instrumental concerts is to be contemplated.

Table 1. Details of Carnatic music forms in the chosen dataset

Carnatic Form	Duration(min)	# of segments
/AlApana/	373	2236
/viruttam/	50	302
/thillAna/	23	136
/krithi/	666	3996
/thani-Avarthanam/	66	396
/thAnam/	22	132

Table 2. Performance of the algorithm on the Carnatic music forms

Carnatic Form	Precision	Recall	f-ratio	Accuracy
Presence of */tALa/*	0.89	0.97	0.92	0.88
Percussive	0.94	0.86	0.88	0.85
/viruttam/	0.80	0.60	0.63	0.80
/AlApana/	0.45	0.79	0.47	0.67
/thillAna/	0.87	0.65	0.7	0.74
/krithi/	0.93	0.5	0.72	0.71
/thani-Avarthanam/	0.95	0.7	0.75	0.94
/thAnam/	0.4	0.68	0.50	0.81

6. CONCLUSIONS

We have proposed a hierarchical approach to classification of Carnatic music forms, using signal processing features derived from energy contours, formant movements and parametric statistic distribution of such features. Based on the parameters of the statistical distributions, simple classifiers are proposed. An initial set of feature-vectors are designed, considering the "production model" of each particular class. An average f-ratio of 0.62 and accuracy of 0.77 is obtained, thus reflecting the promising nature of the feature vectors. The hierarchical approach to group the classes, based on the knowledge of the data is intuitive, and helps deriving better feature vectors for within the group classification. Hence, the trade-off between reduced accuracy with increasing depth of the hierarchy, against complex features for a flat single level classification can be observed as expected. More sophisticated classifiers in an automatic supervised framework can be developed based on these features to get better performance.

7. REFERENCES

[1] B. Benary: "Composers and Tradition in Karnatic Music," *Asian Music,* Vol. 3(2), pp 42-51, 1972.

[2] K. L. Armand and A. L. Armand: "One Hundred Years of Music in Madras: A Case Study in Secondary Urbanization," *Ethnomusicology,* Vol. 27(3), pp. 411-438, 1983.

[3] R. Morris: "Variation and Process in South Indian Music: Some Kritis and their Sangati," *Music Theory Spectrum* Vol. 23(1), pp. 74–89, 2001.

[4] Indira V Peterson: "The 'KRTI' as an integrative cultural form: Esthetic experience in the religious songs of two south Indian classical composers," *Journal of South Asian Literature*, Vol. 19(2),pp 165–179, 1984.

[5] K. Jensen: "A Causal Rhythm Grouping," *Computer Music Modeling and Retrieval*, Vol. 3310, pp 83–95, 2005.

[6] Moore, B. C. J. and Glasberg, B. R. and Baer, T.: "A model for the prediction of thresholds, loudness, and partial loudness," *Journal of the Audio Engineering Society*,Vol. 45.4, pp 224–240, 1997.

[7] J. P. Bello, L. Daudet et al.,: "A Tutorial on Onset Detection in Music Signals," *IEEE Trans. Speech and Audio Processing*, Vol 13.5, pp 1035–1047, 2005

[8] D. W. Ellis: "Beat Tracking by Dynamic Programming", *Journal of New Music Research*, pp 51–60 ,2007.

[9] G. Tzanetakis and P. Cook: "Musical Genre Classification of Audio Signals," *IEEE Trans., Speech and Audio Processing*, Vol. 10(5), pp 293–302, 2002.

[10] C. Duxbury, M. Sandler and M. Davies: "A Hybrid Approach To Musical Note Onset Detection", *Proc. of 5th Int. Conference on Digital Audio Effects (DAFx)*, pp 33–38, 2002.

[11] Boersma, P. and Weenink, D: "Praat: doing phonetics by computer [Computer program]," Version 5.3, 2013.

[12] G. Grindlay and D.P.W. Ellis: "Transcribing Multi-instrument Polyphonic Music with Hierarchical Eigeninstruments" *IEEE Journal Of Selected Topics In Signal Processing*, Vol. 5 (6), pp 1159–1169, 2011.

[13] E. P. Costa, A. C. Lorena et.al,: "A Review of Performance Evaluation Measures for Hierarchical Classifiers," *Proc. of the Association for the Advancement of Artificial Intelligence*, pp 1-6, 2007.

[14] A. Bellur, V. Ishwar and H. Murthy: "A knowledge based Signal Processing Approach to Tonic Identification in Indian Classical Music," *Proc. 2nd CompMusic Workshop*, pp 113–118, 2012.

[15] M. Sokolova, G. Lapalme: "A systematic analysis of performance measures for classification tasks," *Proc Information Processing and Management*, Vol 45, pp 427–437, 2009.

A SIMPLE FUSION METHOD OF STATE AND SEQUENCE SEGMENTATION FOR MUSIC STRUCTURE DISCOVERY

Florian Kaiser
STMS IRCAM-CNRS-UPMC
1 Place Igor Stravinsky
75004 Paris
florian.kaiser@ircam.fr

Geoffroy Peeters
STMS IRCAM-CNRS-UPMC
1 Place Igor Stravinsky
75004 Paris
geoffroy.peeters@ircam.fr

ABSTRACT

Methods for music structure segmentation are based on strong assumptions on the acoustical properties of structural segments. These assumptions relate to the novelty, homogeneity, repetition and/or regularity of the content. Each of these assumptions provide a different perspective on the music piece. These assumptions are however often considered separately in the methods. In this paper we propose a method for estimating the music structure segmentation based on the fusion of the novelty and repetition assumptions. This combination of different perspectives on the music pieces allows to generate more coherent acoustic segments and strongly improves the final music structure segmentation's performance.

1. INTRODUCTION

Music structure segmentation (MSS) is the task of dividing a musical audio signal into its main structural parts. Examples of such main segments for popular music are the verse and the chorus. MSS allows for a large set of applications of interest in the context of digital music, such as automatic summarization or active listening. Because of this, the task emerged as an important challenge for the Music Information Retrieval (MIR) research and industrial communities.

A musical composition is a layered construction of quantifiable musical elements of various temporal scales, e.g. beats, notes, bars, etc. While these elements are qualified by strict musical definition, the higher temporal level music structure is perceptually audible but not qualified by any strict musical definition. Because of this, the MSS task raised questions about the definitions of the segments to be estimated. In order to cope with this lack of definition, MSS researchers have developed assumptions-driven methods. As described by Paulus et al. [13] methods can be categorized according to the used assumptions on the content: novelty, homogeneity and repetition. We can add to this list the Regularity hypothesis proposed by Sargent et al. [15].

1.1 Related Work

MSS algorithms usually rely on two successive steps: segmentation and grouping of segments. The temporal segmentation consists in estimating the borders of potential structural segments, therefore limiting the search space for the structural and fixing its temporal scale. This step is crucial to ensure the global performance of systems. This preliminary temporal segmentation is directly influenced and constrained by the above-mentioned assumptions. This is because the various assumptions give different perspectives of the music pieces content. We briefly review these assumptions here.

With the novelty hypothesis, boundaries between structural segments are considered as time points of high "acoustical contrast". This notion of contrast has been introduced by Foote [5] and extends previously proposed audio novelty segmentation techniques [3]. It considers jointly the homogeneity within segments as well as the dissimilarity between segments. A novelty function is computed by convoluting a Self-Similarity Matrix (SSM) with a kernel that reflects the novelty hypothesis. Peaks of the novelty function define potential structural boundaries. This method has been successfully applied to music structure segmentation and still produces state-of-the-art performances for the temporal segmentation step [4] [12] [9]. Slightly different, the homogeneity assumption only require strong inner acoustical homogeneity of the structural segments. A popular approach then consists in defining the structural segments as the states of a Hidden Markov Model (HMM) [1] [14] [11].

The novelty and homogeneity assumptions assume strong inner-homogeneity of segments, a property that Peeters formalized in [14] as the "state" representation of structural segments, and that is very often related to timbral properties of the music pieces. In contrast, repetition-based temporal segmentation aims at detecting repeated segments, homogeneous or not. In the case of non-homogeneity, repeated segments are visualized in a SSM as stripes on the diagonal and off-diagonals. The segmentation of these stripes is denoted by Peeters as the sequence approach and is usually done in a Time-Lag Matrix [6].

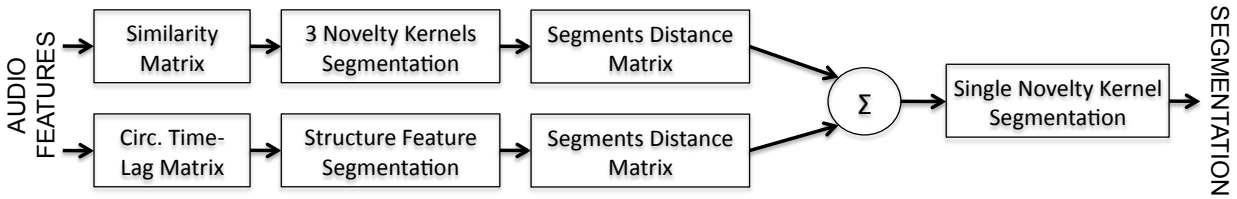


Figure 1. Fusion strategy between novelty and repetition temporal segmentation

Recently, an extension of this lag representation was proposed by Serra et al. [16] and allow for the detection of non-homogeneous as well as homogeneous repeated segments. This method is further described in the latter of this paper.

1.2 Paper overview

In this paper we study the combination of these different temporal segmentation approaches. In particular, we propose a generic fusion method that we apply to the fusion of the repetition and novelty-based approaches. The choice of these two methods is motivated by the strong antagonism that exists between them. Indeed, the detection of repetitions detection is global since it requires considering the whole signal (usually represented through a lag matrix). In contrast, the novelty approach is essentially local and has no global perspective on the music piece.

These two methods thus give different explanations of the music pieces' structure that are both true. Our claim is that these explanations are complementary and that their fusion enhances the global music structure segmentation performance. To this end, we propose a late fusion approach that maintains the acoustic coherency within the final estimated segments. Segments are first estimated under both hypotheses separately. A segments distance matrix is then computed for both segmentations. The sum of these two matrices then serves as the final representation for the temporal segmentation.

1.3 Paper organization

In Section 2 we introduce the algorithms we use for the temporal segmentation under both assumptions. In Section 3 we present the fusion procedure of the segmentations estimated with these two methods. The method is then illustrated in Section 4 on a real signal. In Section 5, we propose a comparative evaluation. Finally, conclusions are drawn in Section 6. The general architecture of the fusion method is illustrated in Figure 1.

2. TEMPORAL SEGMENTATION ALGORITHMS

2.1 Repetition-based Segmentation

The classical approach for repetition-based segmentation relies on the computation of a time-lag matrix representing chroma similarity. Recently, Serra et al. [16] proposed a novel method that extends this approach to a circular time-lag matrix representation. The latter incorporates both past and future samples. The audio signal is therefore represented by a multidimensional time series that contains, for each time frame instant, the chroma vector of the actual frame as well as knowledge of the recent past with the encapsulation of delay coordinates [10]. A recurrence plot R retaining only the nearest similar frames is then computed on this multidimensional time series. Circular shifting of the rows of R then allows to compute the circular time-lag matrix L:

$$L_{i,j} = R_{i,k+1} \tag{1}$$

with N the size of the feature vector, i = 1, ... , N and k = i + j -2 mod(N). An example of such a matrix is shown in Figure 4 (a). After smoothing with a bivariate gaussian kernel, the authors define the so-called "structure feature" that serves for the temporal segmentation as the rows of L. Structural changes are indeed detected by strong changes in the structure feature sequence and can be estimated in the difference between adjacent structure feature vectors. Evaluation at the 2012 MIREX [1] evaluation campaign for structural segmentation showed very convincing performances of this method.

2.2 Novelty-based Segmentation

The novelty-based segmentation was originally proposed by Foote in [5] and allows to detect transitions between homogenous segments of a musical signal. This is achieved by means of the correlation of 2×2 checkerboard novelty kernel along with the main diagonal of a SSM. [8] extends this method by introducing two new novelty kernels that allow for the detection of non-homogeneous to homogeneous segments transitions and vice versa. These kernels are illustrated in Figure 2. Three novelty curves are computed for all three kernels on a SSM computed on timbre-related features (MFCCs, Spectral Centroid, Spread, Skewness and Spectral Flatness). Adaptive peak tracking technique described in [7] allows for the estimation of boundary candidates in the three novelty functions and a final segmentation is obtained by merging the three boundary sets within a tolerance range of 2s. In this paper, we use the novelty method extended by [8].

2.3 Segmentations Agreement

In order to highlight the differences between the two assumptions, we compare the segmentation results obtained

[1] http://www.music-ir.org/mirex/wiki/2012:MIREX2012_Results

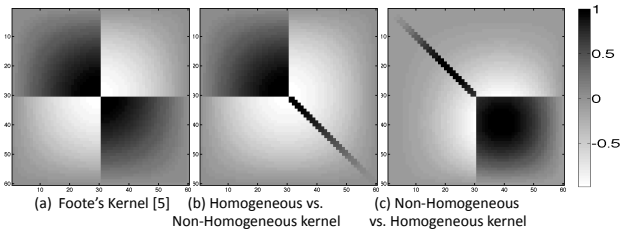


Figure 2. 60×60 Novelty detection kernels

with the repetition-based and novelty-based method presented above. We compare the results obtained on the Isophonics testset that consists of 297 popular music song. If we denote by T_R (T_N) one of the segment boundary obtained with the repetition-based segmentation (novelty-based segmentation), the boundaries T_R and T_N are said equivalent if within a tolerance window of 2 seconds.

Results show that that 45.8% of the T_N were contained in the T_R. Moreover, 55.9% of the T_R were contained in the T_N. There is thus about 50% of the information estimated by the two methods that is very specific to the chosen assumption. While the two methods achieve rather comparative results when evaluated within MIREX, this experiment strengthens the assumption that fusion of both assumptions may increase the performance of the temporal segmentation.

3. SEGMENTATIONS FUSION

Repetition- and novelty- based segmentations are explicitly designed for different representations of the audio content. An early fusion approach of these representations would be therefore irrelevant. Instead, we choose a late fusion approach: the segmentations using both assumptions are first estimated, and then merged together. In the remaining of this section we propose two fusion strategies. We first introduce a baseline method (see part 3.1) that is the simplest way to merge the boundaries. We then present (see part 3.2) a method that merges the boundaries by explicitly considering the acoustical relevancy of the fused segments. This is done by using a segments distance matrix representation.

3.1 Baseline Method

The baseline fusion of boundaries simply consists in merging T_R and T_N if they are within a given tolerance window Δ. Since identical boundaries may be detected in both sets with a slight temporal deviation, the simple union of boundary sets is not precise enough for the fusion. We therefore merge boundaries within a tolerance window. As illustrated in Figure 3 we retain the earliest boundary of the two when a matching is found.

In our experiment, we set the tolerance window at $\Delta = 2$ seconds (1 measure @120bpm) to limit over-segmentation.

3.2 Segments Distance Based Fusion

Because the baseline method is only constrained by the heuristic rule of a tolerance range, it does not consider

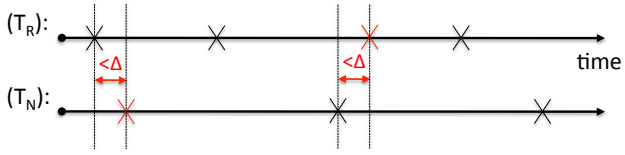


Figure 3. Illustration of the baseline fusion method.

the acoustic relevance of the newly formed segment that derived from excluding or keeping a boundary. The new segmentation might thus produce irrelevant segments and induce errors at the structure clustering step. We therefore propose here an alternative approach for the fusion of boundaries that includes knowledge of the acoustical coherence of the final estimated segments.

3.2.1 Segments distance matrix

We introduce the Segments Distance Matrix (SDM) that measures the acoustical consistency between segments formed by T_R and T_N. In this, the distance between two segments is calculated using the Mahalanobis distance between the features distributions within each pair of segments:

$$S(i,j) = d(X_i, X_j) = \sqrt{(X_i - X_j)^T \Sigma^{-1} (X_i - X_j)} \quad (2)$$

with X_i [m×n] and X_j [p×n] the feature vectors respectively within segments i and j and Σ their covariance matrix. We use the Mahalanobis distance since it provides a good compromise between the homogeneity and repetition assumptions for the segments comparison.

The size of the SDM is defined by the number of boundaries detected. We then temporally-scale this SDM to the original SSM size in order to reflect the music piece's structure. Two examples of temporally-scaled-SDMs are shown in Figure 4 (c) and (c') with the song One Vision by Queen. This example illustrates that SDMs give a perspective on the saliency of each boundary. Indeed, we easily distinguish adjacent segments with very strong acoustical dissimilarity as well as strong similarity.

3.2.2 Fusion and Boundary Selection

The fusion process should select the boundaries by taking into account the acoustic saliency of the newly formed segments. To provide a perspective on the acoustic contrasts given by the fusion of boundaries, we sum the SDMs computed corresponding to T_R and T_N. This summation allows displaying in a single representation the acoustic contrast brought by both assumptions. It gives an acoustical perspective on potentially merged boundaries. This is illustrated in Figure 4 (d). In this summed SDM, the acoustic contrasts of both segmentations are kept. Note that a similar fusion of different matrix data representations was used by Chen in [2] for structure labelling purposes. The summed SDM can be thought as a simplified SSM like representation of the music piece that gives a global acoustic description of the acoustic content.

The final temporal segmentation is then obtained by applying the novelty segmentation on this fused representation. Since the matrix describes only segments of strong inner-homogeneity we solely employ Foote's kernel illustrated in Figure 2 (a).

4. CASE STUDY

In this section we illustrate on a real signal (the song "One Vision" by Queen) our method for segment detection based on late-fusion of repetition and novely-based segmentation. Figures 4 display - for the repetition method: the Circular Time-Lag Matrix (a), structure feature (b) and SDM (c) - for the novelty method: the SSM (a), novelty curve (b) and SDM (c). The SDMs calculated for both methods illustrate the different perspectives given on the song's temporal segmentation.

The sum of the SDMs and corresponding novelty curve with final estimated boundaries are displayed in Figure 4 (d) and (e). This clearly shows the compromise that is made in our method between repetition-based and novelty-based segments. Indeed, the different acoustic contrasts within the two segmentations can be corroborated by the final segmentation of the summed SDM but this not necessarily happening. For example boundaries that were detected within frames 180 and 420 by the repetition-based method are not all contained in the final segmentation because of insufficient acoustic contrast of the newly formed segment. Hence, the fusion method uses consistent acoustic clues to decide of the fusion of segmentations.

5. EVALUATION

We evaluate comparatively the performances of the various segmentation methods taken separately (repetition and novelty-based) and the proposed fusion methods (by baseline or distance-based fusion) as proposed in this paper. In order to investigate the impact of the method on the structure labelling of segments, an evaluation of the segments labelling is also proposed. We first introduce the segment labelling process, evaluation protocol and then present and discuss the results.

5.1 Segments Labeling

For all segmentation methods studied, the labelling of the segments is achieved using the method proposed by [9], i.e. a hierarchical clustering is applied on the basis vectors of the Non-Negative-Matrix-Factorization (NMF) of the SSM. We improve this method here by estimating automatically the optimal number k of clusters (hence of different segment labels) to be formed. For this, we use a method inspired by [17]. The method consists in varying the number k of clusters to be formed, and for each number, to compute the dispersion of the obtained partition. The dispersion D_k is defined as the average distance d_{xx} between all n_i elements x, x within each cluster C_i:

$$D_k = \sum_{i=1}^{k} \frac{1}{n_i} \sum_{x,x' \in C_i} d_{xx'} \qquad (3)$$

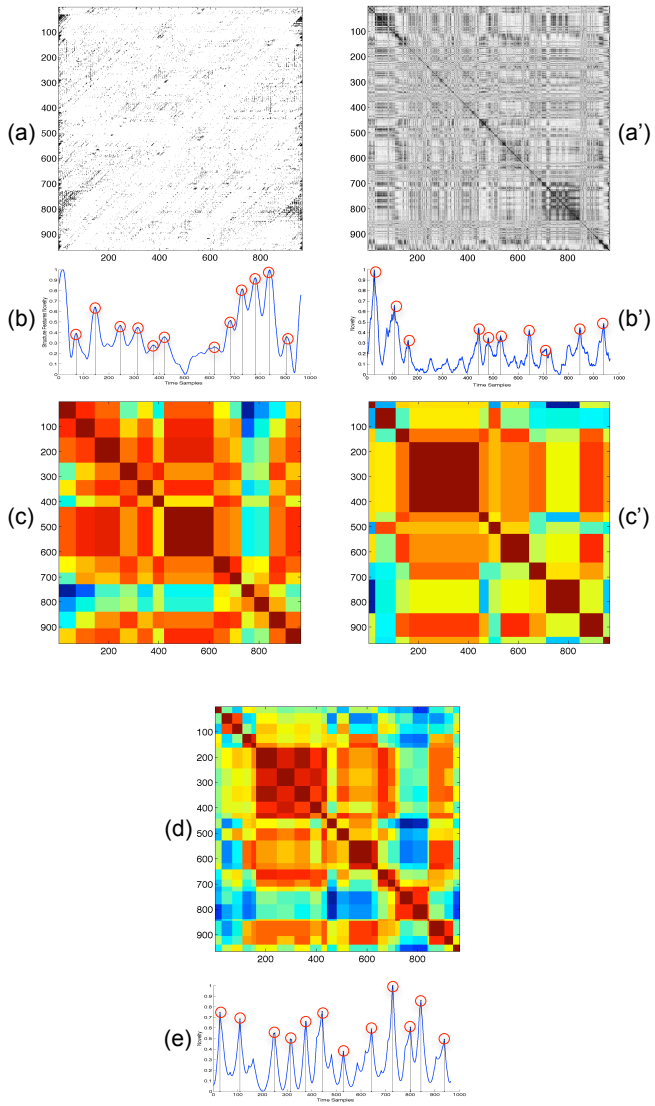


Figure 4. Example of the fusion method with the song example "One Vision" by Queen. (a) Circular Time Lag Matrix - (a') Timbre-related SSM - (b) Structure feature with estimated boundaries - (b') Novelty curve of the SSM with estimated boundaries - (c) Repetition segments SDM - (c') Novelty segments SDM - (d) Summed SDM - (e) Final novelty score with estimated boundaries

D_k monotonically decreases with the number of clusters and flattens for some k that is the ideal number of clusters. Differentiation of the D_k allows to estimate for each song the optimal number of labels.

5.2 Evaluation Protocol: Testset and Metrics

Testset: In order to allow the comparison between the results presented here and the ones obtained at the 2012 MIREX [2] evaluation for structural segmentation, we use the Isophonics testset [3], also known as the MIREX09 testset. This testset consists of 297 popular music songs (the

[2] http://www.music-ir.org/mirex/wiki/2012:MIREX2012_Results
[3] http://isophonics.net/

Method	Temp. Seg. Eval. @0,5s F	P	R	Temp. Seg. Eval. @3s F	P	R	Seg. Group. Eval. pF	pP	pR
Repet	22.8	22.4	24.1	64.3	63.6	67.6	59.9	**65.7**	58.3
Novel	**29.5**	26.7	34.7	61.8	55.9	72.5	60.0	62.5	63.2
BaselineFusion	28.8	24.6	**37.7**	63.4	52.8	**83.4**	59.6	64.2	59.6
SDMFusion	28.9	**29.5**	29.4	**65.2**	**66.7**	65.9	**62.1**	62.4	**66.7**

Table 1. Temporal segmentation and segment grouping evaluation on the Isophonics testset

Beatles, Queen, Michael Jackson...)..

Metrics: The temporal segmentation is, as in MIREX, evaluated using the precision P, recall R and F-Measure F. In order to compute the True Positives, False Positives and False Negatives, we used two tolerance windows: 0.5 and 3 seconds. The segment labelling is evaluated, as in MIREX, using of the pairwise Precision, Recall and F-Measure proposed by [11].

5.3 Results and Discussion

The results are indicated into Table 1. The repetition and novelty methods are respectively denoted by "*Repet*" and "*Novel*". The baseline fusion and segments distance based fusion are respectively denoted by "*Baseline Fusion*" and "*SDM Fusion*".

Repet versus Novel: The results obtained for temporal segmentation shows that both repetition- and novelty-based methods tend to over-segment the signal (recall $>$ precision). This is especially true for the novelty-based method. Evaluation with a 0.5s tolerance window shows better performances (F-measure) for the novelty-based method. Increasing the tolerance to 3s then turns to the advantage of the repetition-based method. The results obtained for segment labelling shows comparable performances (pairwise F-Measure) for both methods. The structural segmentations are however of different natures considering their differences in the pairwise recall and precision balance: - labelling using the Repet method tends to over-estimate the number of labels, hence inherently produce over-segmentation (pairwise Precision $>$ pairwise Recall). - the inverse phenomenon is observed using the Novel method.

Fusion methods: The performance evaluation of the baseline fusion method clearly shows a strong over-segmentation (R $>$ P for both tolerance window). Moreover, labelling of the segments for the baseline fusion method shows the worst performance. In contrast, the SDM based fusion method shows very convincing performances for both the temporal segmentation and segment labelling. Indeed, its performance for temporal segmentation (F-measure) is just behind the novelty- based method's performance at 0.5s and obtains the best score at 3s. It is also interesting to note that the temporal over-segmentation observed for both Repet and Novel segmentations is not observed in the SDM Fusion segmentation. This illustrates how the acoustic information is considered in the fusion. This is further validated by looking at the segmentations agreement. Conducting the same experiment as in Section 3.3 indeed shows that 61,8% of the repeated segments and 61,9% of the novelty segments are contained in the SDM Fusion segmentation.

Finally, the segment labelling evaluation shows a very positive impact of the SDM Fusion segmentation. We increase of about 2 percentage points the pairwise F-measure with very balanced pairwise precision and recall. Again, the SDM fusion of segments yield an original structural interpretation benefitting from both the repetition and novelty hypotheses.

6. SUMMARY AND CONCLUSION

In this paper, we proposed a method for the consistent fusion of repetition- and novelty- based temporal segmentations of music. We showed that this fused segmentation benefits from the temporal perspectives given by both hypotheses and is rather influenced by the acoustical consistency of the final segmentation than from one or the other original segmentation. Moreover, we showed that the fusion of the segmentations allows for a strong increase in the segment labelling performance. This paper thus illustrates the potential benefits of developing multiple hypotheses based structural segmentation algorithms. Moreover, we believe that the method is not restricted to the fusion of the repetition and novelty methods and could be applied to other temporal segmentation methods.

7. REFERENCES

[1] Jean-Julien Aucouturier, François 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.

[2] Ruofeng Chen and Ming Li. Music structural segmentation by combining harmonic and timbral information. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[3] Scott Shaobing Chen and P.S. Gopalakrishnan. Clustering via the bayesian information criterion with applications in speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 645–648 vol.2, May 1998.

[4] Matthew L. Cooper and Jonathan Foote. Automatic music summarization via similarity analysis. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2002.

[5] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, 2000.

[6] Masataka Goto. Chorus-section detecting method for musical audio signals. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003.

[7] A.L. Jacobson. Auto-threshold peak detection in physiological signals. In *Engineering in Medicine and Biology Society, 2001. Proceedings of the 23rd Annual International Conference of the IEEE*, volume 3, pages 2194–2195 vol.3, 2001.

[8] Florian Kaiser and Geoffroy Peeters. Multiple hypotheses at multiple scales for audio novelty computation in music. In *38th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, 2013.

[9] Florian Kaiser and Thomas Sikora. Music structure discovery in popular music using non-negative matrix factorization. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, Aug 2010.

[10] Holger Kantz and Thomas Schreiber. *Nonlinear Time Series Analysis*. Cambridge University Press, 2003.

[11] M. Levy and M. Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Transactions on Audio, Speech & Language Processing*, 16(2):318–326, 2008.

[12] 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.

[13] Jouni Paulus, Meinard Müller, and Anssi Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, 2010.

[14] Geoffroy Peeters. *Deriving Musical Structures from Signal Analysis for Music Audio Summary Generation: "Sequence" and "State" Approach*, volume 2771 of *Lecture notes in Computer Science*, pages 143–166. Springer, 2004.

[15] Gabriel Sargent, Frederic Bimbot, and Emmanuel Vincent. A regularity-constrained viterbi agorithm and its application to the structural segmentation of songs. *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2011.

[16] Joan Serra, Meinard Müller, Peter Grosche, and Josep Ll. Arcos. Unsupervised detection of music boundaries by time series structure features. In *AAAI International Conference on Artificial Intelligence*, 2012.

[17] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistics. *Journal of the Royal Statistical Society, series B*, 63:411–423, 2001.

EVALUATING THE QUALITY OF PLAYLISTS BASED ON HAND-CRAFTED SAMPLES

Geoffray Bonnin
TU Dortmund, Germany
geoffray.bonnin@tu-dortmund.de

Dietmar Jannach
TU Dortmund, Germany
dietmar.jannach@tu-dortmund.de

ABSTRACT

The automated generation of playlists represents a particular type of the music recommendation problem with two special characteristics. First, the tracks of the list are usually consumed immediately at recommendation time; second, tracks are listened to mostly in consecutive order so that the sequence of the recommended tracks can be relevant. A number of different approaches for playlist generation have been proposed in the literature. In this paper, we review the existing core approaches to playlist generation, discuss aspects of appropriate offline evaluation designs and report the results of a comparative evaluation based on different data sets. Based on the insights from these experiments, we propose a comparably simple and computationally tractable new baseline algorithm for future comparisons, which is based on track popularity and artist information and is competitive with more sophisticated techniques in our evaluation settings.

1. INTRODUCTION

Among the different application domains of recommender systems (RS), music is often considered as being particularly difficult to deal with [5, 12]. One specific approach for music recommendation and discovery is the automated generation and provision of playlists (mixes). This strategy however induces additional challenges as the tracks are consumed in sequence and usually immediately at recommendation time. This means that the context of the previous recommendations has some influence on user satisfaction and should be taken into account in the playlist generation process.

When our goal is to automatically generate playlists, i.e., lists of sequentially ordered tracks, one key question is the evaluation of their quality. In fact, there might be a number of different factors that influence the perceived value of a playlist, including, e.g., the coherence of the list, or the variety or *freshness* of the songs [9]. Many playlist generation algorithms have been proposed in the literature, but their quality is hard to compare as they often focus on particular families of techniques and use different baseline algorithms and evaluation criteria.

In this paper, we review existing playlist generation approaches and propose a comparably simple and computationally efficient new baseline algorithm for future comparisons which relies on track popularity and artist information. We discuss evaluation designs from the literature and present experiments that use two evaluations metrics based on comparisons with hand-crafted playlists, i.e., playlists made by hand by music enthusiasts. The results show that our algorithm outperforms the other approaches on two out of three data sets when using hit rates as a metric. In addition, the experiments reveal a considerable limitation of using the log-likelihood metric as a means to compare the quality of automatically generated playlists.

2. AUTOMATED PLAYLIST GENERATION

In the following, we review existing approaches to playlist generation and present our new approach. A playlist is usually defined to be an ordered sequence of musical tracks. The playlist generation problem typically consists in creating such a list given either some *seed* information or semantic description [3]. As another input, we might also have some extra information for each track, e.g., the audio signal, the composer, artists, lyrics, tags, ratings, etc.

In this work, we assume that the seed information consists of the listening *history* so far. Given this history, the system presents recommendation lists of tracks to the user, and each time a track is selected the process is repeated [4]. Thus, the problem comes down to the computation of the score of a candidate track t given a playlist history $h = \langle t_1, t_2, ..., t_i \rangle$. The resulting scores – which in some cases correspond to probability estimates – can then be used to filter and rank the remaining tracks.

2.1 Markov chains

Attempting to recommend tracks that represent a smooth transition from the previous track is an obvious approach. This corresponds to the Markov property and leads to a first-order Markov model in which states correspond to tracks. Given a history h of a playlist and a candidate track t, the probability of t in such a model thus only depends on t_i, the last element of h. Examples of playlist modeling approaches based on this strategy include [14] and [6]. In [14], the authors compare a set of approaches to assign transition probabilities to a Markov model including the uniform distribution, tags, audio signal and artist information. In [6], a more sophisticated **Latent Markov Embedding** (**LME**) model in which tracks are represented by

vectors in the Euclidean space is compared with the bigram model and a uniform distribution.

The major limitation of these models is that the assumptions on which they are based may be too strong as the choice of the next track by a user may or may not depend only on the previous track. Although tracks are usually being listened one after the other and transitions between tracks surely have some importance, in practice, the rules users follow to build playlists can be quite different and often contradict this assumption, see also [8].

2.2 Frequent patterns

Another possibility to recommend tracks for playlist generation is to extract frequent patterns from playlists. The common techniques are **association rule** (**AR**) and **sequential pattern** (**SP**) mining. An association rule [1] has the form $A \rightarrow C$, where A and C are two itemsets. *Sequential patterns* are a sequential version of association rules [2] in which the order of the elements in the pattern is also taken into account in the mining process. The additional constraints of sequential patterns over association rules can in general lead to more accurate recommendations, but the approach has a higher computational complexity and requires a larger amount of training data. Another possible limitation of this approach might be the comparably small confidence values for the extracted patterns given the usually high sparsity of musical data sets.

2.3 Neighborhood recommenders

Another way to exploit co-occurrences of tracks is to use a ***k*-nearest-neighbors** (***k*NN**) recommender which is based on the similarity between playlists. Such a kNN approach was proposed in [11] and used as a basis for a more sophisticated recommender which uses sequential patterns of latent topics based on tags. Similar to association rules, this kNN approach not only exploits information about the collocation of items in playlists but also takes the number of shared items in each playlist into account when estimating the probability. However, association rule mining is based on counting the frequency of patterns for all users in an offline process. The kNN approach, in contrast, dynamically computes a "local" probability using the k most similar playlists. In other words, the limitation mentioned in the previous subsection with respect to low confidence values is reduced. The computation of neighborhoods and playlist similarities is however computationally complex both in terms of time and space, making the approach intractable when recommendations have to be made in real time.

2.4 Playlists as users

In principle, if we interpret the playlist generation problem to be similar to the item prediction problem in typical RS settings by considering playlists to be users, existing RS algorithms for item recommendations can be applied including recent learning-to-rank techniques. In particular, the BPR-approach (Bayesian Personalized Ranking) from [15] has been included in previous comparative evaluations for playlist or music recommendation, see e.g. [11] and [13]. The experiments in the last two papers however show that the plain BPR method can be easily outperformed by other methods in particular problem settings.

2.5 Content-based approaches

Using additional information, one can try to enhance the confidence of pattern-based approaches or avoid the complexity of the kNN approach. Such additional information can be the content of the tracks (lyrics, spectrum, etc.), the similarity of musical features [10], user tags, or more simple elements such as artist names. Some of the aforementioned approaches use some forms of content and metadata. For instance, the topic-aware hybrid recommender of [11] uses tags to determine topics, but does not solve the scalability problem of the underlying kNN approach. Also McFee and Lanckriet [14] experiment with Markov models that use tags, the audio signal and artist names. However, their approach does not solve the problem of the strong assumption of the Markov property.

Regarding the incorporation of additional information into the recommendation process, we hypothesize that the use of artist names in general is particularly promising as this type of data is objective, easy to obtain and to process (as opposed to, for instance, information about the playlist topic, genre or style).

2.6 Popularity-based approaches

In many application domains for RS and in particular in the music domain [5], we can observe a so called "long tail" distribution of items, meaning that a small subset of the items accounts for the majority of transactions or interactions. This popularity bias results in the fact that simple popularity-based approaches, which present the same set of popular items to everyone, can represent a comparably hard baseline [7]. Given these observations, we included two approaches that are based on popularity combined with artist information in the experiments.

"Same artists - greatest hits" (SAGH): In [13], the authors propose a baseline algorithm for music recommendation – not in the context of playlists – called "Same artists - greatest hits", which simply recommends the most popular songs of the artists appearing in the user's listening history. Their experiments on the Million Song data set shows that higher prediction accuracy can be obtained with such an approach than when using, e.g., the above-mentioned BPR method. This method would thus be a hybrid that uses both additional information as well as popularity.

"Collocated artists - greatest hits" (CAGH): In this paper, we do not only apply the previous scheme, but propose an extension to it. Our assumption is that the different artists that are included in playlists by the users are not too different from each other. We thus propose to recommend tracks based on the frequency of the collocation of artists.

More precisely, we compute the similarity between two artists a and b according to the following formula:

$$\text{sim}_{\text{a}}(a,b) = \frac{\sum_p (\delta_{a,p} \cdot \delta_{b,p})}{\sqrt{\sum_p \delta_{a,p} \cdot \sum_p \delta_{b,p}}}$$

with $\delta_{a,p} = 1$ if playlist p contains a and 0 otherwise. The similarity thus depends on the collocations of artists within playlists, which can be computed offline. Our proposed formula for the computation of the score of a next track t with artist a given a playlist beginning h is as follows:

$$\text{score}_{\text{CAGH}}(t,h) = \sum_{b \in A_h} \text{sim}_{\text{a}}(a,b) \cdot \text{counts}(t) \quad (1)$$

where A_h is the set of artist names of the tracks in h and $\text{counts}(t)$ is the number of occurrences of t in the data set, which corresponds to the greatest hits of the data set.

3. A COMPARATIVE EVALUATION OF PLAYLIST GENERATION STRATEGIES

The presented playlist generation techniques follow different strategies and exploit different types of inherent characteristics of the playlists or rely on external information. The goal of this study is to obtain a better understanding of different aspects related to the generation and evaluation of playlists. In particular, we aim to better understand the role of sequentiality and popularity, find out if different metrics follow the same trends in a comparative evaluation and if the observations are consistent across different data sets.

3.1 Data Sets

We used three data sets in our experiments. One is from Artofthemix, which is probably the most commonly used data set for related research [11, 14]. The second was retrieved from last.fm [1]. The third was provided to us by 8tracks [2]. In order to reduce the sparsity of the data, we used the web service of Musicbrainz [3] to correct artist and track misspellings. We also removed playlists of size 1. For the last.fm data, we furthermore decided to select playlists in a way that long-tail tracks are used at least twice. Table 1 shows the data set characteristics.

Notice that the Artofthemix data does not contain user IDs. As implicitly done also by [11], we consider users as being equivalent to playlists, as they usually do not create large numbers of playlists. Regarding track occurrences, the last.fm and 8tracks data sets have a similar average track usage count (5.5 and 5.3) [4]. This usage count is significantly smaller for Artofthemix (2.7). Another related characteristic is the long tail distribution of track usages. Table 1 divides the corresponding distribution into three parts: head, middle and tail. The "head" contains tracks which appeared more than 20 times in playlists, tracks in the "middle" were included in playlist between 2 and 20 times, and songs from the "tail" were only used once or twice. These values are admittedly somewhat arbitrary but allow us to roughly compare the respective distributions. The resulting proportions reveal another difference between the last.fm and 8tracks data sets: although they have a similar average track usage count, the size of the long tail of 8tracks is much larger.

	last.fm	Aotm	8tracks
Playlists	50,000	28,636	99,542
Users	47,603	–	51,743
Tracks	69,022	214,769	179,779
Avg. tracks/playlist	7.6	20.1	9.7
Avg. track usage count	5.5	2.7	5.3
Head	4.8%	1.6%	4.5%
Middle	35.0%	18.5%	25.7%
Tail	60.1%	79.9%	69.8%
Artists	11,788	47,473	29,352
Avg. artists/playlist	4.5	17.3	8.9
Avg. artist usage count	32.2	12.1	32.7
Artist reuse rate	31.1%	21.8%	13.8%

Table 1. Properties of the data sets.

Regarding artist-based recommendation approaches, Table 1 shows that playlists usually contain fewer artists than tracks. The row "artist reuse rate" in the table represents the percentage of cases when the artist of the last track of a playlist already appeared in the same playlist before. The corresponding values are 31.1% for the last.fm data set, 21.8% for the Artofthemix data set and 13.8% for the 8tracks data set. This represents another difference between the last.fm and 8tracks data sets: although they have a similar average artist usage count, the artists are more distributed across the playlists in the 8tracks data because 8tracks' license only allows for up to two songs from the same artist per playlist. Overall, we think that these values represent a strong argument to emphasize on artist names as an additional information when recommending tracks, except maybe for the 8tracks data set.

Using three data sets with quite different characteristics should allow us to analyze how the different algorithms perform in different situations. In general, generating recommendations based on the Artofthemix data set should be much more difficult than with the last.fm and 8tracks data sets, as it is smaller and the individual tracks are less often used. Other factors may however play a major role as well, in particular the size of the long tail.

3.2 Evaluation Metrics

Playlist generation is usually evaluated according to three possible strategies: semantic cohesion, human opinion survey and comparison with hand-crafted playlists. Semantic cohesion corresponds to the assumption that a good playlist is a playlist which tracks are as homogeneous as possible, e.g., in terms of genre or style, which can be considered as a strong assumption. Human opinion surveys do not have this drawback but are time consuming and difficult to reproduce. We thus choose the third option and

[1] http://www.lastfm.com/api
[2] http://8track.com
[3] http://musicbrainz.org/ws/2/
[4] The track/artist usage count means how often a track/artist was used in all playlists

evaluate playlisters by comparing their output with handcrafted playlists. Two evaluation strategies are common, the hit rate and the average log-likelihood.

3.2.1 Measuring hit rates

A first way of measuring the accuracy of playlist generation for music recommendation is to use the *hit rate*, i.e., the proportion of relevant predictions on a test set. An evaluation method of this type is used, e.g., by [11], who hide the last element of each given playlist, which has then to be recommended by the algorithm. In general, any subset of playlist elements could be hidden in such a protocol. Removing the last one however is based on the assumption that the sequential history of a playlist can be relevant.

The limitation of this evaluation metric is that it corresponds to the assumption that the actual next tracks in the playlist are the only relevant tracks that can be recommended, although some other tracks may be relevant. In other words, it is possible that hundreds of tracks are relevant, but as the recommender has to select a subset of them, the actual next tracks of the test playlists might not be recommended. As it is impossible to know how many tracks are relevant for each situation, it is reasonable to analyze the accuracy of a system using longer recommendation lists. Such lists could of course not be used in a real framework, but our goal here is only to compare the algorithms. The assumption is then that there is a correspondence between the size of the recommendation lists and the average number of relevant tracks. Still, the hit rate can only be considered to be a lower bound for the accuracy.

3.2.2 Measuring the average log-likelihood

Another way to measure accuracy is to use the *average log-likelihood*. The average log-likelihood can be used to measure how likely a system is to recommend the tracks of a given set of playlists through a weighted random process. More precisely, given a test set of playlists, the average log-likelihood can be determined by computing the probability of observing each next track according to the corresponding playlist history and some model learned on the training data. Research on music recommendation using playlists that use this metric includes [6] and [14]. Obviously, the application of this measure requires that the output of a playlist recommender can be expressed as probability values for each song, which can be easily obtained by a normalization over the prediction lists.

In contrast to the hit rate, which provides a realistic lower bound on the accuracy that is directly interpretable, this metric is not interpretable on an absolute scale: the possible values vary between $-\infty$ (at least one track in the test set has a corresponding 0 probability in the model) and 0 (all probabilities in the model for all tracks in the test set are 1). Thus, this metric does not tell us if a generative approach leads to good playlists, but allows us to compare the results of different generative approaches. It can thus be considered as a complementary measure to the hit rate.

As only one track having a 0 probability is sufficient to induce a $-\infty$ average log-likelihood, 0 probabilities must be avoided. This requires an additional smoothing step, which might result in a strong bias. For instance, new tracks will always have such 0 probabilities with a frequency-based approach. In that situation a combination with the uniform distribution can be used, but then the weight of the uniform distribution may become too large given the long-tailed distribution of our data sets.

3.2.3 Computational complexity

Another important fact that should be taken into account is the computational complexity. Indeed, as opposed to, for instance, movie recommendation, for which recommendations can be computed offline and updated regularly, music recommendation can be highly dynamic and contextual. Users usually listen to tracks in sequence, where each track lasts a few minutes. Therefore, a music recommender should be able to provide fast contextual recommendations. Moreover, as the number of tracks that can be recommended is usually very high, the efficiency of the training phase can become crucial. In the subsequent analysis of algorithms, we will thus also briefly discuss aspects of computational complexity.

4. EXPERIMENTS

In the following evaluation, we use the two aforementioned accuracy metrics: hit rate and average log-likelihood. A 10-fold cross-validation procedure was applied for both metrics on the three data sets. Recall that the total number of tracks of the data set highly influences the hit rate values. In [11], the results for prediction lists of size varying between 1 and 300 given $21,783$ tracks are reported. This corresponds to the selection of about 1.5% of the tracks. We used a similar proportion in our experiments and set the maximum size of the prediction lists to $1,000$ for last.fm, $3,000$ for Artofthemix and $2,500$ for 8tracks.

4.1 Evaluating Hit Rates

Figure 1 shows the results of comparing five different recommendation approaches on the three data sets using the hit rate. The approaches include the three above-mentioned frequent-pattern approaches AR and SP, a kNN recommender using 50 and 100 neighbors, the SAGH recommender and our new baseline recommender CAGH [5].

We can first notice that all approaches lead to comparably low accuracy values for short recommendation lists. For longer recommendation lists, our new CAGH recommender clearly outperforms the other approaches on the last.fm and Artofthemix data, except for recommendation lists longer than $2,800$ for the frequent-patterns approach on the Artofthemix data. On the data from 8tracks, the frequent pattern approach clearly outperforms all other approaches, followed by the kNN approach with 100 neighbors, and the CAGH recommender. The reason for the lower performance of the CAGH recommender is probably the better distribution of artists across playlists on this particular data set due to the corresponding license restrictions (see section 3.1).

[5] The method of [11] is not included here but is comparable to the kNN method according to their measurements.

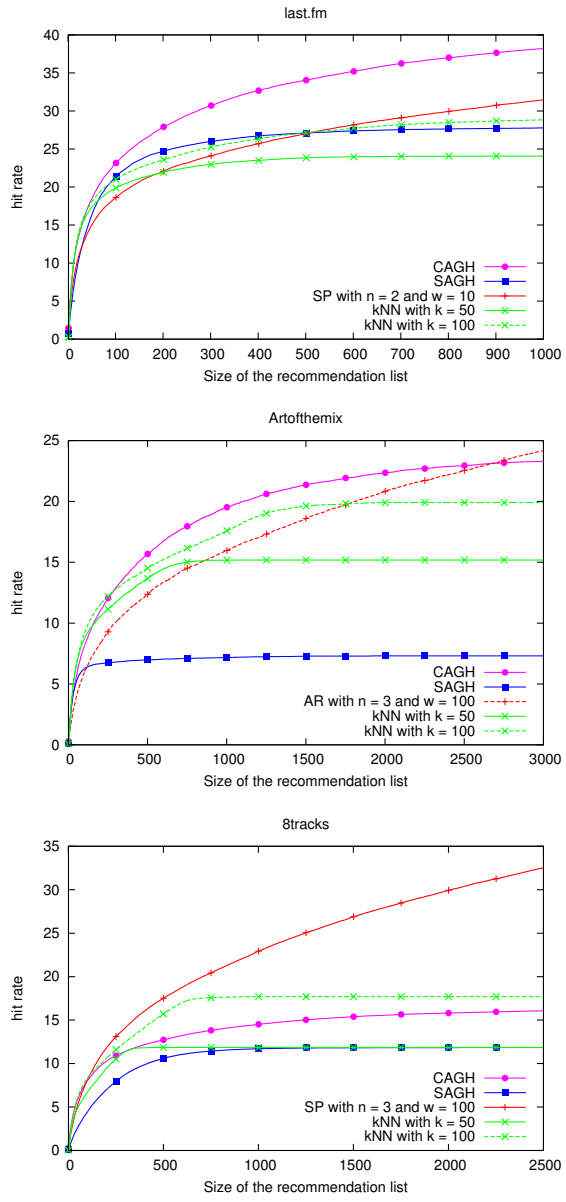


Figure 1. Hit rates of the different approaches.

In general, using more neighbors enhances the accuracy of the kNN approach on the three data sets. The kNN approach may even outperform all the other approaches using more than 100 neighbors. However, both neighborhood sizes used in these experiments are already high and make the recommendation algorithm not only intractable in terms of space requirements, but also in terms of running time. Still, kNN approaches lead to a lower accuracy values than both our new baseline approach and the frequent patterns method. More precisely, on the three data sets the accuracy of the kNN approach seems to be limited by the size of the recommendation lists it is able to build. This is probably the reason why the frequent patterns outperform this approach on the 8tracks data set, as it is close to a kNN approach that uses all the neighbors.

Other observations depend on the used data set. In particular, for the last.fm data set, the SAGH recommender leads to results that are similar to those of the kNN recommender with 100 neighbors. For the Artofthemix data set, the SAGH recommender is clearly outperformed by all other approaches. For the 8tracks data set, it leads to results that are similar to those of the kNN recommender with 50 neighbors for recommendation lists longer than 750.

Beside the results shown in Figure 1, we also experimented with models based on the Markov property, among them the simple bigram model and the recent Latent Markov Embedding (LME) model of [6]. Despite the long time that can be required to train these models – e.g., several weeks for the LME model – these methods led to particularly low accuracy values which were consistently below 10% for recommendation lists of size $1,000$ for the last.fm data set and 5% for recommendation lists of size $3,000$ for the Artofthemix data set. We therefore omit these results in this paper. In general, given these comparably strong differences, assuming the Markov property might be too strong for this problem setting. Furthermore, our results indicate that emphasizing on artist names can be particularly promising for accurate track recommendation in the context of playlist generation.

4.2 Evaluating Average Log-Likelihoods

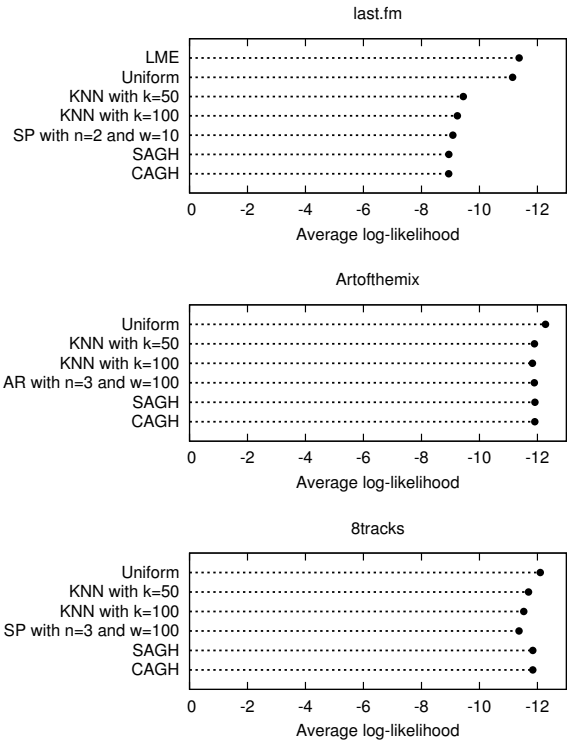


Figure 2. Av. log-likelihood of the different approaches

Figure 2 shows the results of the generative versions of the five approaches on the three data sets using the average log-likelihood. The experimented methods all correspond to mixture models that combine the uniform distribution with the approaches of the previous set of experiments. In order to perform these evaluations, each training set of the cross-validation process was further split into a learning set for obtaining model probabilities and a validation set to compute optimized weights using the Expectation-

Maximization algorithm. We also provide the results of the LME model with standard parameters on the last.fm data (on both other data sets the training process lasted more than a month).

We focus on the correspondences with the previous set of hit rate results. On the last.fm data set the mixture of the uniform distribution with the CAGH and SAGH recommenders leads to the best average log-likelihood values. It is worth noting that both models provide similar results according to this metric although on the same data set the CAGH recommender was a clear winner in terms of the hit rate. We also evaluated all the mixture models of this section in terms of the hit rate and obtained a similar output: both CAGH and SAGH recommenders lead to a similar hit rate when combined with the uniform distribution. More precisely, this combination strongly lowers the hit rate of the CAGH recommender, but not that of the SAGH recommender. This result confirms the bias of using this metric we mentioned in section 3.2: as a relatively important number of tracks are new in the test sets, EM-based mixtures tend to give more weight to tracks coming from the uniform distribution than tracks coming from the similar artists. This phenomenon also appears on the three data sets, although the CAGH and SAGH do not outperform the other models. On the Artofthemix data set, it even applies to all the models: the accuracy of all approaches is strongly lowered when combined with the uniform distribution. These results indicate a limitation of using the average log-likelihood metric, i.e., smoothing the models have highly lowered the accuracy, although in reality a RS does not have to avoid 0 probabilities.

When testing the LME model on the last.fm data set, the experiments showed that the model lead to lower results than the uniform distribution. This confirms the previous conclusion about the use of the Markov property.

5. CONCLUSION

This paper proposes a classification of existing approaches for playlist generation and discusses limitations of typical experimental designs, which for example do not take scalability aspects into account or are based on comparably strong assumptions such as the Markov property. Based on this discussion, we propose a new computationally efficient recommendation scheme based on popularity and artist information. An experimental comparative evaluation showed that our algorithm outperforms the other approaches in terms of hit rate on two of three data sets. On the remaining data set, our recommender is on a par with neighborhood-based approaches and was outperformed by a frequent pattern technique. This difference is probably caused by the high dispersion of artists among playlists due to license constraints. However, other factors may have induced this difference in accuracy, which we are investigating in our current work. Our evaluations also put forward a strong limitation of using the average log- likelihood metric: it implies to smooth models in order to avoid 0 probabilities which resulted in a strong degradation of the quality of the approaches.

6. ACKNOWLEDGMENTS

We thank 8tracks for providing us their valuable data.

7. REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In *SIGMOD 1993*, pages 207–216, 1993.

[2] R. Agrawal and R. Srikant. Mining Sequential Patterns. In *Proc. ICDE 1995*, pages 3–14, 1995.

[3] L. Barrington, R. Oda, and G. Lanckriet. Smarter than Genius? Human Evaluation of Music Recommender Systems. In *Proc. ISMIR 2009*, pages 357–362, 2009.

[4] Dominikus Baur, Sebastian Boring, and Andreas Butz. Rush: Repeated Recommendations on Mobile Devices. In *Proc. IUI 2010*, pages 91–100, 2010.

[5] Ò. Celma. *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.

[6] S. Chen, J.L. Moore, D. Turnbull, and T. Joachims. Playlist Prediction via Metric Embedding. In *Proc. KDD 2012*, pages 714–722, 2012.

[7] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *ACM RecSys 2010*, pages 39–46, 2010.

[8] S. Cunningham, D. Bainbridge, and A. Falconer. 'More of an Art than a Science': Supporting the Creation of Playlists and Mixes. In *Proc. ISMIR 2006*, pages 240–245, 2006.

[9] B. Fields. *"Contextualize Your Listening: The Playlist as Recommendation Engine"*. PhD thesis, Goldsmiths, University of London, London, UK, April 2011.

[10] A. Flexer, D. Schnitzer, M. Gasser, and G. Widmer. Playlist Generation Using Start and End Songs. In *ISMIR 2008*, pages 173–178, 2008.

[11] N. Hariri, B. Mobasher, and R. Burke. Context-Aware Music Recommendation Based on Latent Topic Sequential Patterns. In *Proc. ACM RecSys 2012*, pages 131–138, 2012.

[12] P. Lamere and Ò. Celma. Music Recommendation and Discovery Remastered, Tutorial at ACM RecSys 2011. Online at `http://www.slideshare.net/slideshow/embed_code/9860137`, 2011.

[13] B. McFee, T. Bertin-Mahieux, D. Ellis, and G. Lanckriet. The million song data set challenge. In *Proc. AdMIRe'12*, 2012.

[14] B. McFee and G. Lanckriet. The Natural Language of Playlists. In *Proc. ISMIR 2011*, 2011.

[15] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proc. UAI*, pages 452–461, 2009.

EXPLICIT DURATION HIDDEN MARKOV MODELS FOR MULTIPLE-INSTRUMENT POLYPHONIC MUSIC TRANSCRIPTION

Emmanouil Benetos and Tillman Weyde
Music Informatics Research Group, Department of Computer Science, City University London
{emmanouil.benetos.1, t.e.weyde}@city.ac.uk

ABSTRACT

In this paper, a method for multiple-instrument automatic music transcription is proposed that models the temporal evolution and duration of tones. The proposed model supports the use of spectral templates per pitch and instrument which correspond to sound states such as attack, sustain, and decay. Pitch-wise explicit duration hidden Markov models (EDHMMs) are integrated into a convolutive probabilistic framework for modelling the temporal evolution and duration of the sound states. A two-stage transcription procedure integrating note tracking information is performed in order to provide more robust pitch estimates. The proposed system is evaluated on multi-pitch detection and instrument assignment using various publicly available datasets. Results show that the proposed system outperforms a hidden Markov model-based transcription system using the same framework, as well as several state-of-the-art automatic music transcription systems.

1. INTRODUCTION

Automatic music transcription (AMT) is the process of converting an acoustic musical signal into some form of music notation [13]. In the music information retrieval literature, AMT typically involves the detection of multiple concurrent pitches (multi-pitch detection), the estimation of note onsets and offsets (note tracking) and the estimation of instrument identities (instrument identification/assignment). It is generally considered to be an open problem, especially for highly polyphonic music signals and multiple instruments. For a recent review of AMT systems, the reader is referred to [12].

A large part of AMT systems employ spectrogram factorization methods for multi-pitch detection. These systems attempt to decompose an input time-frequency representation as a series of spectral components and pitch activations, using a variety of constraints (regarding polyphony level, instrument identities, spectral envelopes, and temporal continuity among others); systems related to the proposed work will be presented below.

In [6], Dessein et al. propose an AMT system for piano music which uses non-negative matrix factorization (NMF) with beta-divergence and pre-extracted note templates, which is able to transcribe pieces in real-time. Vincent et al. [16] propose a harmonic variant of NMF for decomposing a spectrogram into a series of narrowband harmonic spectra, which are also smooth across frequency (also called the spectral smoothness assumption [13]). In [4], Carabias-Orti et al. propose a system for multi-pitch detection and instrument identification using NMF with source-filter model constraints. Grindlay and Ellis [11] utilize a probabilistic variant of NMF called probabilistic latent component analysis (PLCA) for decomposing a spectrogram into a series of *eigeninstrument* templates, pitch activations, and source contributions, and evaluate their method for multi-pitch detection and instrument assignment. Yoshii and Goto [17] proposed a non-parametric model for music signal analysis which decomposes an input spectrogram as a series of source-filter templates derived from an autoregressive model. Finally, in [2] Benetos and Dixon proposed a variant of convolutive PLCA for modelling the evolution of notes using sound state templates (such as attack, sustain, decay) with hidden Markov model-based constraints.

In this paper, we integrate explicit duration hidden Markov models (EDHMMs) [7,18] within the spectrogram factorization framework of [2], in order to model the duration of sound states within a note. Contrary to hidden Markov models (HMMs), where the state duration is (implicitly) geometrically distributed, EDHMMs form a specific case of hidden semi-Markov models [18], where each state has a variable duration. Alternatively, it can be viewed that an EDHMM can emit a sequence of observations instead of a single one. The additional information in EDHMMs is modelled through the use of a duration probability per state. EDHMMs have been shown to overcome the limitations posed by HMMs regarding state durations and have been successfully used in a variety of applications (see [18] for a review).

The proposed model uses pitch-wise EDHMMs for constraining the order of the sound states, while also supporting the use of multiple templates per pitch and instrument, and also shift-invariance across log-frequency for supporting tuning changes and frequency modulations. In addi-

E. Benetos is supported by a City University London Research Fellowship.

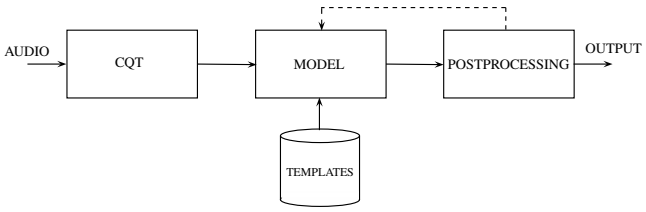


Figure 1. Proposed system diagram.

tion, we propose a two-stage transcription procedure in order to eliminate spurious pitch detections. The proposed model is trained on note samples from the RWC database [10] and is tested using recordings from the MAPS database [8], the TRIOS dataset [9], and the MIREX MultiF0 recording [1]. Multi-pitch detection and instrument assignment results show that the proposed EDHMM-based system is able to model the durations of sound states and the overall evolution of notes, and its temporal constraints lead to improved performance compared to hidden Markov models (HMMs) used in the same framework. Finally, the proposed system outperforms several AMT methods in the literature for the same experiments.

The outline of this paper is as follows. Section 2 presents the proposed EDHMM-based transcription model, along with the postprocessing steps. The datasets used for training and testing, as well as the evaluation metrics and experimental results, are presented in Section 3. Finally, conclusions are drawn and future directions are indicated in Section 4.

2. PROPOSED METHOD

In this section, the proposed EDHMM-constrained automatic transcription model is described, along with the update rules for estimating the various model parameters and the steps used for post-processing; the proposed system diagram can be seen in Fig. 1.

2.1 Model

The proposed model aims to express the evolution of notes in multiple-instrument polyphonic music as a succession of sound state templates, further constrained by the ordering and the expected duration of each sound state. These temporal constraints are incorporated into a model which supports multiple templates per pitch and instrument, and also supports shift-invariance across log-frequency in order to model tuning changes and frequency modulations. In order to achieve this, we integrate independent pitch-wise explicit duration hidden Markov models (EDHMMs) [18] into the HMM-constrained automatic transcription model of [2]. Thus, the proposed model can be called EDHMM-constrained shift-invariant PLCA.

More formally, the normalized magnitude log-frequency spectrogram $V_{\omega,t}$ (ω denotes log-frequency and t denotes time) which is used as input, is decomposed into a series of sound state spectral templates per instrument and pitch, a time-varying pitch shifting parameter, a time-varying instrument contribution per pitch, a pitch activation, and finally a sound state activation per pitch, which is controlled by its respective EDHMM. If we denote the collection of observations for all time frames as $\bar{\omega}$, the proposed model in terms of the observations is given by:

$$\begin{aligned} P(\bar{\omega}) = & \sum_{\bar{q}^{(1)},\cdots,\bar{q}^{(\mathcal{P})}} \sum_{\bar{d}^{(1)},\cdots,\bar{d}^{(\mathcal{P})}} P(q_1^{(1)}) \cdots P(q_1^{(\mathcal{P})}) \\ & P(d_1^{(1)}) \cdots P(d_1^{(\mathcal{P})}) \\ & \left(\prod_t P(q_t^{(1)}|q_{t-1}^{(1)}, d_{t-1}^{(1)}) P(d_t^{(1)}|q_t^{(1)}, d_{t-1}^{(1)}) \right) \cdots \\ & \left(\prod_t P(q_t^{(\mathcal{P})}|q_{t-1}^{(\mathcal{P})}, d_{t-1}^{(\mathcal{P})}) P(d_t^{(\mathcal{P})}|q_t^{(\mathcal{P})}, d_{t-1}^{(\mathcal{P})}) \right) \\ & \left(\prod_t P(\bar{\omega}_t|q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}) \right) \end{aligned} \quad (1)$$

where $p = 1, \cdots, \mathcal{P}$ denotes pitch, $q^{(p)}$ denotes the sound state for the p-th pitch, $d^{(p)}$ denotes the duration distribution for the p-th pitch, $P(q_1^{(p)})$ is the sound state prior for the p-th pitch, $P(d_1^{(p)})$ is the duration prior for the p-th pitch, $\bar{q}$ is the sequence of draws of q, $\bar{d}$ is the sequence of draws of d, and finally $P(\bar{\omega}_t|q_t^{(1)}, \ldots, q_t^{(\mathcal{P})})$ is the observation probability for a given observation $\bar{\omega}_t$.

An EDHMM has state transitions only at the end of a segment, and its duration distributions generate segment lengths only at every state switch [7]:

$$P(q_{t+1}^{(p)}|q_t^{(p)}, d_t^{(p)}) = \begin{cases} \delta(q_{t+1}^{(p)}, q_t^{(p)}), & d_t > 1 \\ P(q_{t+1}^{(p)}|q_t^{(p)}), & \text{otherwise} \end{cases}$$

$$P(d_{t+1}^{(p)}|q_{t+1}^{(p)}, d_t^{(p)}) = \begin{cases} \delta(d_{t+1}^{(p)}, d_t^{(p)} - 1), & d_t > 1 \\ P(d_{t+1}^{(p)}|q_{t+1}^{(p)}), & \text{otherwise} \end{cases}$$

where $P(q_{t+1}^{(p)}|q_t^{(p)})$ is the pitch-wise sound state transition matrix, and $P(d_t^{(p)}|q_t^{(p)})$ is the pitch-wise sound state duration distribution. Also, $\delta(x, y) = 1$ if $x = y$ and 0 otherwise.

Since in the PLCA-based models $V_{\omega,t}$ represents the number of times ω has been drawn at the t-th time frame, the observation probability is calculated as:

$$P(\bar{\omega}_t|q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}) = \prod_{\omega_t} P_t(\omega_t|q_t^{(1)}, \ldots, q_t^{(\mathcal{P})})^{V_{\omega,t}} \quad (2)$$

In the proposed model, $P_t(\omega_t|q_t^{(1)}, \ldots, q_t^{(\mathcal{P})})$ is decomposed as:

$$\begin{aligned} & P_t(\omega_t|q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}) = \\ & \sum_{s_t, p_t, f_t} P_t(p_t) P_t(s_t|p_t) P(\omega_t - f_t|s_t, p_t, q_t^{(p_t)}) P_t(f_t|p_t) \end{aligned} \quad (3)$$

where s denotes the instrument source, f is the pitch shifting parameter, $P_t(p_t)$ is the pitch activation, $P_t(s_t|p_t)$ is the time-varying instrument contribution for each pitch, $P(\omega|s, p, q^{(p)})$ are the sound state spectral templates per

source s, pitch p, and sound state $q^{(p)}$, and $P_t(f_t|p_t)$ is the log-frequency shifting distribution per pitch over time. The subscript t in f_t, ω_t, s_t, p_t denotes the values of variables f, ω, s, p taken at time t. The shifting parameter f is constrained to a semitone range around the ideal tuning position of each pitch. Since in the proposed system the time-frequency representation used is the constant-Q transform (CQT) with a log-frequency resolution of 60 bins/octave and a 40ms step [15], this implies that $f \in [1, 5]$. We also set a maximum duration for each sound state: $d \in [1, 20]$, which means that the maximum duration of each sound state is 800ms.

2.2 Parameter Estimation

The unknown model parameters of Section 2.1 can be estimated using the Expectation-Maximization (EM) algorithm [5]. For the E-step, the posterior for all hidden variables is:

$$P_t(f_t, s_t, p_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega}) = P_t(q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega}) P_t(f_t, s_t, p_t|\omega_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}) \quad (4)$$

We assume that the pitch-wise EDHMMs are independent, thus the joint probability of all sound states is decomposed as:

$$P_t(q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega}) = \prod_{p=1}^{\mathcal{P}} P_t(q_t^{(p)}|\bar{\omega}) \quad (5)$$

where:

$$P_t(q_t^{(p)}|\bar{\omega}) = \frac{\sum_{\tau<t}\big(\alpha_\tau^{*(p)}(q_{\tau+1}^{(p)})\beta_\tau^{*(p)}(q_{\tau+1}^{(p)}) - \alpha_\tau^{(p)}(q_\tau^{(p)})\beta_\tau^{(p)}(q_\tau^{(p)})\big)}{\sum_{q_t^{(p)},\tau<t}\big(\alpha_\tau^{*(p)}(q_{\tau+1}^{(p)})\beta_\tau^{*(p)}(q_{\tau+1}^{(p)}) - \alpha_\tau^{(p)}(q_\tau^{(p)})\beta_\tau^{(p)}(q_\tau^{(p)})\big)} \quad (6)$$

where $\alpha_\tau^*(q_{\tau+1})$, $\alpha_\tau(q_\tau)$ are the EDHMM forward variables and $\beta_\tau^*(q_{\tau+1})$, $\beta_\tau(q_\tau)$ are the EDHMM backward variables; all aforementioned forward-backward variables can be computed using recursive formulae [18].

The second term of (4) can be computed using Bayes' theorem and the notion that $P(\omega_t|f_t, s_t, p_t, q_t^{(p_t)}) = P(\omega_t - f_t|s_t, p_t, q_t^{(p_t)})$:

$$P_t(f_t, s_t, p_t|\omega_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}) = P_t(f_t, s_t, p_t|\omega_t, q_t^{(p_t)}) = \frac{P_t(p_t)P(\omega_t - f_t|s_t, p_t, q_t^{(p_t)})P_t(f_t|p_t)P_t(s_t|p_t)}{\sum_{p_t,s_t,f_t} P_t(p_t)P(\omega_t - f_t|s_t, p_t, q_t^{(p_t)})P_t(f_t|p_t)P_t(s_t|p_t)} \quad (7)$$

For the M-step, the update equations for the unknown parameters are as follows:

$$P_t(p_t) = \frac{\sum_{\omega_t,f_t,s_t,q_t^{(1)},\cdots,q_t^{(\mathcal{P})}} V_{\omega,t}P_t(f_t, s_t, p_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega})}{\sum_{p_t,\omega_t,f_t,s_t,q_t^{(1)},\cdots,q_t^{(\mathcal{P})}} V_{\omega,t}P_t(f_t, s_t, p_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega})} \quad (8)$$

$$P_t(s_t|p_t) = \frac{\sum_{\omega_t,f_t,q_t^{(1)},\cdots,q_t^{(\mathcal{P})}} V_{\omega,t}P_t(f_t, s_t, p_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega})}{\sum_{s_t,\omega_t,f_t,q_t^{(1)},\cdots,q_t^{(\mathcal{P})}} V_{\omega,t}P_t(f_t, s_t, p_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega})} \quad (9)$$

$$P_t(f_t|p_t) = \frac{\sum_{\omega_t,s_t,q_t^{(1)},\cdots,q_t^{(\mathcal{P})}} V_{\omega,t}P_t(f_t, s_t, p_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega})}{\sum_{f_t,\omega_t,s_t,q_t^{(1)},\cdots,q_t^{(\mathcal{P})}} V_{\omega,t}P_t(f_t, s_t, p_t, q_t^{(1)}, \ldots, q_t^{(\mathcal{P})}|\bar{\omega})} \quad (10)$$

$$P(q_1^{(p)}) = \frac{P_1(q_1^{(p)}|\bar{\omega})}{\sum_{q_1^{(p)}} P_1(q_1^{(p)}|\bar{\omega})} \quad (11)$$

$$P(q_{t+1}^{(p)}|q_t^{(p)}) = \frac{\sum_t \alpha_t^{(p)}(q_t^{(p)})P(q_{t+1}^{(p)}|q_t^{(p)})\beta_t^{*(p)}(q_{t+1}^{(p)})}{\sum_{q_{t+1}^{(p)},t} \alpha_t^{(p)}(q_t^{(p)})P(q_{t+1}^{(p)}|q_t^{(p)})\beta_t^{*(p)}(q_{t+1}^{(p)})} \quad (12)$$

$$P(d_t^{(p)}|q_t^{(p)}) = \frac{\sum_t \alpha_t^{*(p)}(q_{t+1}^{(p)})P(d_t^{(p)}|q_t^{(p)})\beta_{t+d}^{p}(q_{t+d}^{(p)})\prod_\tau P(\bar{\omega}_\tau|q_\tau^{(p)})}{\sum_{d_t^{(p)},t} \alpha_t^{*(p)}(q_{t+1}^{(p)})P(d_t^{(p)}|q_t^{(p)})\beta_{t+d}^{p}(q_{t+d}^{(p)})\prod_\tau P(\bar{\omega}_\tau|q_\tau^{(p)})} \quad (13)$$

where $\tau = t+1, \cdots, t+d$.

It should be noted that we consider the sound state templates to be fixed, so no update rule for $P(\omega|s, p, q^{(p)})$ exisits. Using fixed templates, 10-15 iterations using the update rules presented in the present section are sufficient for convergence. The output of the system is a pitch activation which is scaled by the energy of the log-spectrogram:

$$P_t(p)\sum_\omega V_{\omega,t} \quad (14)$$

In order to further constrain the model so that it reaches more meaningful solutions, sparsity is enforced in $P_t(p)$ and $P_t(s|p)$, by modifying the update rules in (8) and (9), where a power greater than 1 is applied to the numerators and denominators, which leads to sharpened distributions, thus encouraging sparsity [2]. Even though convergence is not guaranteed, it is observed in practice. This procedure implies that only few pitches need to be active at each time frame, and also that for a note at a given time frame, only few instruments are responsible for producing it.

As an example of the learned EDHMM parameters using the proposed system, Fig. 2 shows the learned duration distributions and sound state transitions for a D4 note, using a piano recording as input to the system. It can be seen that the duration distribution for the 1st sound state (which corresponds to an attack state) favors short durations, while the duration distribution for the 2nd state (which corresponds to the steady state) favors much longer durations. Also, the resulting transition matrix also shows the linear succession between the sound states.

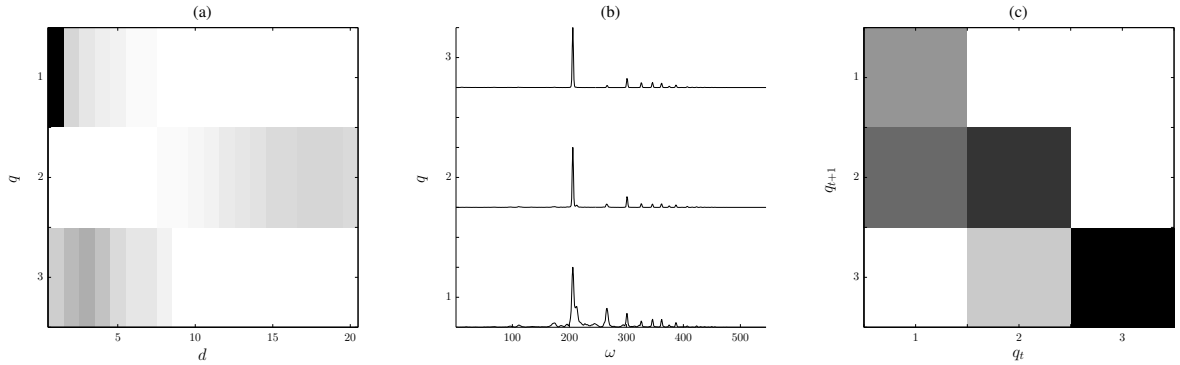


Figure 2. EDHMM parameters learned for note D4 using the 'MAPS_MUS-alb_se2_ENSTDkCl' piece from the MAPS database. (a) The duration distribution matrix $P(d|q)$. (b) Pre-computed piano sound state templates for note D4. (c) The sound state transition matrix $P(q_{t+1}|q_t)$.

2.3 Post-processing

Since the resulting pitch activation from (14) is non-binary, a postprocessing procedure needs to take place in order to convert it to a binary piano-roll or a MIDI-like representation (this procedure is also called note tracking). As in the vast majority of spectrogram factorization-based automatic transcription systems (e.g. [6, 11]), we perform thresholding on the pitch activation, followed by a process for removing note events with a duration less than 80ms. We should note that the HMM-based postprocessing method of [2] was found not to perform well on pieces with fast tempo and rapid note changes. An example of the output of the post-processing step compared with a ground-truth transcription is given in Fig. 3, using a segment from a piano sonata.

A system variant is also proposed, where after detecting all active pitches in the final piano-roll, the update rules of subsection 2.2, are run again, but instead of setting p as to cover the entire pitch range, we only use the list of active pitches estimated in the first run. This two-stage process also helps in further constraining the solution by removing any pitches that might appear in $P_t(p)$ but are nevertheless removed in the postprocessing step.

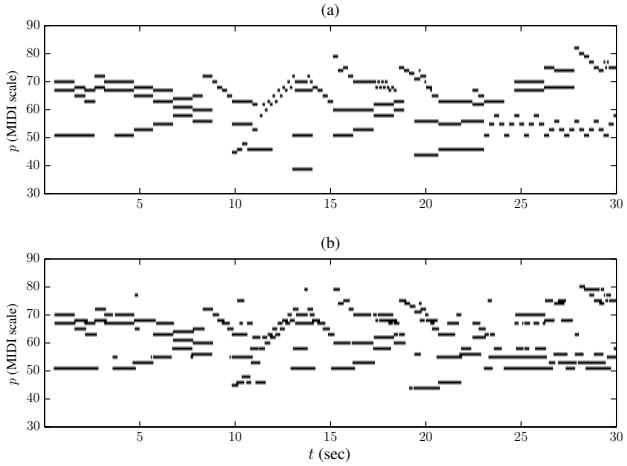


Figure 3. (a) The ground-truth piano-roll of the first 30sec of W.A. Mozart's Piano Sonata K.333, 2nd movement (from the MAPS database). (b) The piano-roll computed from the proposed transcription system.

3. EVALUATION

3.1 Training Data

Sound state templates are extracted for several orchestral instruments, using isolated note samples from the RWC database [10]. Specifically, we extract templates for bassoon, cello, clarinet, flute, guitar, harpsichord, oboe, organ, piano, tenor sax, and violin, using the CQT as a time-frequency representation [15]. The complete note range of the instruments is used, given the available training data. The sound state templates are computed in an unsupervised manner, using a single-pitch and single-instrument variant of the model of (3), where the number of sound states is set to $\mathcal{Q}^{(p)} = 3$.

3.2 Test Data

For testing, we use recordings from three publicly available transcription datasets. Firstly, we used thirty 30sec piano segments from the MAPS database [8], specifically from the 'ENSTDkCl' subset that has been used in the past for multi-pitch evaluation in [4, 17].

We also utilized the woodwind quintet recording used as a development set in the MIREX multiF0 and note tracking task [1]. Instruments present include bassoon, clarinet, flute, horn, and oboe, while manually-aligned ground truth for each instrument track is available online [1].

Finally, we used the TRIOS dataset [9], which includes five multitrack recordings of trio pieces of classical and jazz music. For the current experiments, we used the existing mixes of the multitracks. Instruments included in the dataset are: bassoon, cello, clarinet, horn, piano, saxophone, trumpet, viola, and violin. The dataset includes manually-aligned ground truth with instrument information per pitch. To the authors' knowledge, no transcription results have been reported for the TRIOS dataset.

3.3 Metrics

For evaluating the performance of the proposed system for multi-pitch detection, we employ two sets of metrics: frame-based and note-based ones. For note-based evaluation, we use the onset-based transcription metrics which are used in the MIREX note tracking task [1]. A detected note is considered correct if its pitch matches a ground truth pitch and its onset is within a 50ms tolerance of a ground-truth onset. The resulting note-based precision, recall, and F-measure are defined as:

$$Pre_n = \frac{N_{tp}}{N_{sys}} \quad Rec_n = \frac{N_{tp}}{N_{ref}} \quad F_n = \frac{2Rec_n Pre_n}{Rec_n + Pre_n} \tag{15}$$

where N_{tp} is the number of correctly detected pitches, N_{sys} is the number of pitches detected by the system, and N_{ref} is the number of reference pitches.

For the frame-based metrics, evaluations are performed in a 10ms step as in the MIREX multiF0 evaluations [1], and we use the frame-based precision, recall, and F-measure, which are defined in a similar way to (15) and are denoted as Pre_f, Rec_f, and F_f, respectively.

3.4 Results

Experiments are performed using the proposed system of Section 2 using two variants; a one-stage version (using the update rules and the note tracking step) and the two-stage version presented in subsection 2.3. The proposed EDHMM-based system is compared with the HMM-based system of [2] using the same time-frequency representation and note tracking steps. In all cases, the Markov models were initialized as ergodic, with uniform priors and state transition probabilities.

In Table 1, multi-pitch detection results using the MAPS recordings are shown. It can be seen that using both sets of metrics, the EDHMM-based systems outperform the HMM-based one. It can also be seen that the two-stage version of the system makes a significant improvement in terms of performance. The differences in performance are not as clear using the frame-based metrics, but they are still evident. In all cases, the precision is higher compared to recall (e.g. for the two-stage EDHMM case, $Pre_n = 74.73\%$ and $Rec_n = 64.46\%$), which signifies that there is a larger number of missed detections compared to the number of false alarms. When comparing the reported results with other methods using the same dataset, it can be seen that the proposed system outperforms both the infinite composite autoregressive system of [17] (which reported $F_f = 48.4\%$) and the source-filter NMF model of [4], which reported $F_f = 52.4\%$ (where the best reported performance in [4] was reported using the using the SONIC algorithm [14], reaching $F_f = 58.0\%$). Finally, the note-based accuracy measure for the MAPS recordings is 53.42%; the accuracy reported in [3] for the MAPS-Disklavier dataset was 68.7%, although it should be stressed that in [3] the dataset was also used for training the system.

Results using the MIREX woodwind quintet recording are shown in Table 1; again it can be seen that the EDHMM-based system outperforms the HMM-based one. In the literature, an experiment using the first 30sec of the MIREX recording was made in [16], where $F_f = 62.5\%$. Using the first 30sec in the 2-stage EDHMM system, the frame-based F-measure reaches 66.95%.

Method/Instrument	HMM-based	EDHMM-based
Bassoon	47.72%	41.42%
Clarinet	64.33%	67.68%
Flute	51.18%	57.53%
Horn	39.86%	44.59%
Oboe	23.84%	22.17%
Mean	**45.39%**	**46.68%**

Table 2. Instrument assignment results (F_f) using the first 30sec of the MIREX MultiF0 recording.

Also, using the TRIOS dataset, similar results are reported, as can be seen in Table 1. It should be noted though that there is a large difference between the note-based metrics and the frame-based metrics, which can be attributed to the fact that the TRIOS dataset contains notes with long durations, which get oversegmented in the proposed system (where small gaps do not significantly affect the frame-based metrics).

Finally, we perform experiments on instrument identification using information from matrix $P_t(s|p)$. In the instrument assignment task [11], a detected pitch is considered to be correct if, in addition to pitch and timing constraints, it is assigned to a correct instrument source. We performed experiments using the MIREX woodwind quintet, using a system variant which utilizes templates found in the recording (bassoon, clarinet, flute, horn, oboe). The output (for instrument s) is given by $P_t(p)P_t(s|p)\sum_\omega V_{\omega,t}$. For comparative purposes, we evaluated the first 30sec of the MIREX recording, as in [4], using the frame-based F-measure. Instrument assignment results are shown in Table 2, where it can be seen that the proposed EDHMM-based method performs better compared to the HMM-based one. It can be seen that the best performance is reported for clarinet, which has a relatively different spectral shape compared to the other instruments. It should be noted though that the HMM-based method performs better for bassoon and oboe, while the EDHMM-based method performs better for clarinet, flute, and horn. The reported F_f for the method in [4] is 37.0%, which indicates that the proposed method (which uses pre-extracted spectral templates instead of source-filter models within a spectrogram factorization framework) is more appropriate for the task.

4. CONCLUSIONS

In this paper, we proposed a model for automatic music transcription which models the temporal evolution of notes using pitch-wise explicit duration hidden Markov models, within a spectrogram factorization framework supporting multiple pitch and instrument templates, as well as shift-invariance across log-frequency. It was shown that the tem-

Dataset	MAPS 'ENSTDkCl'		MIREX		TRIOS	
Method / Metric	F_n	F_f	F_n	F_f	F_n	F_f
HMM-based	65.93%	66.41%	63.64%	66.01%	55.94%	67.76%
EDHMM-based	67.12%	66.82%	65.14%	66.42%	56.95%	69.54%
EDHMM-based (2-stage)	68.61%	67.99%	66.60%	66.98%	57.66%	71.17%

Table 1. Multi-pitch detection results (in F_n and F_f) using the three employed datasets.

poral constraints posed by the EDHMMs resulted in improved multi-pitch detection and instrument identification performance when compared to HMM-based constraints. Evaluation results outperformed state-of-the-art multi-pitch detection methods using the MAPS and MIREX datasets. Finally, a proposed two-stage transcription procedure helps in further eliminating transcription errors.

One of the main drawbacks of the proposed method is its computational complexity. Even with independent EDHMMs, the proposed method performs about 60×real-time, which is prohibitive for large-scale experiments or real-time applications. In the future, we will attempt to create computationally-efficient versions of the proposed system using more compact time-frequency representations and by replacing the expensive expectation-maximization algorithm with variational Bayesian methods. Finally, we will expand the existing spectrogram factorization framework in order to introduce additional constraints via musicological models, for example integrating information from chord and key detection for improving multi-pitch detection performance.

5. REFERENCES

[1] Music Information Retrieval Evaluation eXchange (MIREX). `http://music-ir.org/mirexwiki/`.

[2] E. Benetos and S. Dixon. Multiple-instrument polyphonic music transcription using a temporally-constrained shift-invariant model. *J. Acoustical Society of America*, 133(3):1727–1741, March 2013.

[3] S. Böck and M. Schedl. Polyphonic piano note transcription with recurrent neural networks. In *IEEE Int. Conf. Audio, Speech and Signal Processing*, pages 121–124, March 2012.

[4] J. J. Carabias-Orti, T. Virtanen, P. Vera-Candeas, N. Ruiz-Reyes, and F. J. Cañadas-Quesada. Musical instrument sound multi-excitation model for non-negative spectrogram factorization. *IEEE J. Selected Topics in Signal Processing*, 5(6):1144–1158, October 2011.

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society*, 39(1):1–38, 1977.

[6] A. Dessein, A. Cont, and G. Lemaitre. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. In *11th Int. Society for Music Information Retrieval Conf.*, pages 489–494, August 2010.

[7] M. Dewar, C. Wiggins, and F. Wood. Inference in hidden Markov models with explicit state duration distributions. *IEEE Signal Processing Letters*, 19(4):235–238, 2012.

[8] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Trans. Audio, Speech, and Lanhuage Processing*, 18(6):1643–1654, August 2010.

[9] J. Fritsch. High quality musical audio source separation. Master's thesis, UPMC / IRCAM / Telécom ParisTech, 2012.

[10] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: music genre database and musical instrument sound database. In *Int. Conf. Music Information Retrieval*, October 2003.

[11] G. Grindlay and D. Ellis. Transcribing multi-instrument polyphonic music with hierarchical eigeninstruments. *IEEE J. Selected Topics in Signal Processing*, 5(6):1159–1169, October 2011.

[12] P. Grosche, B. Schuller, M. Müller, and G. Rigoll. Automatic transcription of recorded music. *Acta Acustica united with Acustica*, 98(2):199–215, March 2012.

[13] A. Klapuri and M. Davy, editors. *Signal Processing Methods for Music Transcription*. Springer-Verlag, New York, 2006.

[14] M. Marolt. A connectionist approach to automatic transcription of polyphonic piano music. *IEEE Trans. Multimedia*, 6(3):439–449, June 2004.

[15] C. Schörkhuber and A. Klapuri. Constant-Q transform toolbox for music processing. In *7th Sound and Music Computing Conf.*, Barcelona, Spain, July 2010.

[16] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Trans. Audio, Speech, and Language Processing*, 18(3):528–537, March 2010.

[17] K. Yoshii and M. Goto. Infinite composite autoregressive models for music signal analysis. In *13th Int. Society for Music Information Retrieval Conf.*, pages 79–84, October 2012.

[18] S.-Z. Yu. Hidden semi-Markov models. *Artificial Intelligence*, 174(2):215–243, 2010.

A COMPREHENSIVE ONLINE DATABASE OF MACHINE-READABLE LEADSHEETS FOR JAZZ STANDARDS

François Pachet
Sony CSL
pachetcsl@gmail.com

Jeff Suzda
Sony CSL
jeff@jeffsuzda.com

Daniel Martín
Sony CSL
daniel.martin@csl.sony.fr

ABSTRACT

Jazz standards are songs representative of a body of musical knowledge shared by most professional jazz musicians. As such, the corpus of jazz standards constitutes a unique opportunity to study a musical genre with a "closed-world" approach, since most jazz composers are no longer in activity today. Although many scores for jazz standards can be found on the Internet, no effort, to our knowledge, has been dedicated so far to building a comprehensive database of machine-readable scores for jazz standards. This paper reports on the rationale, design and population of such a database, containing harmonic (chord progressions) as well as melodic and structural information. The database can be used to feed both analysis and generation systems. We report on preliminary results in this vein. We get around the tricky and often unclear copyright issues imposed by the publishing industry, by providing only statistical information about songs. The completeness of such a database should benefit many research experiments in MIR and opens up novel and exciting applications in music generation exploiting symbolic information, notably in style modeling.

1. MOTIVATION

Building a *reference* database for music information retrieval is a complex issue. Many databases of audio content have been made available with some success to the research community, raising essential annotation issues [25]. For scores and symbolic information in general, the situation is more problematic. There is a large amount of this information on the net, and many illegal scans of scores (e.g. in pdf format) but, to our knowledge, there is no machine-readable online *reference* database for well-defined corpora, such as jazz standards.

A difficulty when defining a reference database is to define its boundary. In the case of jazz, most composers are no longer active, so it is relatively easy to define such a boundary. For instance, Pepper Adams composed exactly 43 songs; most of Charlie Parker's compositions are known and available in various formats, and the same holds for almost all composers of jazz standards. Such a closed-world approach to jazz standards is key to scholarly and academic work, in particular for *evaluating* operational music systems. Ideally, research experiments involving analyzing and generating jazz compositions should exploit, or apply to, all jazz tunes ever composed, but the absence of such information makes it impossible in practice. As a consequence, many research papers dealing with jazz compositions are based on *ad hoc* databases which are not publicly available ([2], [11-12], [20-21], [23]).

An obvious option to build such a reference database would be to use automatic chord recognition and melodic extraction software on existing audio repositories. There are two problems with this approach. Most importantly, unlike many other musical genres, scores in jazz, called *leadsheets*, play a central role as they represent the "essence" of a tune, harmony- and melody-wise. As a consequence, jazz musicians rarely play the chords as they are written, and part of the game of jazz is precisely to take liberty and interpret the score: unlike classical music, the leadsheet, in general, cannot be deduced from actual performances. Second, the accuracy of chord recognition software is not sufficient to enable fully automatic processes. State of the art methods such as [3], [7] report accuracies in the order of 70%, which is insufficient for our task.

There are numerous attempts at building databases of scores in various genres. For instance, the International Music Score Library Project (IMSLP) assembles scores for classical music composers, but only those in public domain. UCLA's score library proposes many popular music scores, including jazz but it is by no means complete.

2. A REFERENCE CORPUS OF STANDARDS

The notion of jazz standards is ubiquitous in jazz, although not completely well-defined: Jazz standards and pieces that are routinely performed by jazz musicians and widely known to listeners. Most of these songs were composed from the 20s up to the 80s. In practice, jazz standards are often thought of as the songs which appear in the so-called "Fake Books". The most well-known of these is probably the "Real Book", published by Berklee students in the 70s as a reaction to previous Fake Books, which were considered as over simplified to be used by jazz musicians [13]. This book, still widely used today, contains 460 hand-written songs with the melody, the chord sequence, and basic editorial information (composer, style, tempo, and a reference recording of the song).

Since the 70s however, Real Books have evolved significantly. The original Real Book being illegal, several publishers subsequently released other songbooks containing sets of songs for which they obtained or cleared copyrights. The most important publishers are Sher (New Real Books, Volume I to III [26] and Hal Leonard (the Real Book Sixth edition, and the Real Book Volume II, III, IV and V [15]). However, other sources of jazz standards are commonly available through various channels (printed, online as well as illegal). Other notable sources are composer-specific songbooks, which often contain yet different versions of songs, such as the Charlie Parker Omnibook [24] or Michel Legrand song book [14]. As a result, songs appear usually in several song books, with sometimes significant differences. For instance, Figure 2 and Figure 3 show several versions of the song Solar by Miles Davis (or rather, Chuck Wayne, see [1]). Subtle differences are visible concerning chords. In some cases more significant differences appear, including mistakes or different harmonizations.

Finally it is important to observe that, in our experience at least, some songs (such as Body and Soul) are played in almost every jam session, but many others are hardly played at all: all songs are not equally "standard".

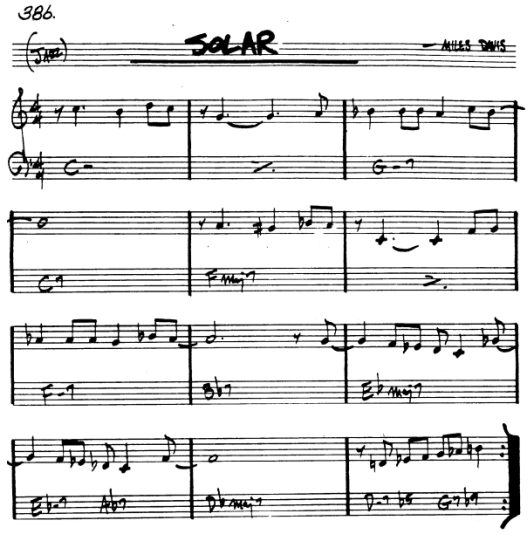


Figure 1. The original Real Book version of Solar.

To summarize, we can point out two important facts about jazz standards that provide us with guidelines:

1) There is no *official version* of any given score unless directly from the author's personal collection, and even then, composers often "update" their compositions afterwards. There are indeed significant differences between scores, depending on the publisher. Differences affect the chord notation used as well as the chords themselves (e.g. their various enrichments) as well as the song structure.
2) The very notion of a standard relies on the existence of songbooks. These books are the medium by which musicians learn and play songs, maintain and evolve the repertoire. The publication of new volumes or new editions of existing volumes impacts the evolution of standards, though on a slow time pace.

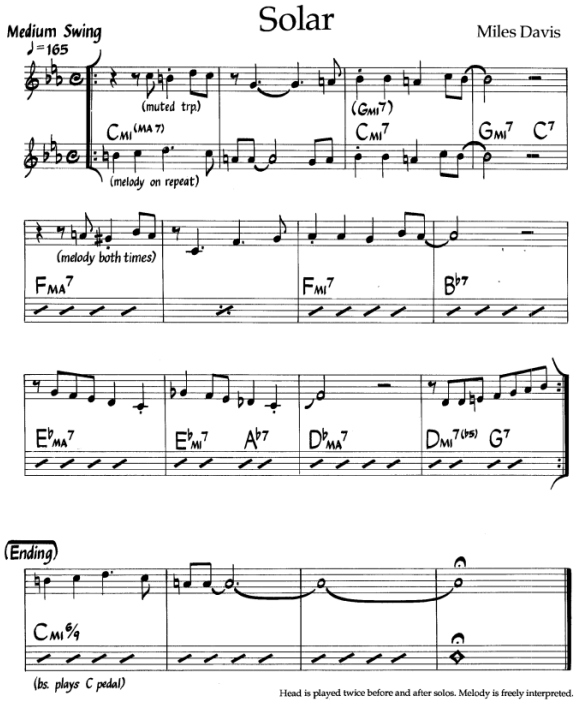


Figure 2. The New Real Book version (Sher) of Solar. Note the different chords (e.g. first chord is C min maj7 instead of C minor), the different chord, and the different structure (ending).

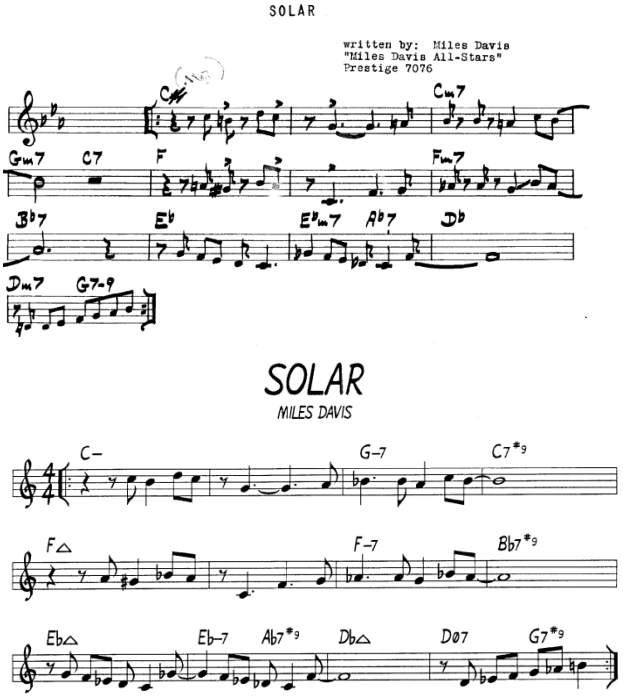


Figure 3. Two other versions of Solar found in popular fake books. Note that none of them can be considered as the official version.

3. AN ONLINE DATABASE

There is a wealth of information about jazz standards on the Internet, but no online database of machine-readable jazz standards exists, to our knowledge. Har-

monic information (chord progressions) is known to be copyright-free so several collections can be found on the web, notably the smartphone application iRealB [10]. But this database does not contain melodies, because of copyright issues, and their content is determined in part by users through a social, collaborative process, with no guarantee on coverage and quality.

3.1 Design: Sources and Songsets

Our database is a web service based on two concepts: *sources* and *songsets*. We define the scope of jazz standards by referring to the already substantial body of work one by reference publishers (such as Sher or Hal Leonard). The primary concept in the database is therefore the "source", which contains the list of songs of a given, published corpus. Figure 4 shows a list of currently entered sources. Sources already contain implicit editorial information concerning the *choice of songs* (publishers want to publish songs that people will actually play), as well as their notation (they try to propose an accurate and consistent notation for musicians). Of course, there are many redundancies in sources, as a popular song will typically appear in various published collections. This redundancy in itself is informative, and can be used, to some extent, to derive automatically information about the popularity of a title, from the viewpoint of publishers. A preliminary analysis of occurrence of songs within 10 sources shows that only one song, *Body and Soul* appears in 8 sources (out of 10), a fact that is confirmed, e.g. by the site jazz-standards.com in which Body and Soul appears as the most popular song to record among jazz musicians. Only 3 compositions occur 7 times (*Here's that rainy day*, *In a Sentimental Mood*, *Bye Bye Blackbird*), and, like *Body and Soul*, they are all famous and routinely performed. More precise information will be enabled as the repository grows, and many analysis can be performed, e.g. on the distribution of popularity in relation with composers, eras, styles, etc.

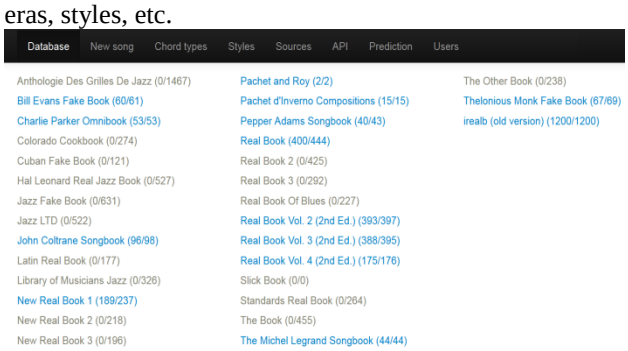


Figure 4. A snapshot of the interface showing the list of currently entered sources (number of completed songs between parenthesis).

Songsets are defined by users, and contain meaningful collections of songs, taken from various sources. Typical songsets are: *all* (the list of all songs in all versions), *bebop* (the complete collection of all compositions by bebop composers such as Charlie Parker or Dizzy Gillespie), *Charlie Parker blues*, the list of all Charlie Parker compositions which are 12-bar blues (see Section 4), *ternary*, the list of all standards in 3/4, etc.

Users define songsets by selecting sources, authors or individual songs, and by filtering them using the information in the database. Information about the redundancy can also be used for specifying songsets (e.g. all songs that appear only once in a given source, or at least 3 times, etc.). Songsets are stored in the database cloud, and can be shared and reused by other users.

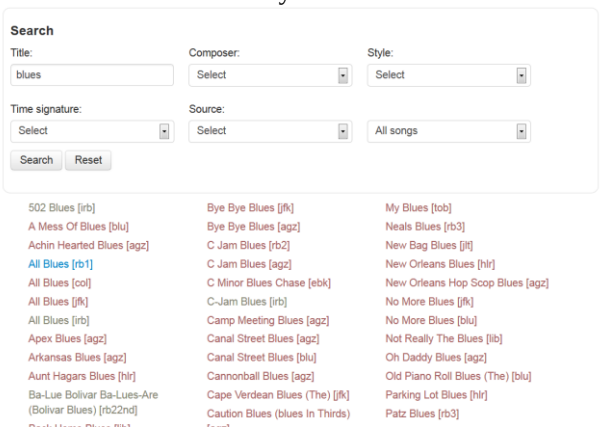


Figure 5. A search tool, here all songs with the word "blues" in the title.

3.2 Song entering

Songs are entered by professional musicians (including the second author), source by source. For each song, a specific online song editor is used, that enables the musician to enter the structure, chords and then melody, as well as basic editorial information (composer, tempo, style, metrics). Average time to enter a song is 3 minutes, but this varies greatly from about 2 to 15 minutes, for complex songs. Note that only basic information about the melody is entered (pitch, quantized position and duration). For instance, the melody of the song Solar, from the Real Book (original) source is illustrated in Figure 6. It can be noted that no typographic information is saved, only the basic MIDI data. This melody is then synchronized to the structure (organization in sections) and chord sequences of the song.

Song enterers do not "copy" the source, but reinterpret it to be stored in the database. Interpretation concerns *chord notation* (see next section) and *structure*. Indeed, one of the problems with extrapolating musical information from a leadsheet is the "folding" problem: Many leadsheets are published in a condensed, folded format - usually a one page leadsheet - of musical information, which is very practical for use in performance situations. However, this is not always the best solution for a machine-readable format. For this reason, some of the compositions are "unfolded" in terms of their form so that there is no ambiguity with regards to repeats, codas, or melodic variations. Of course, such transformations preserve the semantics as both versions describe the same sequence of events (chords and notes).

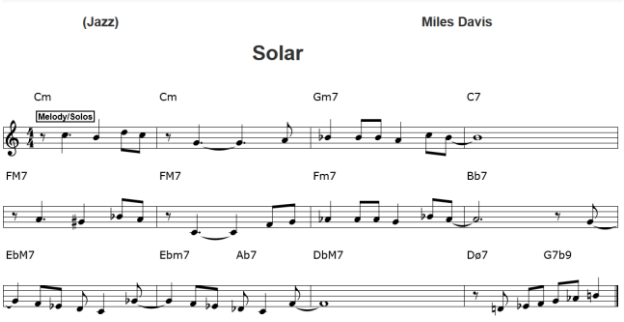


Figure 6. The melody and chord sequence of Solar [Real Book, 5th edition] entered with our online editor.

Figure 7. The song entering process: interpreting a published leadsheet to enter it in a machine-readable format.

Finally a few songs are ignored, either because they contain no melody (*Domino Biscuit* by Steve Swallow) have no time signature (*And now, the Queen* or *Batterie* by Carla Bley), or because the melody is too polyphonic (*Ay Arriba* by Stu Balcomb), and therefore outside the scope of our target (all examples from the original Real Book).

Error checking is performed using two means. First, automatic checks are performed to ensure that the durations of melodies in each bars and section are the same as the corresponding durations of chord sequences. Second, song enterers periodically manually check about 5% random songs entirely (melodies and chords) entered by other song enterers. Manual checking has revealed so far that very little errors are encountered (less than 1% of songs contain errors).

3.3 API and Implementation

The API is a delicate matter. Because we do not own copyrights to the compositions, melodies in particular, we provide an API that only delivers statistical information. The API provides, for a given songset, the following information:

- The chords prior probabilities for songset with id *s*:

http://.../api/getChords.php&songset_id=s returns the list of chords in *s* with their probability:

```
{{"prob": 0.217634, "chord": "Am7"},
 {"prob": 0.119352, "chord": "CM7"},
 {"prob": 0.112842, "chord": "G7"}...
```

- The prior probabilities for pitches occurring in a songset. For instance, query *http://../api/getPitches.php&songset_id=s* would return:

```
{{"prob": 0.251634, "pitch": "G"},
 {"prob": 0.250932, "pitch": "C"},
 {"prob": 0.247842, "pitch": "D"}...
```

- For any prefix of chords, the probabilities of all possible continuation chords, at the order equal to the prefix length. For instance, to get the continuations of Gm7, the query *http://.../api/chords.php?method=getTransitions&chord=Gm7&songset_id=s* would return:

```
{"+5/7": {"prob": 0.537634, "chord": "C7"},
"+5/m7": {"prob": 0.071774,"chord": "Cm7"},
"+5/7b9": {"prob": 0.028494...
```

where for each continuation, we have the distance in semitones between G and the continuation's root (+5 between G and C), type (7, minor7 and 7b9), probability and actual chord name.

- For any prefix of pitches, the list of probabilities of all possible continuations, at the order corresponding to the length of the prefix. For instance, *http://.../api/chords.php?method=getTransitions&pitch=A&songset_id=s* would return:

```
{"-2": {"prob": 0.064516, "pitch": "G"},
 "+5": {"prob": 0.043709, "pitch": "D"}...
```

Additionally, the API provides, for each song in a songset, the histogram of chords and pitches, as well as the joint probabilities of chord and pitches.

To our knowledge, such an API does not violate copyright, as it is, in general, impossible to completely reconstruct a melody or even a chord sequence from this statistical information. This API will, however, evolve, to adapt to the needs of applications and the evolution of copyright policies of the music publishing industry. Songs for which copyright has ceased will be made progressively available to users in their entirety. Chord sequences, in principle not copyrighted, are provided entirely in text format.

Current implementation uses standard web technology HTML/CSS and Javascript in the client side, PHP in the server side with a noSQL database in JSON format. Melodies are stored in musicXML format [19].

3.4 Chord notation and substitution rules

As can be seen by the example, there is no common, reference notation for jazz chords, and sources use different notations [6]. Some works in MIR have addressed the problems of chord notation ([8], [16-17], [28]) but these notations are mostly used for automatic audio chord extraction tasks.

Additionally, within a given notation, there are differences in precision. For instance, a dominant seventh chord can be written simply as “7”, or, in other sources, with additional notes (e.g. “9”, or “dim9”). In order to preserve as much as possible the data accuracy we have chosen to enter sources with chord names that are as close as possible to the chord written in the source, and adding them when the score enterer considered it is not in the current list (we have reached currently a total of 86 chord names, see Figure 8): no effort at consistency or uniformity has been conducted at this step.

Such an approach is obviously not sufficient when several sources are mixed together to form a coherent songlist. In order to cope with this problem (seen here as a *sparsity* problem), we use sets of substitution rules, that

transform chords from their original formulation (e.g. *C 7#4#5*) into a sparser formulation that is significant for the task at hand. For instance, some applications may need to distinguish only between, say, 4 chord types (major, minor, dominant 7^{th}, diminished), while other may need more.

To address this issue, we introduce *transformers:* sets of substitution rules that transform a chord in a source into the most relevant chord name in a given vocabulary. For instance, *C 7#4#5 => C7*, or *DM7#11 => D M*.

Such a use of chord substitution rules can be extended to cope not only with lexical redundancy, but also with some form of semantic equivalence. This problem has been well studied in computer music ([20], [27]) and accepted sets of rules can be easily identified. For instance, many forms of "ii-V7-I" can be considered as more or less equivalent: a dominant chord such as *C 7* can be rewritten as *G min7 / C7*, or even as *G min7 / F# 7*, depending on the degree of precision requested and the task. Such application-dependent considerations can all be handled through sets of substitution rules, defined once and for all by users and shared, like songsets.

(empty)	2	5	6	m
+	7	9	11	13
+7	m6	69	M7	m9
m7	M9	7b9	7#11	aug
Alt	m13	m#5	m69	m11
Dim	7#5	7#9	9b5	7b5
mb6	9#5	7#4	M13	7b6
#11	Sus	7b13	add9	11b9
7alt	6#11	m7#5	M7b9	+7#9
+7b9	m9M7	(b5)	7sus	13b9
9#11	mM7	dim7	9sus	4sus
M7#5	M7#4	m9b5	M9#5	13b5
sus2	sus4	M9b5	M7#9	7#9b5
7#5b5	7#4#5	13#11	M9#11	13sus
7b9#9	7#5#9	pedal	+add9	7b5#9
(#11)	m(M9)	dimM7	7#9#5	7b9b5
M7#11	7b9#5	aug#4	+(b9)	6sus4
m11b5	madd9	5add9	7#5#11	7b9#11
Lydian	7#9#11	7b9b13	Dorian	M7#9b9
m7add4	m7b5#5	(add9)	m7sus4	7b9sus
dim7M7	add9b5	mM7#11	mM7b13	13b9b5
add#11	M13#11	7omit5	Aeolian	m(add9)
13b9sus	+(add9)	m7b5b13	(no3rd)	m(m7M7)
(b9b13)	7b13#11	7b13sus	13b9#11	M7add13
m9add13	m7addM7	Phrygian	M7(?4)	m7(b5b2)
(9, #11)	halfdim7	7susadd3	13(b9b5)	m(omit5)
sus4add9	7b9b13sus	13(b9#11)	m7(omit5)	7susomit5
13(add11)	6#9	M7b5	13#9	m9#11
m7#11	7#5b9	69#11	mb5b13	m13#11
M7#5#11	M7#9#11	add9addb13	madd9add11	halfdim7b9
m7add11add13	halfdim7add11			

Figure 8. The current chord names used in about 12 reference sources.

4. APPLICATIONS

Our database is developed in the context of a large-scale project about the representation of musical style, in particular for popular music. In this context, songsets are considered as concrete representations of a user-defined style. Various style analysis and generation mechanisms, e.g. using the technology of *Markov constraints* [22] can be implemented to generate sequences "in the style of", that also satisfy arbitrary user constraints. An example was exhibited in [22] with the so-called *Boulez Blues*: a 12-bar Blues chord sequence in the style of Charlie Parker blues (the *Parker Blues* songset) that satisfies an "All different" constraint (hence the Boulez label), and is optimally Parkerian, i.e. maximizes its probability w/r the Parker Blues corpus.

Other applications can be developed to exploit this database. Generation algorithms based on statistical information, in particular using *random walk* algorithms can be trivially implemented with our API. Indeed, random walk consists in selecting at random the "next" event (chord or note) using the transition probabilities, given a prefix (the sequence already created), which is exactly what our API provides.

The database is also used for analysis studies. To our knowledge, few studies attempt to assess to what extent composers are recognizable through their chord sequences only, or through their melodies, or both. Attempts to address these issues (e.g. [18-19]) are not comprehensive, nor easily reproducible. Such studies are under way [9], and its results will be made credible only the comprehensive nature of this database.

5. CONCLUSION

We described the motivation and rationale for a comprehensive online database of machine-readable leadsheets of jazz standards[1]. The specification of the database is simple because its goals are very clear: provide a machine-readable representation of melodies and chord progressions as found in reference, published fake books, and following a "closed-world" approach. The database is already being used by several projects dealing with analysis and generation of jazz compositions.

The closed-world approach does not mean that this database effort is to be stopped soon. First, new compositions are regularly been published, such as the European Real Book [5], though not at a pace comparable to that of the Fake Books of the 1970s and after. The contents of such books will be added progressively to the database, which will enable interesting experiments, for instance, regarding the evolution of compositional styles.

We do not infringe on copyrights, because 1° our database does not contain typographical information specific to publishers and 2° we provide an API that prevents reverse engineering to the original sources.

Other sources of editorial information will be progressively added, such as the list of official recordings for each standard, with the audio content when possible, or the exact date of composition, when available.

Our effort can be generalized to other music genres, notably for which leadsheets play such a central role. This concerns for instance large chunks of the Brazilian popular music repertoire such as Bossa Nova or Choros: like jazz, these repertoire are somewhat closed but rich enough musically to deserve such a treatment. Several works have already addressed analysis tasks on partial databases [4]. Most importantly our approach applies to songs that can be reduced to their leadsheet representation without losing their essence.

[1] www.flow-machines.com/lsdb

Our jazz database targets a total of 15 sources (see Figure 4) and 8000 songs (4000 of them unique) by the date of presentation of this paper, obtained through a steady song entering process. With such a consistent mass of information, the first comprehensive style-based jazz composition and analysis systems will, at last, see the light of day. The corresponding research will be easily reproducible. Hopefully, more genres will follow.

6. ACKNOWLEDGEMENTS

This research is conducted within the Flow Machines project which received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 291156.

7. REFERENCES

[1] L. Appelbaum: "Performing Art Blog", http://blogs.loc.gov/music/2012/07/chuck-wayne-sonny-solar, 2012.

[2] J. Biles: "GenJam: A Genetic Algorithm for Generating Jazz Solos", *International Computer Music Conference*, pp. 131-137, 1994.

[3] J. A. Burgoyne, J. Wild, and I. Fujinaga: An Expert Ground Truth Set for Audio Chord Recognition and Music Analysis, *ISMIR*, pp. 633-638, 2011.

[4] G. Cabral and R. Willey: "Analyzing Harmonic Progressions with HarmIn: the Music of Antonio Carlos Jobim", *11th Brazilian Symposium on Computer Music*, São Paulo, 2007.

[5] Europe: *European Real Book*, Sher music, 2012.

[6] M. Granroth-Wilding and M. Steedman: "Statistical Parsing for Harmonic Analysis of Jazz Chord Sequences", *International Computer Music Conference*, pp. 478–485, 2012.

[7] B. de Haas, J. P. Magalhães, F. Wiering: Improving Audio Chord Transcription by Exploiting Harmonic and Metric Knowledge, *ISMIR*, pp. 295-300, 2012.

[8] C. Harte et al: "Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations", *ISMIR*, pp. 66-71, 2005.

[9] T. Hedges, P. Roy and F. Pachet: Predicting the Composer and Style of Jazz Chord Progressions, *submitted*, 2013.

[10] iRealB, smartphone application, http://www.irealb.com, 2013.

[11] R. Keller and D. Morrison: "A Grammatical Approach to Automatic Improvisation", *Fourth Sound and Music Computing Conference*, Greece, 2007.

[12] R. Keller et al.: "Jazz Improvisation Advisor", http://www.improvisor.com, 2009.

[13] B. Kernfeld: *The Story of Fake Books: Bootlegging Songs to Musicians*, Scarecrow Press, 2006.

[14] M. Legrand, *The Michel Legrand Songbook*, Warner Bros. Publications, 1997.

[15] H. Leonard: *The Real Book, Volume I, II, III, IV and V*, Hal Leonard, 2012.

[16] M. Mauch, S. Dixon, C. Harte, M. Casey, and B. Fields: "Discovering Chord Idioms through Beatles and Real Book Songs", *International Symposium on Music Information Retrieval*, 2007.

[17] M. Mauch et al.: "Can Statistical Language Models be used for the Analysis of Harmonic Progressions?" *International Computer Music Conference*, Japan, 2008.

[18] L. Mearns, D. Tidhar, and S. Dixon: Characterisation of composer style using high-level musical features, In *3rd ACM Workshop on Machine Learning and Music*, 2010.

[19] MusicXML 3.0 Specification, MusicXML.com. MakeMusic, Inc. Retrieved 26 February 2013.

[20] M. Ogihara and T. Li: N-Gram Chord Profiles for Composer Style Representation, *ISMIR*, pp. 671-676, 2008.

[21] F. Pachet: "Surprising Harmonies", *International Journal of Computing Anticipatory Systems*, Vol. 4, 1999.

[22] F. Pachet and P. Roy: "Markov constraints: steerable generation of Markov sequences", *Constraints*, 16(2):148-172, 2011.

[23] G. Papadopoulos, G. Wiggins: "A genetic algorithm for the generation of jazz melodies", *STeP, 8th Finish Conference on Artificial Intelligence*, Jyväskylä, 1998.

[24] C. Parker: *Charlie Parker Omnibook*, Atlantic Music Corp, 1978.

[25] G. Peeters, K. Fort: "Towards A (Better) Definition Of The Description Of Annotated M.I.R. Corpora", *ISMIR*, pp. 25-30, Porto, 2012.

[26] Sher Music, *The New Real Book, Volume I, II and III*. Sher Music Co, Petaluma, USA, 2012.

[27] M. J. Steedman: "A Generative Grammar for Jazz Chord Sequences", *Music Perception* 2(1):52–77, 1984.

[28] C. Sutton, Y. Raimond, M. Mauch, and C. Harte: "The Chord Ontology", http://purl.org/ontology/chord, 2007.

MEUSE: RECOMMENDING INTERNET RADIO STATIONS

Maurice Grant
Ithaca College
mgrant1@ithaca.edu

Adeesha Ekanayake
Ithaca College
aekanay1@ithaca.edu

Douglas Turnbull
Ithaca College
dturnbull@ithaca.edu

ABSTRACT

In this paper, we describe a novel Internet radio recommendation system called MeUse. We use the Shoutcast API to collect historical data about the artists that are played on a large set of Internet radio stations. This data is used to populate an *artist-station* index that is similar to the term-document matrix of a traditional text-based information retrieval system. When a user wants to find stations for a given seed artist, we check the index to determine a set of stations that are either currently playing or have recently played that artist. These stations are grouped into three clusters and one representative station is selected from each cluster. This promotes diversity among the stations that are returned to the user. In addition, we provide additional information such as relevant tags (e.g., genres, emotions) and similar artists to give the user more contextual information about the recommended stations. Finally, we describe a web-based user interface that provides an interactive experience that is more like a personalized Internet radio player (e.g., Pandora) and less like a search engine for Internet radio stations (e.g., Shoutcast). A small-scale user study suggests that the majority of users enjoyed using MeUse but that providing additional contextual information may be needed to help with recommendation transparency.

1. INTRODUCTION

Prior to the advent of personal computers and the Internet, there were two primary music recommendation technologies: the jukebox and the AM/FM radio. A Jukebox was a common feature of many social spaces such as bars and diners. Using a jukebox, an individual could chose from a small set of *on-demand* songs to play for the rest of the people in the nearby vicinity. AM/FM radios were found in more personal spaces such as the home or car. However, listeners were connected to one another through a common stream of music that was *broadcast* over the air waves. This gave popular, trend-setting DJs the opportunity to be heard by millions of listeners at the same time.

Today, we see a number of new music recommendation technologies emerging as a result of the availability of personal computers, mobile devices and the Internet. We can generally classify them as being similar to a jukebox, AM/FM radio or a hybrid of the two (see table 1). Celestial jukeboxes [2] such as Apple iTunes [1] and Spotify [2] provide users with instant on-demand access to millions of songs at the click of a button. Internet radio allows computer users to listen to broadcasts of songs or programs from traditional radio stations like NPR and BBC or through radio aggregators such as Shoutcast [3]. AccuRadio [4], and Live365 [5].

The hybrid of these two digital technologies is *personalized* Internet radio which allows users to listen to a personalized stream of music based seed artists or semantic tags (e.g., genres, emotions). Popular examples include Pandora and Slacker. These systems are like jukeboxes in that a user has some control over which songs are selected, but like radio in that there is some element of serendipity in that the user cannot predict exactly what will be played ahead of time. In this paper, we describe a system called MeUse which attempts to harness the strengths of Internet radio but functions more like personalized Internet radio.

The core of our system relies on data collected using the Shoutcast API [6]. Shoutcast is a music service that aggregates 50,000 Internet radio stations, many of which are curated by human DJs. They provide a simple user interface that allows users to search by artist or genre. This search returns a list of stations that are currently playing the artist or genre of music that a user is seeking. They also provide metadata for each station which includes the station's web address, the number of current listeners, bit rate and stream format. Using the web address, users can launch a 3rd party media player (e.g., Apple iTunes, Winamp [7], VLC [8]) to connect to the radio stream.

One problem with Shoutcast is that they may not recommend any stations for a given seed artist if no station is currently playing that artist. They may also return too many stations if the seed artist is currently very popular (e.g., a search for Rihanna might return over 100 stations.) This causes the *paradox of choice* in which increasing consumer choices can also increase the chance of user dissatisfaction [8]. A user may also not know which station to

[1] http://www.apple.com/itunes/
[2] https://www.spotify.com
[3] http://www.shoutcast.com/
[4] http://www.accuradio.com/
[5] http://www.live365.com/
[6] http://wiki.winamp.com/wiki/SHOUTcast_Radio_Directory_API
[7] http://www.winamp.com
[8] http://www.videolan.org/index.html

Technology	Discovery Mode	Delivery Medium	Examples	Advantages
Internet Radio (analogous to AM/FM radio)	Passive	Broadcast	NPR, BBC, Shoutcast, AccuRadio	Often Curated by Human DJs, Ease of Use, Serendipity
Personalized Internet Radio	Passive	Automatic Playlist	Pandora, Slacker, Jamendo	Personalization, Ease of use, Serendipity
Celestial Jukebox (analogous to a physical jukebox)	Active	Metadata Search	iTunes, Spotify, Tomahawk	Customized Playlists, Immediate Gratification, Personalization

Table 1. Comparison of Music Recommendation Technologies

choose based on the limited metadata that is provided for each radio station. The ability to search by genre is also limited because radio stations are only allowed to have one genre (even though stations often play music from a diverse set of genres) and the genre label may be vague, rarely updated, or simply inaccurate.

Historically, another problem with Internet radio aggregators is they used to forces the user to use a web browser for recommendation and a 3rd party audio player for listening. This meant that it took a relatively long time before a user could start listening to music and switch between stations. More recently, web technologies such as the VLC web plugin and the SoundManager2 JavaScript API [9] have made it possible to stream radio stations directly from the browser.

In this paper, we are interested in improving Internet radio station recommendation both in terms of backend recommendation algorithms as well as developing a better frontend user experience. Specifically, we use the Shoutcast API to collect historical data about the artists that each station plays over time. We use this data to populate an *artist-station* index that is similar to the term-document matrix of a traditional text-based information retrieval system but where the documents are stations and the terms are artists. When a user wants to find stations for a given seed artist, we check the music index to determine a set of stations that are either currently playing or have recently played that artist. These stations are grouped into three clusters and one representative station is selected from each cluster. Clustering is intended to promote diversity among the stations that are returned to the user. In addition, we provide additional information such as relevant tags and commonly played artists to give the user more contextual information about the recommended stations. Finally, we incorporate an embedded audio player directly into our website so that users do not need to use an external 3rd party media player.

2. RELATED WORK

While we have been unable to find related work on the specific task of Internet radio station recommendation, our work is functionally equivalent to a generic text-based search engine in that the main data structure is an inverted index and we rely on a vector space model to access relevance and cluster our data [4]. That is, each radio station is represented as a vector over a vocabulary of artists. When given a seed artist, we can rank stations by the dimension corresponding to that artist. We are also able to cluster stations once they are each represented as a vector.

There has been considerably more work on using historical playlists from Internet radio stations as data for studying music similarity and automatic playlist algorithms [1, 3, 5, 6]. Although it is not the focus of this work, our artist-station index can also be used to calculate artist similarity if we instead think each vector corresponding to an artist over a vocabulary of stations.

3. SYSTEM OVERVIEW

When designing MeUse for Internet radio recommendation, we focus on three information retrieval concepts: *Relevance, Diversity and Transparency.* In this sections we describe both the backend recommendation algorithm and frontend user interface in terms of these important design concepts.

3.1 Backend Recommendation Architecture

In this subsection, we discuss how we collect data, use this data to find a set of relevant stations, clustering these stations, and then select stations that are recommended to the user. An overview of our system architecture is shown in figure 1.

3.1.1 Data Collection

The set of artists in the database was initialized by downloading the top 100 tags on Last.fm, and then downloading the top 100 artists for each tag. This provided us with a diverse set of 4460 artists. The set of tags was initialized by downloading the set of tags for all of these artists. Once downloaded, the set of tags was pruned by first removing weak song-tag associations (e.g., a Last.fm tag score < 5) and then making sure that each tag was associated with a minimum of 5 artists.

We then grow a set of Internet radio stations by querying the Shoutcast API multiple times for each of our artists.

[9] http://www.schillmania.com/projects/soundmanager2/

That is, Shoutcast returns a set of stations that are currently playing a given seed artist. For each of these stations, we increment a counter (e.g., a cell in our artist-station index matrix) each time we see that the stations is playing the artist. If we have never observed the station in the past, it is added to our set of stations.

In addition, whenever a user searches for an artist using our frontend user interface, we use the given artist as a query to the Shoutcast API. The results are used to further grow the set of artists, the set of stations, and increment the values in the artist-station index. Finally, we apply a daily decay to the values in the artist-station index so that newer observation have more weight than older observations.

3.1.2 Relevance

Once we had gathered the above data, we used the following algorithm to recommend stations for a given seed artist. To begin, we selected a set of between 10 and 30 *candidate* stations. These stations have played the seed artist the highest number of times in the past or are currently playing the seed artist according to Shoutcast. If we have too few candidate stations, we find the top ranked similar artist to the seed artist according to Last.fm and use that artist to find additional candidate stations. If we have too many candidate stations, we rank order stations by the number of times they have played the seed artist but also taking into account observation decay as described above.

3.1.3 Diversity

Next, we use the information that is stored in our artist-station index to represent each candidate stations as a vector over our vocabulary of artists (e.g., columns of the artist-station matrix). These vectors are grouped into three clusters using the k-mean algorithm [7]. We then selected a representative station from each cluster by selecting the station with the highest listen count while giving stations currently playing the seed artist priority. This ensures that the station is not only relevant but also important.

3.1.4 Transparency

To make our station recommendation more transparent, we provide three representative artists and three representative tags for each of the recommended station. To select representative artists for a given station, we ranked artists according to the difference between the play count on that station and the sum of the play counts for that artist on the other two stations. By this method, we select representative artists that *differentiate* the recommended stations from one another.

The three representative tags for a recommended station are found by first finding all of the tags for all of the artists that are played on that station and then removing the tags that are also associated with artists who have been played on the other two recommended stations. The remaining tags are then rank ordered by the average Last.fm artist-tag score for all artists associated with the station. Again, this algorithm has been designed to pick tags that differentiate the three recommended stations rather than select the most representative tags for the station.

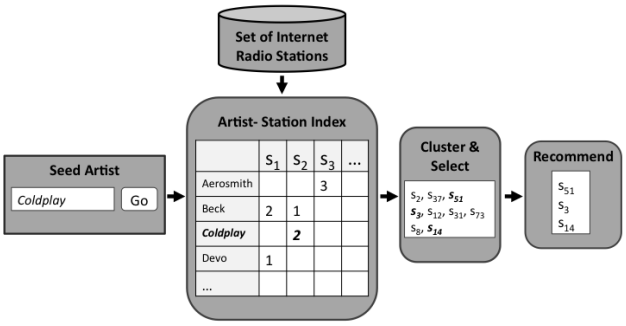


Figure 1. Backend System Architecture. A *artist-station index* is created by counting the number of times an artist is played on a station. When a user provides seed artist (e.g., Coldplay), the index is used to find a set of relevant station. These stations are clustered and a representative station is selected from each cluster.

3.2 Frontend User Interface

The web-based user interface for MeUse is shown in figure 2. We wanted the interactive experience to be more like a personalized Internet radio player (e.g., Pandora) and less like a search engine for Internet radio stations (e.g., Shoutcast). This is accomplished in a few ways.

First, after a user has entered a seed artist in the search bar, only three stations are recommended to the users based on clustering and station selection as is described in the preceding subsection. We provide a clear and concise snippet of information for each station. This includes the name of the station and other important metadata that is provided directly from Shoutcast (e.g., current number of listeners, bit rate, audio format). In addition, we provide the lists of representative artists and tags that differentiate the recommend stations.

We also provide an embedded VLC audio player that is hidden from the user but can be manipulated by the user through various control mechanisms found on the page (e.g., play/pause, volume). In addition, the user can easily switch between the recommended stations, request new recommendations, or change the seed artists. While the VLC plugin is useful for removing the need for an external 3rd party player (e.g., iTunes, Winamp), it does require the user to have the (free) pluggin be installed for their browser. In the future, we expect that web technologies (e.g., HTML5) will allow for more seamless streaming of Internet radio directly through the web browser.

4. EVALUATION

To evaluate MeUse, we first explore how well our algorithm is able to recommend a diverse set of Internet radio stations. We then describe a small-scale user study that was primarily directed at evaluating our user interfaces but also allows us to ask questions about our backend recommendation system.

Figure 2. MeUse User Interface

4.1 Exploring Diversity

One of the primary goals of MeUse is to dramatically limit the number of recommended stations while providing a diverse set of relevant stations to match the user's interests. We also want to provide users with contextual information, such as representative tags and artists, so that the user can make an informed decision when choosing between the recommended stations. To evaluate this, we designed an experiment that compares how often a representative artist for a station is played on that station verses how often the representative artist is played on one of the other recommended stations.

For the experiment, we randomly selected 100 artists from the set of 500 most popular artists (according to Last.fm) in our database. For each artist, we obtained three recommended stations using MeUse. We then *listened* to each of these three stations for the next two hours by recording the currently playing artist every 10 minutes. Finally, we counted how often the seed artist was played, how often one of the three representative artists for the station was played, and how often one of the six representative artists from the other two recommended stations was played.

The results for our experiment are shown in table 2. While we should have collected on the order of 3600 song-play observations (e.g., 100 artists, 3 stations, 12 observations), we found data collection to be a more noisy process than expected. That is, some stations did not appear to update their "recently playing" information and other stations' "recently playing" information contained information about the station and not about the music currently being played. In both these cases, we ignored the duplicated "recently playing" information. We also found a few cases where a station stopped broadcasting during our two hour observation window which prevented us from collecting some additional song-play observation. In the end, we were able to collect 2627 observations.

	# of Artists	Play Count	$\frac{\text{Play Count}}{\text{\# of Artists}}$
Seed Artist	1	147	147
Representative Artists for Station	3	25	8.3
Representative Artists for Other Recommended Stations	6	30	5

Table 2. Results from diversity experiment after recording 2627 song-play observations.

The results show that stations play the seed artists 5.6% of the time. We also observe that the stations play each of our representative artists 0.32% of the time. While this appears to be rather low, our goal in picking representative artists is to pick artists that differentiate the station from the other two recommended stations rather than simply pick-

ing popular artists that have been played on the station in the past. We also note that this is higher than the 0.19% of times that each of the representative artists from the other two recommended stations are played on the station. However, this is not a statistically significant improvement ($\alpha = 0.18$, one-tailed two-proportion pooled z-test). We suspect that the ability to find better representative artists will improve as we are able to collect more data to populate the artist-song index. [10]

4.2 User Study

To evaluate the usability of MeUse we conducted a small-scale user study of our interface. The study involved 20 college-aged individuals who were asked to play around with the interface and then fill out a short survey about their experience. About half of the test subjects were observed in our lab while using the system. The other half were asked to use MeUse in their own environment. Of the ones that we observed, all users seemed to find MeUse easy-to-use, were quickly able to listen to music through the embedded player, and seem to enjoy switching between the recommended stations.

No	Not Really	Sort of	Mostly	Definitely
0	3	3	9	4

Table 3. Relevance: Were the recommend stations that were relevant to you?

No	Not Really	Sort of	Mostly	Definitely
2	3	5	7	2

Table 4. Transparency: Did we give you enough information to make a clear choice between the 3 stations we recommended?

In terms of our ability to recommend Internet radio stations, 70% of the test subjects stated that the recommended stations were mostly or definitely relevant (see table 3) but only 45% of users felt that they were given enough information to make an informed decision on which of the three stations to choose (see table 4). This suggest that we need to think about additional ways to provide the user with contextual information about the stations. For example, one test subject suggested adding a point-rating system for stations. The test subjects did indicate that the stations that were recommended were diverse in nature (see table 5) and that, in general, they enjoyed using MeUse to listen to Internet radio (see table 6).

5. DISCUSSION

In this paper we described MeUse as a complete Internet radio recommendation system. The results of our small-scale user study suggests that the system shows promise but additional user testing is required. In particular we planned to do extensive A/B testing to isolate specific aspects of our system (e.g. UI design, recommendation algorithm). This will include both observing a small number of users in our lab as well as large-scale and long-term user studies in natural user enviroments.

[10] At the time of submission, we have only been able to search the Shoutcast API approximately 5 times for each of the ∼4,500 artists in our database. This is because Shoutcast limits the number of query's one can make on a daily basis.

No	Not Really	Sort of	Mostly	Definitely
1	1	3	11	3

Table 5. Diversity: Were the three stations we recommended different enough from each other to make selecting a station meaningful?

No	Not Really	Sort of	Mostly	Definitely
2	1	2	6	8

Table 6. Overall: Did you enjoy using MeUse to listen to Internet radio?

We also would like to further develop MeUse by exploring additional ways in which we can make MeUse more like a personalized Internet radio player. This will include allowing users to be able to rate stations and using collaborative filtering to improve our station recommendation algorithm. Finally, we plan to explore modifying our artist-tag index to benefit from common text retrieval techniques (e.g. tf-idf) to further improve recommendations.

Acknowledgments: Steven Lam help implement MeUse. This research was supported by NSF Award IIS-1217485.

6. REFERENCES

[1] N. Aizenberg, Y. Koren, and O. Somekh. Build your own music recommender by modeling internet radio streams. In *WWW*, 2012.

[2] P. Lamere and J. Donaldson. Tutorial on using visualization for music discovery. In *ISMIR*, 2009.

[3] F. Maillet, D. Eck, G. Desjardins, and P. Lamere. Steerable playlist generation by learning song similarity from radio station playlists. In *ISMIR*, 2009.

[4] C.D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[5] B. McFee and G. R. G. Lanckriet. The natural language of playlists. In *ISMIR*, 2011.

[6] J. Moore, S. Chen, T. Joachims, and D. Turnbull. Learning to embed songs and tags for playlist prediction. In *ISMIR*, 2012.

[7] 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.

[8] B. Schwartz. *The paradox of choice*. HarperCollins e-books, 2009.

IMPROVED AUDIO CLASSIFICATION USING A NOVEL NON-LINEAR DIMENSIONALITY REDUCTION ENSEMBLE APPROACH

Stéphane Dupont
University of Mons
stephane.dupont@umons.ac.be

Thierry Ravet
University of Mons
thierry.ravet@umons.ac.be

ABSTRACT

Two important categories of machine learning methodologies have recently attracted much interest in classification research and its applications. On one side, unsupervised and semi-supervised learning allow to benefit from the availability of larger sets of training data, even if not fully annotated with class labels, and of larger sets of diverse feature representations, through novel dimensionality reduction schemes. On the other side, ensemble methods allow to benefit from more diversity in base learners though larger data and feature sets. In this paper, we propose a novel ensemble learning approach making use of recent non-linear dimensionality reduction methods. More precisely, we apply t-SNE (t-distributed Stochastic Neighbor Embedding) to a large feature set to come up with embeddings of various dimensionality. A k-NN classifier is then obtained for each embedding, leading to an ensemble whose estimates can then be combined, making use of various ensemble combination rules from the literature. The rationale of this approach resides in its potential capacity to better handle manifolds of different dimensionality in different regions of the feature space. We evaluate the approach on a transductive audio classification task, where only part of the whole data set is labeled. We confirm that dimensionality reduction by itself can improve performance (by 40% relative), and that creating an ensemble through the proposed approach further reduces classification error rate by about 10% relative.

1. INTRODUCTION

Feature transformation and dimensionality reduction approaches have attracted a lot of interest as pre-processors in classification problems, including in the area of multimedia information retrieval. In general, they are able to reduce correlation of feature dimensions as well as noise. They also help to tackle the issues related to the curse of dimensionality, to avoid over-fitting on the training data, and to reduce the computational cost of the classification scheme.

Unsupervised non-linear dimensionality reduction schemes have shown their benefit in semi-supervised learning problems for classification [11, 23]. This follows from the so-called cluster and manifold assumptions. In the first, it is assumed that data samples are organized into distinct clusters and that samples from different classes belong to different clusters. In the second, it is assumed that data samples from different classes occupy distinct manifolds of lower dimensionality in the original feature space. If one of these assumptions holds, the more the available data can be used, the better these cluster or manifold structures can be discovered, to the benefit of classification accuracy.

Ensemble approaches constitute another popular research theme in machine learning and classification problems. They consists in training a set of diverse estimators (referred to as base learners) for the same problem, and combine their estimates or decisions when new data samples have to be classified [24]. These have become popular with approaches such as bagging [2] and boosting [18] (f.i. AdaBoost), to name a few.

Intuitively, in order to gain accuracy when combining such estimators, these have to be different. This has led to research into generating base learners that are as diverse as possible, by acting on one or more of the factors that will have an effect on the end result of the learning process [4, 24]. There is literature on using different subsets of the training data for each member of the ensemble, on manipulating the parameters involved in the training (up to the architecture of the individual learners), or on using different output representations.

Methods for altering the input feature space have also been researched. In this paper, we propose to make use of recent developments in the area of unsupervised non-linear dimensionality reduction in order to alter the feature space and extract low-dimensional embedding whose dimensions can be used to define multiple classifiers. Their estimates are then combined through an ensemble scheme.

Previous attempts and popular methods in creating ensemble diversity through feature transformations and dimensionality reduction will first be summarized in Section 2. Section 3 will then describe the proposed approach,

This research has received funding from Région Wallonne (Belgium) through the Numediart long-term research programme (grant nbr. 716631) and the MediaWorkflows project (grant nbr. 1117549), and from the European Union Seventh Framework Programme through the i-Treasures project (grant agreement no. 600676).

and in particular introduce the non-linear dimensionality reduction algorithm that have been applied (t-SNE, or t-distributed Stochastic Neighbor Embedding), the way it is used to obtain multiple classifiers, and the combination rules used to obtain the ensemble decision. We then apply this method on a use case of interest to the creative community concerned with the classification of musical instrument loops used in rhythmic music composition/production. Section 4 presents the experimental protocol and evaluation metrics, as well as the experimental results, together with a discussion. We conclude the paper in Section 5.

2. ENSEMBLE METHODS AND FEATURE MANIPULATIONS

If earlier proposals in ensemble learning relied on selecting different subset of the training data (bagging or boosting) for each ensemble member, subsequent research has explored various approaches for transforming and manipulating the available feature set, including feature selection or more generally supervised and unsupervised dimensionality reduction. This is summarized in chronological order in the following paragraphs.

With the Random Subspace (RS) method [9, 17], random subsets of the original features are presented to the classifiers. This was followed by a more general approach called Random Forests (RF) [3], combining with the idea of bagging, to end up with ensembles of classifiers (initially decision trees, hence the name "forest") constructed from random samplings on both features and data.

Selecting features after PCA has been proposed in [12], with ensembles where each constituent classifier is trained on a user-determined number of principal components. Supervised dimensionality reduction approaches have also lead to some ensemble learning trials. In [13], Input Decimation (ID) is proposed. Its goal is to decouple the classifiers by exposing them to different features. The method does so by training N classifiers (N being the number of classes of the problem) and selecting for each the input feature dimensions (a user-determined number of them) having the highest absolute correlation to the presence or absence of the corresponding class. In [14], the ID-based approach was shown to compare favorably with the PCA-based approach. One explanation is that unsupervised dimensionality reduction approaches such as PCA are not well suited for finding features useful for classification as they totally disregard class information. Remember however that random or unsupervised feature selection also work in some contexts.

Rather than selecting feature dimensions randomly as in RS, or making use of feature transformations, simply projecting on randomly defined axes as also been proposed with the Random Projections (RP) approach [7, 20].

In [16], the ideas of RS and PCA are combined to lead to Rotation Forests (RotF), where the feature set is randomly split into a number of subsets and PCA is applied to these. Diversity-error diagrams revealed that RotF-based ensembles construct individual classifiers which are more accurate than these in AdaBoost and RF, and more diverse than these in Bagging, sometimes more accurate as well.

More recently [1], it is proposed to make use of Diffusion Maps (DM) [10], a non-linear dimensionality reduction scheme, and to develop classifiers based on the transformed space dimensions. The approach is compared to RP, RS and RF methods cited above, as well as Bagging and Boosting (through the AdaBoost algorithm). A multi-strategy approach combining DM and Boosting was shown to be superior to other algorithms in many cases.

Ensemble methods can sometimes look like an art. This is without accounting for the theoretical considerations and developments that participate to this research area. One area concerns the study of so-called diversity metrics and diversity generation approaches. All these are however out of the scope of this paper, and the interested reader may refer to recent literature for more information [19, 24].

3. PROPOSED APPROACH

In this paper, we propose to make use of recent developments in non-linear dimensionality reduction approaches in order to obtain several sets of features enabling the development of an ensemble of classifiers. This hence follows up on the literature summarized in the previous section. An earlier proposal was indeed reported in [1], with the use of Diffusion Maps. Here, we will make use of t-SNE (t-distributed Stochastic Neighbor Embedding), a recent method. As explained in [22], t-SNE is less susceptible than other classical approaches (including Diffusion Maps) to assigning much higher importance to modeling the large pairwise distances than the small ones. Hence, it is better at retaining the local structure, which is definitely thought to be beneficial in visualization but also classification problems.

In a previous paper [6], we showed on two semi-supervised classification tasks that t-SNE (even when reducing to a very low dimensional space) can perform as well and sometimes even better than classification in the original high-dimensional feature space.

3.1 Dimensionality Reduction using t-SNE

The popularity of approaches derived from Multidimensional Scaling (MDS) has inspired variants, in particular through methods attempting to preserve local properties of the data in a "softer" probabilistic fashion. In particular, SNE (Stochastic Neighbor Embedding) tries to preserve neighborhood identity [8]. It does so using a cost function that favors the probability distributions of points belonging to the neighborhoods of other points to be similar in the high-dimensional space and in its low-dimensional embedding. In the original formulation, a Kullback-Leibler (KL) divergence is used to measure that similarity, and probabilities for a sample to belong to a neighborhood of another one is based on Gaussian distributions.

More recently, a symmetric version of SNE has been proposed. It has also been proposed to use a Student-t distribution rather than a Gaussian distribution to com-

pute the similarity between pairs of samples in the low-dimensional space. These modifications have lead to the t-SNE [22] method. The t-Student heavy-tailed distribution in the low-dimensional space significantly alleviate the so-called "crowding" problem observed with SNE where far away data samples, for instance low density areas in between natural clusters, come close together in the low-dimensional embedding.

In details, we first estimate the (symmetric) probability that sample x_i in the high-dimensional space would pick sample x_j as its neighbor using the following expression:

$$p_{ij} = \frac{\exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq l} \exp\left(\frac{-\|x_k - x_l\|^2}{2\sigma_i^2}\right)} \quad (1)$$

where σ_i is the standard deviation of a Gaussian centered on x_i. Similarly, we model the probability that y_i, the low dimensional counterpart of x_i, would take y_j as its neighbor using the following (symmetric) expression:

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}} \quad (2)$$

where the model of proximity is Student-t distributed. t-SNE then proposes to find a representation for which the probabilities q_{ij} are faithful to p_{ij}. This is achieved by minimizing the mismatch between q_{ij} and p_{ij} measured using a KL-divergence:

$$C = KL(P\|Q) = \sum_i \sum_j p_{ij} \ln\left(\frac{p_{ij}}{q_{ij}}\right) \quad (3)$$

If P_i represents the probability distribution of p_{ij} over all data points given point x_i, t-SNE first performs a binary search for the value of σ_i producing a P_i with a fixed perplexity specified by the user, where the perplexity is defined based on the Shannon entropy of P_i measured in bits.

The minimization of the cost function in Equation 3 is performed using a gradient descent method.

In our experiments, the perplexity of the conditional probability distribution was set to 20; and we performed 2000 iterations of gradient descent. Also, we used the refinements proposed in [22], including a momentum term in the gradient descent, as well a tricks referred to as "early compression" and "early exageration" in [22].

3.2 Ensemble of t-SNE Features

By varying the parameters involved in dimensionality reduction through t-SNE, it is possible to come with several diverse feature representations of the data samples, and to obtain a classifier for each of these. In particular, it is possible to either (1) alter some of the meta-parameters of t-SNE learning, and in particular the size of the local neighborhood (perplexity of the conditional probability distribution), (2) alter the number of dimensions preserved by t-SNE, (3) alter the high-dimensional input features used as input to t-SNE, and in particular select different subsets from the initial feature set, for instance, as in the Random Projection approach, or through more principled feature groupings.

Here, we have been using the later two approaches. Classification tasks can benefit from dimensionality reduction, which is sometimes presented as enabling the reduction of noise and unimportant details in the data, while preserving the multi-dimensional manifold structures. There is however a tradeoff between more denoising through lower dimensional target spaces, and better preservation of the inherent dimensions of the data. The optimal choice may depend on the selected class of the problem, or on the considered regions of the space. One assumption is that ensemble methods making use of classifiers obtained from various choices of target space dimensionality are able to mitigate this tradeoff. Experimental results on the proposed classification task will show that some classes indeed strongly benefit from dimensionality reduction, while others do much less, or not at all. Combining the obtained classifiers leads to improvement over the single best one.

The details of the experimental setup and of the way the multiple classifiers are obtained are provided in Section 4.

3.3 Combination Rules

As soon as the different classifier are available, several approaches are possible for "combining" the individual estimations they provide. Suppose we have K classifiers available. In this work, we consider classifiers that provide estimates of the posterior probabilities of each class. The "sum rule" consists in averaging these posteriors for each class, possibly using a weight dependent on the classifier. It follows from considering classification as a regression problem on posterior probability estimates, and benefit from the literature on ensemble combination through the averaging of various estimates [24]:

$$P(q|x) = \sum_{k=1}^{K} \alpha_k P_k(q|x) \quad (4)$$

where q is the class label, x the feature vector, and $P_k(q|x)$ the posterior probability for class q assigned by classifier k.

The "product rule" consists of a product of probability estimates for each class. If follows from an independence assumption. When one has available several classifiers making use of distinct and statistically independent feature descriptors, the posterior probability of classes can easily be computed from the a priori probabilities of classes and posteriors estimated by the different classifiers. Let $x = (x_1, ..., x_K)$ be the feature vector built from K independent sub-vectors. Bayes rules tells us:

$$P(q|x) = \frac{P(x|q)P(q)}{P(x)} \quad (5)$$

Assuming that the different subparts of the feature vector

are statistically independent, we successively get:

$$\begin{aligned} P(q|x) &= \frac{P(q)}{P(x)} \prod_{k=1}^{K} P(x_k|q) \\ &= \frac{P(q)}{P(x)} \prod_{k=1}^{K} \frac{P(q|x_k)P(x_k)}{P(q)} \\ &= \left[\frac{\prod_{k=1}^{K} P(x_k)}{P(x)}\right] \left[\frac{\prod_{k=1}^{K} P(q|x_k)}{(P(q))^{(K-1)}}\right] \end{aligned} \quad (6)$$

The first term is independent of k, and if the initial independence assumption holds, its value will be 1. In practice, the posterior probability will be estimated based on the second term normalized in such a way that the sum of estimates for all classes is the unity.

Besides averaging, the use of order statistics has also been proposed in the literature [21]. The "maximum rule" consists in approximating the posterior probability for each class using the maximum of the various classifier estimates for this class:

$$P(q|x) = \max_{k=1}^{K} P_k(q|x) \quad (7)$$

The "minimum rule" follows a similar principle:

$$P(q|x) = \min_{k=1}^{K} P_k(q|x) \quad (8)$$

Finally, the "median rule" is expressed as:

$$P(q|x) = med_{k=1}^{K} P_k(q|x) \quad (9)$$

These five approaches have been compared in this work. Majority voting is another popular approach, but it has not been used in this work as it can not benefit from posterior estimates.

4. EXPERIMENTS

As data set, we used a production music library (ZeroG ProPack). This library contains more than ten thousand "loops" and samples of various instruments and music styles. Each soundfile is typically a few seconds long of monophonic or polyphonic sound (f.i. in the case of guitars). We manually annotated the files within 7 classes of instruments: Brass, Drums, Vocals, Percussion, Electric Bass, Acoustic Guitar and Electric Guitar. After discarding more complex sounds or effects, we ended up with 4380 samples to be used in our evaluations.

The experimental work that follows is based on a transductive classification task. It hence considers a closed data set that has to be classified with minimal effort. Part of the data is hence annotated with class labels to guide the supervised machine learning, but all the data set can be used in an unsupervised mode. Transductive learning has many interesting applications [5].

http://www.zero-g.co.uk/

Data sets definitions and labels available as supplementary material at http://www.numediart.org/tools/mediacycle/

4.1 Low-level Features for Audio and Music

A large body of recent work in the music information retrieval literature has been devoted to the design of feature extraction algorithms for the purpose of characterizing, analyzing, searching or classifying audio content. Here, we consider timbral properties. Audio analysis approaches for extracting feature descriptors rely on isolating and analyzing short-term windows of temporal signal (typically around 30 ms long), to end up with one feature vector per window. For representing and be able to classify longer-term signals as used in our experiments, we extracted statistics (up to order 4) from the short-term window feature vectors. From previous research, we ended-up using two groups of features, covering the spectral envelope and the noisiness of the sounds, both being important for characterizing the perceived timbre. The state-of-the-art feature set that we used contains:

- Mel-Frequency Cepstral Coefficients (MFCC) as used in [15], computed using 30 ms frames every 10 ms, using a filterbank of 20 filters covering the audible frequency range, and keeping the first 12 coefficients. To be able to capture the temporal characteristics and statistics of the MFCCs, we actually used as features the MFCCs means along the sample duration, as well as their standard deviation, skewness and kurtosis; the means of the first order temporal derivatives of the MFCCs, as well as their standard deviation, and the means of the second order temporal derivatives of the MFCCs, as well as their standard deviation.
- Spectral Flatness (SF), which is a correlate of the noisiness (opposite of sinusoidality) of the spectrum computed on the same audio frames as MFCCs. It is computed as the ratio between the geometric and arithmetic means of the spectrum energy values. As proposed in [15], the spectrum was divided into 4 sub-bands for computing the flatness: 250-500Hz, 500-1000Hz, 1000-2000Hz and 2000-4000Hz. Here too, we used the mean of the SF over the sound extract duration, as well as its standard deviation, skewness and kurtosis.

4.2 Experimental Protocol

We are interested in semi-supervised transductive classification, where only part of the whole corpus can be annotated with the desired class labels. We hence performed experiments with different percentages of randomly selected labeled data (from 10% to 50%, f.i. 10% means that only 438 samples have been labeled using their instrument class in the production music database). Being unsupervised, t-SNE is always making use of the whole data set however. For each training condition, we ran 100 different training and evaluation batches (with selected labeled data randomized for each of them) for each classification system (either single classifier, or various ensemble configurations). The classifiers (either individual classifiers or classifiers taking part in the ensembles) are using k-NN (with k=5).

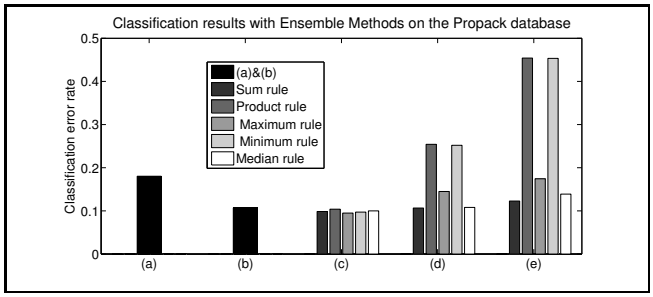


Figure 1. Classification error rates when making use of 10% of labeled data: (a) using the full high-dimensional feature set (b) using a 5-dimensional feature set obtained through t-SNE on the full high-dimensional feature set (best single classifier from different t-SNE based classifiers with various target space dimensionalities) (c) Ensemble composed of 5 classifiers obtained using 1 to 5-dimensional feature sets obtained through t-SNE on the full high-dimensional feature set; various combination rules (d) Ensemble classifier composed of 15 classifiers: 5 target dimensionalities x 3 sub features set (mean, standard deviation, and high-order statistics of the baseline high-dimensional feature set) (e) Ensemble classifier composed of 20 classifiers: 5 target dimensionalities x 4 sub features set (MFCCs, MFCCs first and second derivatives, SF).

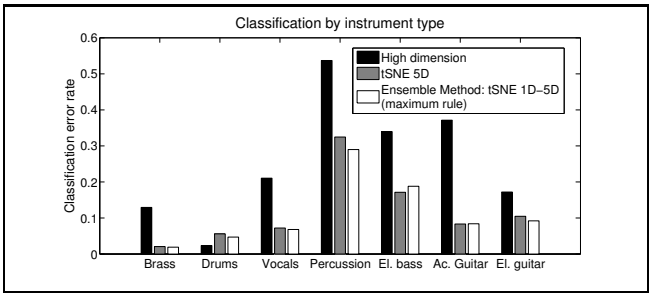


Figure 2. Classification error rates when using 10% of labeled data on each instrument class. Comparison between: (1) using the full high-dimensional feature set, (2) using a 5-dimensional feature set obtained through t-SNE on the full high-dimensional feature set, (3) Ensemble composed of 5 classifiers obtained using 1 to 5-dimensional feature set obtained through t-SNE on the full high-dimensional feature set; maximum rule for combination.

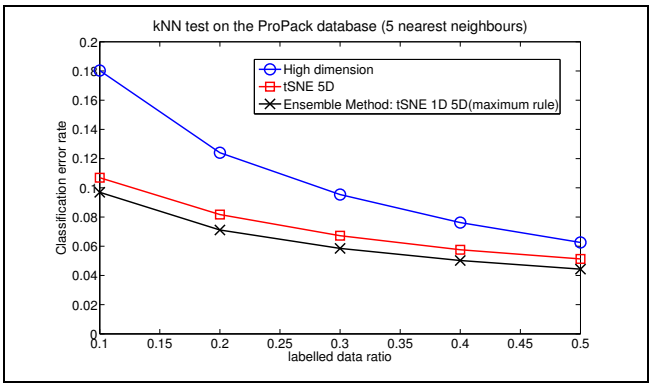


Figure 3. Classification error rates for various proportions of labeled data. Comparison between: (1) using the full high-dimensional feature set, (2) using a 5-dimensional feature set obtained through t-SNE on the full high-dimensional feature set, (3) Ensemble composed of 5 classifiers obtained using 1 to 5-dimensional feature set obtained through t-SNE on the full high-dimensional feature set; maximum rule for ensemble combination.

The baseline classification system uses the high-dimensional features set described earlier. We normalized each feature to zero-mean and unity-variance. We then created various classifiers used standalone, or involved in ensemble configurations according to three principles:

- creating various classifiers by altering the dimensionality of the t-SNE embedding from 1 to 5. This enabled the design of an ensemble of 5 classifiers.

- creating various classifiers by altering both the dimensionality of the t-SNE embedding and the input features of t-SNE. Rather than selecting random subset of feature dimensions as done in the RD or RS approaches, we partitioned the full feature set according to the order of the statistics used to represent the sound files. More precisely, three subspaces were obtained, one gathering the means of the raw feature vectors, a second one for the standard deviations, and a third one gathering the skewness and kurtosis. This enabled the design of an ensemble of 15 classifiers: 5 target dimensionalities x 3 subsets of features.

- similar to the previous approach but where the category of the feature is use to define the feature partitions. More precisely, we split the individual features into four groups: MFCCs, MFCCs first derivatives, MFCCs second derivatives, and SF. This enabled the design of an ensemble of 20 classifiers: 5 target dimensionalities x 4 subsets of features.

4.3 Results and Discussion

In Figure 1, we present classification results for the case 10% of the whole data set is labeled. It shows results for the baseline system, for a system where features are first processed using t-SNE (with dimensionality of 5), as well as for various ensembles and combination rules. We can observe that on this kind of data, an efficient dimensionality reduction scheme is a useful pre-processing step for semi-supervised classification. Classification performance on the reduced dimensional space is indeed better. A 40% relative reduction of the error rate is obtained.

The first proposed ensemble approach yields a further error rate reduction of 10% relative. All five combination rules bring some improvement, but best results are obtained using the simple maximum rule, followed by the sum rule. The two other proposed ensembles are unconclusive, and the product and minimum combination rules perform notably much worse. This can be explained by

the discrepancy in classification performance of the ensemble members: base classifiers using the standard deviations of features in the first case, and using SF features alone in the second are much worse than the other base classifiers (detailed results not reported here). More complex or weighted combination rules may help in those cases.

Overall, the best single classifier is using a 5-dimensional feature set obtained through t-SNE on the full high-dimensional feature set. The best ensemble classifier is composed of 5 classifiers obtained using 1 to 5-dimensional feature sets obtained through t-SNE on the full high-dimensional feature set; and a maximum rule for ensemble combination. We then present more detailed comparisons of these two with the baseline classifier In Figure 2, we observe that some classes indeed strongly benefit from dimensionality reduction, while others do much less, or not at all. As suggested earlier in the text, the tradeoff between reducing the feature space dimension and preserving its representativity may depend on the class and on the region of the feature space. This also suggests further theoretical and empirical work in ensemble approaches that could account for non-uniform intrinsic dimensionalities of the data set manifolds. In Figure 3, we present results for various proportions of labeled data, showing that our conclusions hold when one can obtain labels for a larger part of the data set. With 50% of labeled date, t-SNE allows to reduce the error rate by 18% relative over high-dimensional features, and the ensemble approach further reduces it by 14% relative.

5. CONCLUSIONS

In this paper, we presented a new method for designing multiple classifiers system relying on non-linear dimensionality reduction through t-SNE, together with an experimental study of its performance on an audio-based musical instruments transductive classification task. We first observed that classification performance can be boosted when applying t-SNE as a pre-processing step, even when going down to as low as a few dimensions. Designing multiple classifiers by altering the dimensionality of the t-SNE embedding and combining them using a simple combination rule further improved the results.

These promising initial results invite further work, in particular in the application of other dimensionality reduction schemes and more complex ensemble combination rules, as well as in understanding how ensembles can be used for mitigating the tradeoff between denoising and feature preservation properties. The application of the proposed approach to larger scale data sets can also be the subject of future work, together with experimental evaluation on non-transductive tasks using out-of-sample extensions.

6. REFERENCES

[1] Amir Amit. Ensemble Classification via Dimensionality Reduction. Master's thesis, Efi Arazi School of Computer Science, Israel, 2011.

[2] Leo Breiman. Bagging predictors. Technical Report 421, Department of Statistics, University of California, Berkeley, California, September 1994.

[3] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[4] Gavin Brown, Jeremy L. Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

[5] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[6] Stéphane Dupont, Thierry Ravet, Cécile Picard, and Christian Frisson. Nonlinear dimensionality reduction approaches applied to music and textural sounds. In *Proceeding of IEEE International Conference on Multimedia and Expo (ICME)*, San Jose, California, jul 2013.

[7] Xiaoli Zhang Fern and Carla E. Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *ICML'03*, pages 186–193, 2003.

[8] Geoffrey Hinton and Sam Roweis. Stochastic neighbor embedding. *Advances in neural information processing systems*, 15:833–840, 2003.

[9] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, August 1998.

[10] S. Lafon and A.B. Lee. Diffusion maps and coarse-graining: a unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1393 –1403, sept. 2006.

[11] John A. Lee and Michel. Verleysen. *Nonlinear dimensionality reduction*. Springer, New York; London, 2007.

[12] Christopher J. Merz and Michael J. Pazzani. A principal components approach to combining regression estimates. *Mach. Learn.*, 36(1-2):9–32, July 1999.

[13] Nikunj C. Oza and Kagan Tumer. Dimensionality reduction through classifier ensembles. Technical Report NASA-ARC-IC-1999-124, National Aeronautics and Space Administration, Moffett Field, CA, 1999.

[14] Nikunj C. Oza and Kagan Tumer. Input decimation ensembles: Decorrelation through dimensionality reduction. In *LNCS*, pages 238–247. Springer, 2001.

[15] Geoffroy Peeters. A large set of audio features for sound description (similarity and classification). in the CUIDADO project. Paris, IRCAM, 2004.

[16] Juan J. Rodriguez, Ludmila I. Kuncheva, and Carlos J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1619–1630, October 2006.

[17] Niall Rooney, David Patterson, Alexey Tsymbal, and Sarab An. Random subspacing for regression ensembles. Technical report, Department of Computer Science, Trinity College Dublin, Ireland, 2004.

[18] Robert E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, July 1990.

[19] Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. The MIT Press, 2012.

[20] Alon Schclar and Lior Rokach. Random projection ensemble classifiers. In Joaquim Filipe and José Cordeiro, editors, *Enterprise Information Systems, 11th International Conference, ICEIS 2009, Milan, Italy, May 6-10, 2009. Proceedings*, volume 24 of *Lecture Notes in Business Information Processing*, pages 309–316. Springer, 2009.

[21] Kagan Tumer and Joydeep Ghosh. Linear and order statistics combiners for pattern classification. *CoRR*, cs.NE/9905012, 1999.

[22] Laurens van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[23] L.J.P. van der Maaten, E. O. Postma, and H. J. van den Herik. Dimensionality reduction: A comparative review. Technical Report TiCC-TR 2009-005, Tilburg University, 2009.

[24] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman and Hall/CRC, 2012.

A STUDY OF ENSEMBLE SYNCHRONISATION UNDER RESTRICTED LINE OF SIGHT

Bogdan Vera, Elaine Chew
Queen Mary University of London
Centre for Digital Music
{bogdan.vera,eniale}@eecs.qmul.ac.uk

Patrick G. T. Healey
Queen Mary University of London
Cognitive Science Research Group
ph@eecs.qmul.ac.uk

ABSTRACT

This paper presents a quantitative study of musician synchronisation in ensemble performance under restricted line of sight, an inherent condition in scenarios like distributed music performance. The study focuses on the relevance of gestural (e.g. visual, breath) cues in achieving note onset synchrony in a violin and cello duo, in which musicians must fulfill a mutual conducting role. The musicians performed two pieces – one with long notes separated by long pauses, another with long notes but no pauses – under direct, partial (silhouettes), and no line of sight. Analysis of the musicians' note synchrony shows that visual contact significantly impacts synchronization in the first piece, but not significantly in the second piece, leading to the hypothesis that opportunities to shape notes may provide further cues for synchronization. The results also show that breath cues are important, and that the relative positions of these cues impact note asynchrony at the ends of pauses; thus, the advance timing information provided by breath cues could form a basis for generating virtual cues in distributed performance, where network latency delays sonic and visual cues. This study demonstrates the need to account for structure (e.g. pauses, long notes) and prosodic gestures in ensemble synchronisation.

1. INTRODUCTION

In ensemble performance, musicians rely on a complex mixture of non-verbal communication via visual and auditory gestures (such as breathing) and the inherent timing information present within the acoustic signal of the performance. Together, these cues contribute to the musicians' common perception of musical time, and allows them to synchronize one with another. In certain cases, such as in distributed music performance, some of these cues are disrupted by factors such as network latency, which has been shown to affect synchronisation between musicians and delay video transmissions so much so as to make visual gestures ineffective. We are, therefore, interested in understanding the effects of disruptions of visual communication on ensemble performance, so as to advance research on assistive systems for distributed performance.

This paper presents a study on the effect of line-of-sight restriction between musicians working to synchronize onsets and interact in a performance. The remainder of the paper is organized as follows: Section 2 reviews related work in ensemble interaction and networked performance; Section 3 describes the experimental design; Section 4 presents the analysis; results and conclusions follow in Sections 5 and 6.

2. LITERATURE REVIEW

Musical gesture analysis has been an area of interest for researchers in music cognition, music performance, and human-computer interaction. McCaleb [1] compares ensemble interaction to the communication paradigm (likened to a telephone or postal service), acknowledging that this approach has not yet been critiqued from the perspective of a performing musician.

Leman and Godoy [2] performed a classification of musical gestures, distinguishing gestures that are part of sound production from those that are purely communicative and those which simply accompany music (such as dancing). Lim [3], as a step towards creating a robotic accompaniment system for flute, identified start and end cues which serve to visually mark the onsets and offsets of notes, and beat cues which are used to keep time during sustained notes, all of which were motion based. Eye contact has also been discussed as being important in ensemble synchronisation [4]. Breathing, as a musical gesture, has been touched upon by Vines et al. [5], and mentioned as an important cue in conversation, where it helps in coordinating turn taking.

In the context of network music performance, the effects of audio latency itself have been studied by researchers such as Chafe and Gurevich [6], Chew et al. [7] and Schuett [8]. This research shows that when latencies higher than around 25 ms are present, the tempo tends to decrease and synchronisation is adversely affected. It is not understood what effects visual isolation has on the performance in these cases, though the DIP project reports iniitial explorations of this area with attempts to provide distributed musicians with visual cues via video streaming [9]. They found that video latencies are to be too high for the trans-

mitted gestural cues to be of much use, especially over long distances. Solutions to the problem of latency based on prediction were theorized by Chafe [10], and Sarkar's TablaNet project [11] predicts tabla drum players' strokes ahead of time in order to create the appearance of zero transmission latency. Further work on predicting drum strokes has recently been done by Oda et al. focusing on estimating the velocity of drum mallets with high speed cameras and predicting their impact times [12]. Combining these ideas with Lim's approach of predicting gestures, we hypothesize that the issue of latency in video transmission could be ameliorated using predictive modeling of gestural cues.

3. EXPERIMENTAL SET-UP

Two simple violin-cello duet pieces [1] were composed by Vera for the experiment, and were played by a violin and cello duo under three different line of sight conditions. The participating musicians were both classically trained, and active in chamber orchestras, but had never played together before. The first condition, S1, involved normal performance with no line of sight obstruction, with the musicians located in their preferred positions. In the second scenario, S2, the musicians were made to face in opposite directions, removing line of sight, but allowing auditory gestures such as breathing. In the third scenario, S3, a translucent curtain was placed between the musicians, and their shadows were cast onto it by two bright studio lamps, allowing them to view only each other's silhouettes, with no fine details such as facial expression (see Figure 1). The musicians were asked to play the chosen pieces for the first time, with little rehearsal time.

The pieces were specially designed with a focus on key experimental features. The first piece is relatively easy to play (i.e. the musicians were not expected to greatly improve over time and they required almost no rehearsal to play it). It consists of a sequence of very long notes (lasting two bars at a moderate tempo) followed by equally long pauses. In the middle of the piece, the pauses are replaced by faster rhythmical dialogues between the performers, before returning to the long separations. The aspect explored in this piece is the timing between the musicians at the beginning of each note, where they have to cue each other into a new section without relying on rhythmic information, the only exception being the middle section where the fast paced rhythms are expected to improve synchronisation. The main hypothesis in this case is that lack of visual contact would result in greater asynchrony between the onsets of simultaneously sounded notes, and that asynchrony would be reduced where timing information is carried by rhythmic patterns in the music itself.

The second piece is a similarly slow paced composition, but without pauses. After four bars of solo cello, the two instruments play simultaneous notes, until later in the piece where some counterpoint is introduced between the parts. In this case, our hypothesis was that the presence of a stronger rhythm and lack of pauses will result in less asynchrony, compared to the performance of the first piece, when line of sight is affected.

The musicians were recorded playing 3 takes of each piece (four in the case of the 'no line of sight' recordings, due to extra time at the end of the recording sessions), in each scenario, over two days. Due to time constraints, the musicians played through each scenarios in sequence, and thus some improvement over time is expected as the musicians became accustomed to playing the pieces. Both instruments were recorded with attached pickups, in an attempt to isolate the two instruments as much as possible.

4. ANALYSIS

A quantitative analysis of the difference in time between the note onsets of the performers' simultaneously sounded notes was performed, comparing their performances in the three scenarios. As obtaining reliable note onsets from bowed instruments is difficult with automatic methods, the onsets were hand annotated using Sonic Visualiser [13]. Even when annotating onsets by hand, it can be difficult to determine the exact onset time of soft notes. Notes played by bowed instruments have varying attack times, and one can, for example, choose either the start of the unpitched bowing sound or the moment when a fundamental frequency becomes audible. In this case, the annotation focused on the latter feature, using Sonic Visualiser's adaptive spectrogram to inspect the notes. The resulting set of onset time differences was then analyzed in Matlab, simply by subtracting the onset times of the violinist from those of the cellist for simultaneous score notes.

4.1 First Piece: Long Notes, Long Pauses

For this first piece, the onset annotations for an example recording are shown in Figure 2. In this case the onset annotations separate the piece into four-bar segments. No fast section onsets were considered for the initial analyses. Fast section annotations marking two bar long sections were later added to inspect the effects of rhythmic vs. non rhythmic patterns on synchronization, and they were treated as secondary to the longer notes, allowing a more focused comparison between the onset times of the long notes separated by pauses, and those linked by rhythmic sections.

Asynchrony analysis for the three scenarios is visualized in the box plots in Figure 4, showing the extent of asynchrony, which we define as the unsigned time difference between the onsets of ideally simultaneously sounded notes, from all the recordings in each scenario. The results show a median asynchrony of 52.1 ms in the normal line of sight scenario. This increases to 104.8 ms in the no line of sight scenario, showing a worsening of synchrony. In the partial line of sight scenario, the median asynchrony was 46.3 ms, which is slightly lower than in the normal line of sight scenario. Table 1 contains the p-values of pairwise Kolmogorov-Smirnov tests between the scenarios, showing that the scenario with no line of sight was

[1] Scores available at http://tinyurl.com/nqhq2pp

Figure 1. The two musicians on either side of the shadow curtain in the partial LoS scenario

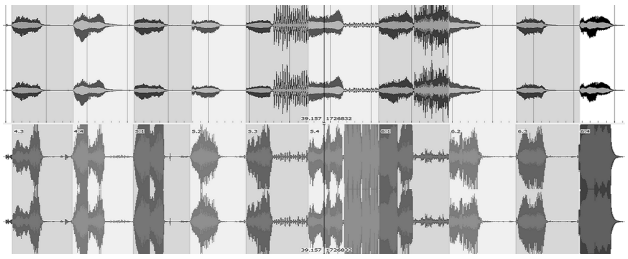


Figure 2. Example segmentations for one take of the first piece (top - cello, bottom - violin)

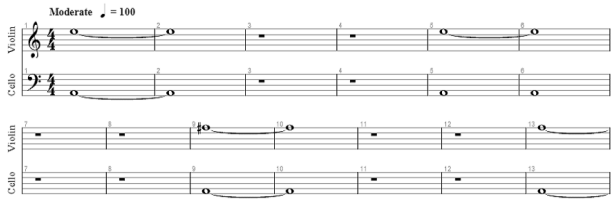


Figure 3. Excerpt of the first piece showing the long notes separated by pauses before the rhythmical section

significantly different from the others, and that the normal and partial line of sight conditions were not significantly different. Figure 5 shows the box plots of onset time differences, without taking the absolute value. The analysis showed that the violinist tended to play ahead of the cellist (i.e. most values are positive). Figure 6 shows median absolute onset time difference per segment, for each scenario. From this graph it is notable that for segments 6, 7 and 8 – the segments linked by rhythmic patterns – the musicians seem to have achieved better synchrony than in the rest of the piece.

Scenario Pair	S1 vs S2	S1 vs S3	S2 vs S3
P-Value	0.0222	0.9360	0.0205

Table 1. Pairwise Kolmogorov-Smirnov p-values between asynchronies in each scenario for the first piece

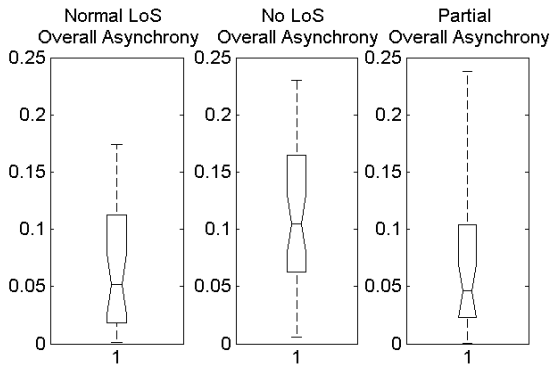


Figure 4. Asynchrony boxplots for each scenario for the first piece (y-axis is in seconds)

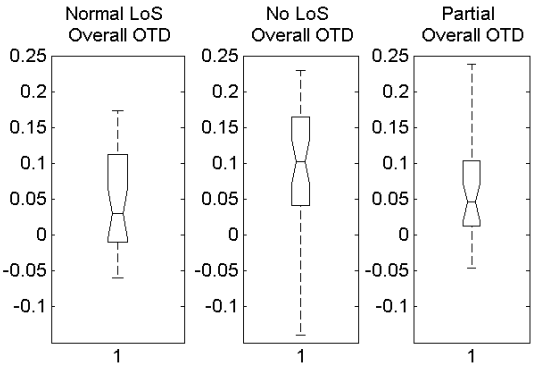


Figure 5. Onset time difference boxplots for each scenario for the first piece (y-axis is in seconds)

4.2 Second Piece: Long Notes, No Pauses, Counterpoint

The same analysis was performed for the second piece. In this case the segments examined were chosen to correspond with all note onsets. Because the violin and cello parts contain many notes that do not have simultaneous onsets, segmentation points from each part were replicated in the other part by automatically choosing time points at appropriate note subdivisions between adjacent segmentations. This provided a set of estimated segmentations based on available data ensuring that both parts have com-

Scenario Pair	S1 vs S2	S1 vs S3	S2 vs S3
P-Value	0.6055	0.0234	0.1072

Table 2. Pairwise Kolmogorov-Smirnov p-values between asynchronies in each ccenario for the second piece

parable segmentation points.

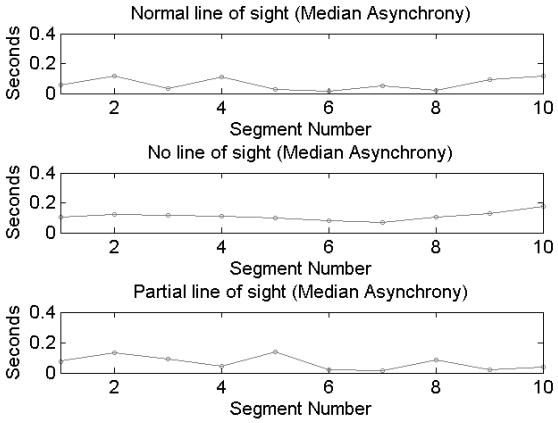


Figure 6. Asynchrony against segment number for the first piece

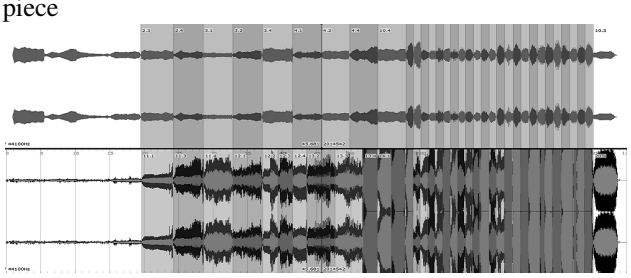


Figure 7. Segmentations for the second piece (top - cello, bottom - violin)

Although comparing a real onset with an estimated one does not give a precise value for note onset synchrony, it does however give an indication of the performers' degree of synchronisation to each other's timing, i.e. a note played by the cellist that starts in the middle of the violinists' held note would have its onset time compared to the calculated middle of the violin's two closest adjacent note onsets. The first four notes, which are played only by the cellist are not annotated. The segmentation points (before the addition of estimated segmentations) are shown in Figure 7.

Unlike for the previous piece, the median asynchrony decreases with each scenario, indicating that the effect of the musicians getting better at playing the pieces was more significant than that of reduced line of sight. The median asynchrony was 80 ms for the baseline scenario, 74.7 ms for the second, and 59.6 ms for the third. The paired Kolmogorov-Smirnov test results, presented in Table 2, show that there was no significant worsening caused by reduced line of sight. We instead see a significant improvement of synchrony between the first and last scenarios.

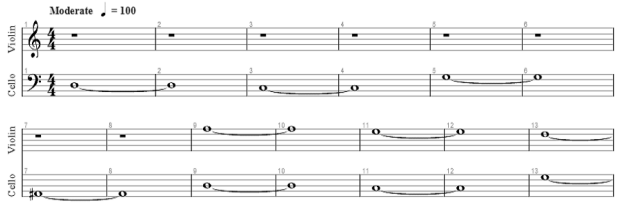


Figure 8. Excerpt of the second piece showing long notes (w/o pauses) in the cello joined by long notes in the violin

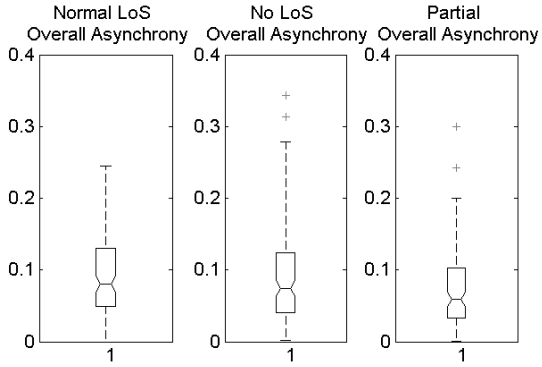


Figure 9. Asynchrony boxplots for each scenario for the second piece (y-axis is in seconds)

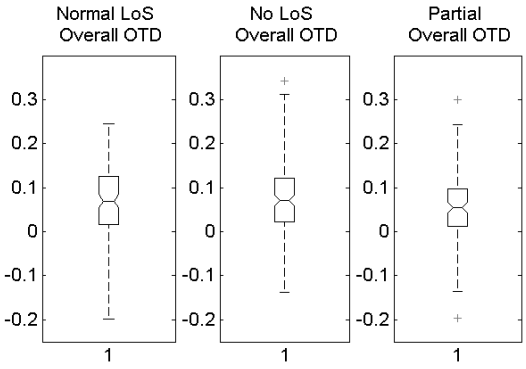


Figure 10. Onset time difference boxplots for each scenario for the second piece (y-axis is in seconds)

4.3 Use of Breath for Cueing

In the recordings of the first piece, it was notable that the violinist took highly regular and audible breaths before each note following a pause, which the musicians identified as important cues. To investigate the use of breath, an extra set of annotations was created, marking the start and end times of each breath, picked by investigating the spectrograms of the recordings. However, as the breaths do not have clear onsets or offsets, this data may be noisy and usable only at a fairly coarse level. An example annotated breath sound is shown in Figure 12. This is a task that could possibly be automated, for example using an algorithm like the one presented by Ruinskiy and Lavner [14].

The first feature of interest was the set of breath start times as a ratio of pause length, or how far into the pauses did the breaths tend to start. Another problem in this case

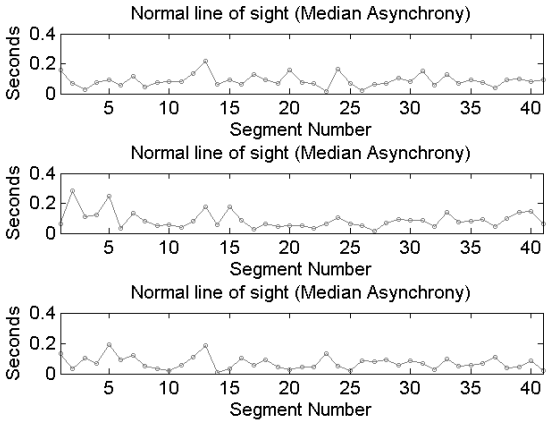


Figure 11. Asynchrony against segment number for the second piece

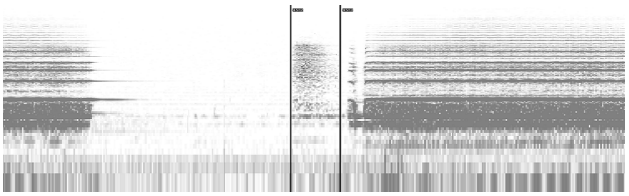


Figure 12. Example of annotated breath sound on spectrogram

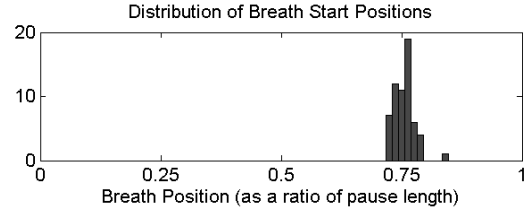


Figure 13. Histogram of breath position.

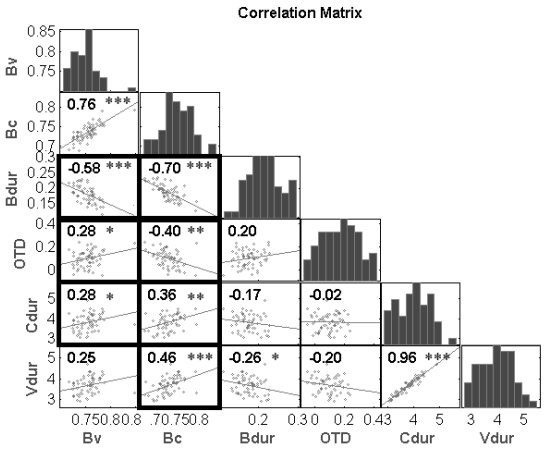


Figure 14. Correlation table of breath and pause variables (stars indicate significance levels: $^{*}0.05$,** 0.01,*** 0.001)

was that finding the start times of the pause segments is difficult as string instruments do not always have distinct note offsets. Annotation of these offsets was based on finding the point where the higher harmonics start to decay, as the fundamental often had a much longer decay, and the musicians often left one string resonating through the pause itself. Pause start times were then taken to be the average offsets of the cellist and violinist's previous notes. Breath position with respect to the violinist's pause offset (i.e. note onset) was then expressed as a value between 0 and 1 representing how far along a pause a breath occurs, as shown in Equation 1:

$$B_{violin,i} = \frac{2b_{i,0} - c_{i,0} - v_{i,0}}{2v_{i,1} - c_{i,0} - v_{i,0}} \quad (1)$$

where i is the pause index, $b_{i,0}$ is the breath onset time, $c_{i,0}$ is the cello's pause onset time, and $v_{i,0}$ and $v_{i,1}$ are the violin's pause onset and offset times, respectively. We correspondingly define B_{cello} as the breath position with respect to the cello's pause offset:

$$B_{cello,i} = \frac{2b_{i,0} - c_{i,0} - v_{i,0}}{2c_{i,1} - c_{i,0} - v_{i,0}}. \quad (2)$$

Figure 13 shows the histogram of breath start positions for all pauses from all recordings. The violinist mostly used breath gestures at the 0.76 point, which closely corresponds to the last half note in the 2-bar pause. To better understand the effect of the breath cues, regression and correlation analysis was performed. This is shown in Figure 14, where Bv and Bc represent B_{violin} and B_{cello}, and Bdur is the duration of the breath as a ratio of the violinist's pause length; OTD is the difference between the violin and the cello onset times, and Cdur and Vdur are the durations of the violinist's and cellist's pauses (a value greater than zero means that the violinist played before the cellist).

From this analysis it is notable that the breath position, despite its small range of variation, had a significant effect on the onset time difference. Later breath positions with respect to the violin's pause had a slight tendency to correspond with positive differences (meaning that the violinist started first), possibly by indicating a later start time to the cellist and making her start later. The correlation between the violinist's breath position with respect to the cello pause time and the onset time difference was much more significant, and the two variables are inversely related, higher values (nearer 0.75) essentially lowering the gap between the musicians' onsets. We also see a very significant inverse correlation between the breath's start position and the length of the breath sound. We also see a positive correlation between the length of the pause (from both musician's perspectives) and the position of the breath along the pause, meaning that in longer pauses the breath started later. The correlations of interest are emphasized in Figure 14. These characteristics identify breath as a type of cue that could be used in a networked scenario to predict a performer's intent to begin a note, serving as the basis for synthesis of virtual cues that can be sent ahead of time to bypass latency, in a manner similar to the rhythm prediction in Sarkar's TablaNet project [9].

5. RESULTS

The results of this study suggest that line of sight is important in achieving good synchronization in a string duo, especially when the music being played contains pauses during which the musicians cannot easily track time. As the partial line of sight scenario did not cause a significant decrease in synchrony, it appears that very simple body motion was sufficient for effective gestural cueing. In scenarios with restricted line of sight, performers can rely on non-visual and extra-musical cues such as breath for synchronization. In this study, the leading musician issued breath cues that were synchronized to their own perception of musical time, and served as advance warnings of note onset intent. The following musician then used this cue to estimate the beginning of the next note. Small variations in breath onset within pauses were correlated with variations in note onset time delay, suggesting that musicians pay close attention to these cues, and that mis-communication of timing by breathing too soon or too late can have direct consequences on synchronization.

When the music had no pauses and contained counterpoint and rhythm, the musicians did not exhibit worse synchronization in the absence of visual contact, suggesting that auditory cues embedded in the music itself were sufficient for synchronization.

6. CONCLUSIONS

The findings of this study indicate a need for further research into the fine dynamics of cues that are transmitted both visually and sonically. Auditory features of interest may be variations in dynamics or pitch in relation to critical synchronization points. Due to the improvement seen in the partial line of sight scenario, we propose that visual cues are likely more dependent on general motion than on eye contact, facial expression, or other such fine details.

Further work should focus on obtaining a larger dataset for study, although the main difficulty is obtaining multitrack, acoustically isolated recordings done in controlled conditions. We intend to continue the study of ensemble synchronization by including visual and motion tracking data in our analysis, in order to discover the most important types of visual gestures and their relationship with the music being performed.

7. ACKNOWLEDGEMENTS

The authors thank Laurel Pardue and Dr. Kat Agres for participating in the experiment. This project was funded in part by the Engineering and Physical Sciences Research Council (EPSRC).

8. REFERENCES

[1] M. McCaleb: "Communication or Interaction? Applied environmental knowledge in ensemble performance," *Proceedings of the CMPCP Performance Studies Network International Conference*, 2011.

[2] R. I. Godoy, M. Leman: *Musical Gestures Sound, Movement and Meaning*, Routledge, 2010.

[3] A. Lim: "Robot Musical Accompaniment: Real-time Synchronization using Visual Cue Recognition," *Proceedings of the IEEE/RSJ International Conference on Intelligent RObots and Systems (IROS)*, 2010.

[4] A. Williamon: "Coordinating Duo Piano Performance," *Proceedings of the Sixth International Conference on Music Perception and Cognition*, 2000.

[5] B. W. Vines, M. M. Wanderley, C. L. Krumhansl, R L. Nuzzo, D. J. Levitin: "Performance Gestures of Musicians: What structural and emotional information do they convey?" In A. Camurri, G. Volpe (eds.): *Gesture-Based Communication in Human-Computer Interaction*, LNCS 2915, Springer, 2004.

[6] C. Chafe, M. Gurevich "Network Time Delay and Ensemble Accuracy: Effects of Latency, Asymmetry," *Proceedings of the 117th AES Convention*, San Francisco, 2010.

[7] E. Chew, A. Sawchuk, C. Tonoue, R. Zimmerman "Segmental Tempo Analysis of Performances in User-Centered Experiments in the Distributed Immersive Performance Project," *Proceedings of the Sound and Music Computing Conference*, 2005.

[8] N. Schuett "The Effects of Latency on Ensemble Performance," Undegraduate Honors Thesis, Stanford University, 2009.

[9] A.A. Sawchuk, E. Chew, R. Zimmermann, C. Papadopoulos and C. Kyriakakis "From Remote Media Immersion to Distributed Immersive Performance," *Proceedings of the ACM SIGMM 2003 Workshop on Experiential Telepresence*, 2003.

[10] C. Chafe "Tapping into the Internet as a Musical/Acoustical Medium," *Contemporary Music Review*, 2009.

[11] M. Sarkar "TablaNet: a real-time online musical collaboration system for Indian percussion," S.M. Thesis, MIT, 2007.

[12] R. Oda, A. Finkelstein and R. Fiebrink "Towards Note-Level Prediction for Networked Music Performance," *Proceedings of the 13th International Conference on New Interfaces for Musical Expression*, 2013.

[13] C. Cannam, C. Landone, M. Sandler "Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files," *Proceedings of the ACM Multimedia 2010 International Conference*, 2010.

[14] D. Ruinskiy, Y. Lavner "An Effective Algorithm for Automatic Detection and Exact Demarcation of Breath Sounds in Speech and Song Signals," *IEEE Transactions On Audio, Speech, and Language Processing, Vol. 15*, 2007.

GROOVE KERNELS AS RHYTHMIC-ACOUSTIC MOTIF DESCRIPTORS

Andy M. Sarroff
Dartmouth College
Department of Computer Science
sarroff@cs.dartmouth.edu

Michael Casey
Dartmouth College
Departments of Computer Science and Music
michael.a.casey@dartmouth.edu

ABSTRACT

The "groove" of a song correlates with enjoyment and bodily movement. Recent work has shown that humans often agree whether a song does or does not have groove and how much groove a song has. It is therefore useful to develop algorithms that characterize the quality of groove across songs. We evaluate three unsupervised tempo-invariant models for measuring pairwise musical groove similarity: A temporal model, a timbre-temporal model, and a pitch-timbre-temporal model. The temporal model uses a rhythm similarity metric proposed by Holzapfel and Stylianou, while the timbre-inclusive models are built on shift invariant probabilistic latent component analysis. We evaluate the models using a dataset of over 8000 real-world musical recordings spanning approximately 10 genres, several decades, multiple meters, a large range of tempos, and Western and non-Western localities. A blind perceptual study is conducted: given a random music query, humans rate the groove similarity of the top three retrievals chosen by each of the models, as well as three random retrievals.

1. INTRODUCTION

The propensity to move to music in a particular way is widespread and fundamental to our experience of music listening and enjoyment. Anyone who has spontaneously bopped their head, clapped their hands, jumped the pogo, swayed their cigarette lighter in the air, or tapped their fingers or toes to music has shared this common experience of near-involuntary musical response. Yet this aspect of music has been little studied in music information retrieval.

The phenomenon has been variously described as flow [3], sensorimotor synchronization [9], feel [16], and groove [8, 11, 12, 16]. Although related, the concept of groove is different from beat, which is the property of a predictable underlying periodic pulse [17]. The degree of groove is correlated with the degree to which the music induces the desire to move rather than the location and frequency of periodic entrainment.

We propose a new algorithm that extracts *groove kernels*—underlying audio patterns that correlate with the propensity to move. We use these features to measure groove similarity between pieces of music. We define groove similarity as that aspect of the sound pattern that induces, within a subject, the desire to move *in the same way*.

We conducted a groove similarity experiment using human subjects. The experiment evaluated three automatic groove extraction and similarity algorithms: an extant tempo-invariant rhythm similarity measure [6], and two versions of a proposed tempo-and-shift invariant groove kernel extraction system. The proposed system is inspired by the work of [18,20] which we extend with a full groove-oriented system architecture.

Our evaluation with human subjects used a diverse dataset of real-world audio files. Results show that our system retrieves music that corresponds more closely to human judgements about groove similarity than random baselines. In summary, the primary contributions of this work are:

- a new *groove kernel* feature based on shift and time-scale invariant Probabilistic Latent Component Analysis,
- a new dataset consisting of over 8,000 musical audio tracks in 10 contrasting genres,
- evaluation of three extraction and retrieval systems and two random baselines against human groove similarity judgments.

To the best of our knowledge, this is the first work to detail an unsupervised system architecture for groove features and experimentally evaluate it on human subjects. The following section provides background and motivation. Section 3 describes how groove kernels are extracted from audio files. Experimental evaluation is presented in Section 4, followed by concluding remarks in Section 5.

2. BACKGROUND

2.1 Groove

Underlying the operational character of groove is the psychological sensorimotor synchronization of auditory stimuli and human movement. In their study Janata et al. [9] showed that subjects strongly agreed that groove was described by the extent to which music induces movement, a positive affect due to such proclivity, and a feeling of being part of the music. From a sensorimotor perspective [12] defined groove as "wanting to move some part

of the body in relation to some aspect of the sound pattern". Other studies consider timing deviations [1], or temporal discrepancies [11], with respect to precise metronomic timing as the source of the groove, relating these to expressiveness and proclivity for motion. Pressing describes groove, or feel, as a "firmly structured temporal matrix" [16]. It is a temporal foundation and an emergent phenomenon formed out of concurrent recurring pulses (a stable sense of tempo), perception of a cycle of time that lasts for 2 or more pulses, and is effective in engaging synchronization of bodily movement.

Following [16] we take the position that groove induces characteristic responses in subjects and these responses stem from specific repeated acoustic patterns. Substantially different patterns induce different tendencies of motion, therefore the feel or the groove is different. Music that grooves is characterized by strong repetition. Therefore also following [16], we expect to observe a foundational "temporal matrix" that expresses the acoustic pattern corresponding to a particular groove at the time scale of roughly two bars. We hypothesize that such foundational patterns are invariant to shifts in time (i.e. within a song) and shifts of tempo (i.e. between songs).

2.2 Beat, Meter, Rhythm

To express invariance to shifts in tempo the description of groove must be normalized to the concept of beat. Alignment of a temporal matrix to the beat is not enough for comparisons between musical excerpts. There must also be a way to normalize for the phase of a temporal pattern with respect to beat hierarchy, or meter. There are two approaches to this problem: bar extraction and circular shifting of the temporal matrix. If we wish to represent groove as a multi-bar pattern, then we must rely on circular shifting.

Holzapfel et al. [6] addresses tempo invariant representations of rhythm at multiple time scales, therefore characterizing multi-scale rhythm. This work is unique in that it provides a scale-invariant song-level rhythm descriptor for music. Holzapfel et al. show that music with similar albeit complex rhythmic structure may be successfully categorized, even when the tempos are rather different. Therefore we see their algorithm as a candidate representation for groove similarity.

2.3 Rhythm Retrieval and Classification

While groove retrieval has not been explicitly treated by the music information retrieval community, rhythm classification and retrieval has been studied in recent years. Rhythm similarity metrics typically extract a rhythm descriptor that exhibits tempo-invariant properties. For instance [5] measures pairwise rhythm similarity using beat spectra based upon beat-synchronous low level features. Pattern segmenting is used in conjunction with dynamic time warping of acoustic features for pairwise rhythm similarity in [13].

The annotation of a large-scale rhythm-based dataset is expensive. Several authors have leveraged the music recordings available at the Ballroom Dancer's website [1] which have tempo and genre annotations. With this dataset authors have presented the results of genre classification tasks using rhythm descriptors such as amplitude envelopes of bar/beat synchronous features [4]; log-scale autocorrelation of onset strength signals [10]; fluctuation patterns [15]; and spectral rhythm patterns [14]. The success of many of these approaches is augmented when tempo metadata from the dataset is included. Hence the experimental results reported often reflect a semi-supervised approach.

2.4 Shift-Invariant Representation

To extract the most salient repeated aspects of the music we use shift-invariant probabilistic latent component analysis (SI-PLCA) [18]. A convolutional variant of non-negative matrix factorization (NMF), SI-PLCA places NMF in an explicitly Bayesian framework and extracts time-frequency components that are stable to shifts in time or frequency.

Our work focuses on time-shift invariant PLCA. Given a nonnegative matrix V, time-shift invariant PLCA factorizes V such that $V \approx \sum_k z_k W_k * \mathbf{h}_k$, k is an index to the factor components, Z is a diagonal matrix containing mixing coefficients, and $*$ is the convolutional operator. In our models we extract one component. Hence $z = 1$, W is a two-dimensional matrix, and $\mathbf{h}$ is a vector. We refer to W as a kernel and $\mathbf{h}$ as an activation function locating the kernel at multiple positions in a track.

Weiss and Bello [20] used SI-PLCA to evaluate song structure segmentation in a Beatles data set. They employed chroma features to extract multiple phrase-level blocks within songs whereas we use constant-Q spectral and cepstral features to extract single rhythmic kernels at the bar level using different sparseness constraints. Finally, we assessed kernels in a groove similarity task with a large and diverse dataset using human evaluators.

3. SYSTEM ARCHITECTURE

The groove kernel is built in four stages. In the first stage, bar and beat detection is performed. In the second stage, beat-synchronous features are extracted and bar/beat activation templates are generated. The third module estimates meter. The fourth stage extracts a shift-invariant groove kernel. The overall architecture is depicted in Figure 1. The details of each stage are described below.

3.1 Beat and Bar Tracking

Given a discrete time audio signal, we perform bar and beat tracking using the Queen Mary bar and beat tracker [2] reported in [2, 19]. The meter of the audio is a required input parameter for the bar tracker. Since we do not know the meter of a given musical audio file, we run the bar tracker twice: once assuming 3/4 meter and again assuming 4/4 meter. We note that changing the value of the input parameter to the beat/bar tracker does not affect the estimated

[1] http://www.ballroomdancers.com/

[2] Available as a Vamp plugin at http://isophonics.net/QMVampPlugins.

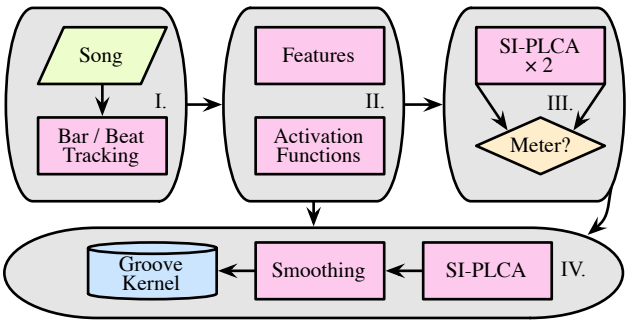


Figure 1. Overview of system architecture.

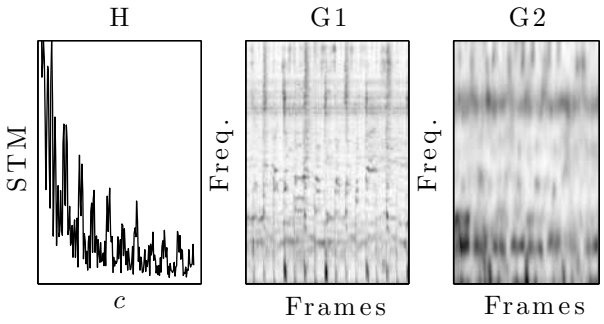


Figure 2. Left: *H* descriptor. The x axis plots the scale coefficient, with $1 \leq c \leq 100$, averaged across all frames. Middle: *G1* descriptor. Right: *G2* descriptor.

locations of the beats, only the indices that represent estimated bar onsets. The Queen Mary beat tracker has a reported accuracy of 73.6% when metrical level is not taken into account. The downbeat detector has a reported accuracy of 52.6%. Beat and downbeat tracking is an open problem and these results are comparable to the state of the art.

3.2 Beat Synchronous Features and Activation Templates

We extract frequency domain beat synchronous features. In this work, we use two feature types—the Constant Q Fourier Transform (CQFT) and the Low-Quefrency Constant Q Fourier Transform (LCQFT). The CQFT is computed by applying a log frequency-spaced filterbank to the Short Time Fourier Transform (STFT) of the audio signal. The LCQFT is computed by transforming the CQFT of the audio signal to the cepstral domain, applying a low-pass lifter, and inverting the signal back to the log frequency domain, analogous to MFCCs.

There are several parameters to choose when extracting the low level features. In this work, our audio has a sample rate of 22050 Hz; we use 2048-point FFTs over hamming-windowed frames of audio. The hop size is dynamically determined based upon estimated beat locations. The duration of each estimated beat is allocated 16 feature frames. The CQFT is computed using 24 bands per octave beginning at the approximate frequency of the musical note C2, yielding 178 CQFT coefficients per frame. We use 15 lower cepstral coefficients of the CQFT to compute the LCQFT (the first cepstral coefficient is ignored). A linear pre-emphasis is placed over the frequency channels to give higher weight to higher frequencies. This weighting helps SI-PLCA avoid placing too much probability on the lower frequencies.

The CQFT preserves pitch information while reducing spectral resolution at higher frequencies. The liftering stage of the LCQFT effectively removes much of the pitch information from the signal while retaining the timbre information. Hence the CQFT-based model is a pitch-timbre-temporal model, while the LCQFT-based model is a timbre-temporal model. We refer to the respective models as *G1* and *G2* in the rest of this paper.

We generate several activation templates based upon the estimated locations of beats and bars. The templates are used as priors over the activation function $\mathbf{h}$ for meter estimation and groove kernel extraction. The amplitudes of the spikes sum to one. A duple meter and a triple meter activation temple are generated without regard to the bar locations. These have a spike every second and third beat, respectively. The templates are used for meter estimation. A duple and triple meter activation template are also generated for groove kernel extraction. The duple-meter activation template has a spike every fourth beat and the triple meter activation template has a spike every third beat. These templates are organized such that the first spike is centered on the estimated location of the first bar line.

3.3 Meter Estimation

We do not know *a priori* what the meter of a given song is. To extract an effective groove representation in the following stage, we set the size of the kernel based upon a meter assumption. In this stage, we make a decision about the meter assumption using the log probabilities of two SI-PLCA models.

The meter estimation activation templates are given as prior probabilities to two independent SI-PLCA models: one with a triple, and the other with a duple meter assumption. The triple-meter model has a kernel window size of 96 frames (2 bars in 3/4). The duple-meter model has a kernel window size of 128 frames (2 bars in 4/4). The triple and duple models are run until convergence, with updates to the activation functions allowed. There are no sparsity constraints imposed on the model optimizations. However, since the initial activation functions are sparse, the final activation functions are also sparse. The meter of the song is chosen according to whichever model has the highest log probability after convergence.

3.4 Groove Kernel

Once the meter has been chosen, we extract a groove kernel using a final stage of SI-PLCA. We provide a new initial activation template $\mathbf{h}$ to the model in which there is an impulse every 4 or 3 beats, based upon the assumed meter. Note that a duple meter model now has activations every

4 beats, instead of 2. We also use the bar estimations produced in the second stage to center the activations at the onsets of 4/4 or 3/4 bars. The window size is set to two bars.

Weiss and Bello [20] have suggested that the optimal window size and meter may be learned by setting a sloping prior over an initial $\mathbf{h}$. We tried this approach using varying initializations of $\mathbf{h}$, slope degrees, initial window sizes, and sparsity parameters. In informal listening tests we found that our method provided better qualitative results when the bar and beat tracker was accurate.

Once a groove kernel has been extracted we smooth it along the time axis using a gaussian window. In this paper *G1* has no smoothing and *G2* is smoothed with a gaussian window having a standard deviation of 1 frame. Since our features have 16 frames per beat this window places approximately 95% of the window over 4 frames, or 1/16 note.

We do not know whether the phase of the groove kernel is aligned with respect to a latent two-bar groove structure of the music. Therefore for every groove kernel we enter a zero-phase and a circularly-shifted 1/2-phase version into our database.

4. EXPERIMENTS

4.1 Dataset

We built a dataset consisting of thousands of songs to evaluate the algorithms presented in this paper. All data is publicly available and we will provide the aggregate dataset and all associated metadata upon request. The data collection steps are summarized below.

We used the Echo Nest developer's API [3] to construct a list of 10,000 song titles across 10 genres and 10,000 unique artists. We began by querying the top styles in Echo Nest's database. An Echo Nest "style" is a search term associated with artists. Styles are essentially genres; the top ranked styes are those that Echo Nest believes yield the strongest search results. We will refer to Echo Nest styles as genres hereafter. We handpicked 10 genres from the highest ranked members of the list that we associated with having groove and variety. Table 1 shows the genres we selected, along with their Echo Nest rank.

For each genre we queried 1000 unique artists that were also cross-indexed with the 7digital [4] database, ranked by genre relevance. For each artist we queried 1 unique song that was in the 7digital database, ranked by Echo Nest's highest "danceability" estimation.

7digital is a commercial music distribution service that maintains .mp3 previews for most of the songs in their catalogue. We downloaded all previews in our list from the 7digital website. While sampling the dataset we discovered anomalous files. We filtered these out, resulting in 8249 unique song/artist clips each between 30 and 60 seconds long. We believe that the dataset dually exhibits a wide representation of groove and low redundancy.

[3] http://developer.EchoNest.com/

[4] http://us.7digital.com/

Rank and Genre										
1	rock	2	electronic	3	hip hop	6	jazz	14	pop	
17	reggae	19	funk	88	latin jazz	168	world	179	country	

Table 1. Echo Nest ranks and genres used in this work.

4.2 Models

We investigated three models, designated *H*, *G1*, and *G2*. *H* yields a temporal rhythm descriptor. *G1* yields a pitch-timbre-temporal groove kernel. *G2* yields a timbre-temporal groove kernel.

The *H* model is the scale invariant rhythm descriptor presented by Holzapfel and Stylianou in [7]. *H* computes multiple Direct Scale Transforms (DSTs) on the autocorrelation function of an Onset Strength Signal. We follow the procedure outlined in [7]. The DST is computed over a range of scale coefficients. The value of the maximum coefficient is denoted as C. Holzapfel and Stylianou show that the optimal value of C is related to the source material, but a value of $C > 80$ achieves nearly constant accuracy in their rhythm similarity tasks. The *H* model sets $C = 100$. The final descriptor is the average of the scale transform magnitudes across frames.

The other two models are *G1* and *G2*. Their architecture and parameterization are described in Section 3. Note that the key differences between *G1* and *G2* are that *G1* uses CQFT features. *G2* is built with LCQFT features and has smoothing over the groove kernel.

Figure 2 graphically depicts three extractions from the same song clip using *H* (left), *G1* (middle), and *G2* (right). Observe that the *H* descriptor is a vector of Scale Transform Magnitude (STM) against a range of scaling coefficients (denoted c). *G1* and *G2* exhibit different images even though they are extracted on the same audio clip. *G1* has finer-grained detail in the temporal domain and a sustained tone with harmonics in the upper third of the image. The rhythmic structure is apparent in *G2*, but the tonal and temporal detail has been smoothed.

4.3 Methods

A subset of 100 musical queries—10 from each genre—were randomly selected from the dataset. For each query and each model the top-3 nearest neighbors were selected (excluding the same song), as measured by cosine similarity. Retrievals for the *G2* model were additionally restricted to a have an estimated tempo difference of 8 BPM to limit the range of tempo variation.

There were two sets of random retrievals. The *R1* retrieval set has 3 songs chosen at random for each query. Each retrieval in the *R2* set has an estimated tempo difference from its associated query of less than or equal to 8 BPM. Tempos were estimated by computing the median beat onset differences derived from the bar and beat tracker. Random retrieval sets were not restricted by genre.

There were two types of participants: solicited and anony-

mous. Solicited participants were paid if they completed the entire experiment. Both types of participants were presented the same web-based experiment interface. Participants were randomly assigned one of ten genre-based test subsets. A test subset consists of 10 same-genre queries and 5 retrieval sets. We collected 2,436 ratings from solicited participants and 236 ratings from anonymous participants. There were 56 unique human evaluators that participated in our experiment.

The experiment required that participants utilize a quiet listening room or headphones. Participants were presented with the following definition of groove [9]: "The groove is that aspect of the music that induces a pleasant sense of wanting to move along with the music."

We are unaware of any studies in the literature on the perception of groove *similarity*. We therefore asked participants to consider the similarity of groove based upon the given definition. The experimental interface further stated, "Please try to avoid judging groove similarity based upon song genre. For instance, you may find that two songs are from different genres, but you would move your body in a similar way to them. You should rate these songs as having high groove similarity."

Each participant was presented query-retrieval pairs from their test subset in random order. The audio clips were approximately 5 seconds in duration, corresponding to the expected duration of a two bar motif. They were asked to rate groove similarity on a coarse scale using radio buttons having the labels "Not Similar", "Somewhat Similar", and "Very Similar". These ratings were later assigned numerical values from the set $\{0, 1, 2\}$. We denote these as "coarse" ratings. Participants were also asked to rate the groove similarity of each pair on a fine scale with a slider. The slider had a range of $[0, 100]$, but the slider's numerical value was not exposed to the user. We denote the ratings as "Fine".

A participant was required to listen to each pair of audio clips at least once and assign ratings before moving to the next comparison. Multiple listens were permitted. We note that the experimental design was modeled after the MIREX Audio Music Similarity and Retrieval [5] evaluation procedure. One minor difference is that the Mirex Audio Similarity task asks its evaluators to rate the top-5 ranked songs per query and model. Due to limited human resources, we restricted the retrieval space to top-3.

4.4 Results

Figure 3 shows the mean coarse and fine ratings per retrieval set. The red cross hairs show standard error. Table 2 shows the results of pairwise t-tests between appropriate baselines and models. Note that *G2* may only be directly compared with *R2*; these were the retrieval instances where the search space was restricted by tempo difference.

The first thing that we notice is that participants were pessimistic about the groove similarity of query-retrieval pairings. The mean coarse value across all ratings was 0.780; the mean fine rating was 40.909. Users were more likely to rate a pair of songs as being not similar or somewhat similar than very similar. The songs in the dataset spanned a range of 10 base genres. Several participants expressed that they had difficulty cognitively separating genre and preference from groove. Indeed, Janata et al. have shown that enjoyment is correlated with groove [9]. We are not aware of a study that evaluates correlation between genre and groove similarity.

[5] http://www.music-ir.org/mirex/wiki/Audio_Music_Similarity_and_Retrieval

	R1-H	*R1-G1*	*R2-G2*
Coarse	$\mathbf{4.52 \times 10^{-5}}$	**0.0038**	0.1160
Fine	$\mathbf{1.66 \times 10^{-4}}$	**0.0094**	**0.0142**

Table 2. Pairwise t-test p-values. Boldface indicates a p-value less than 0.05.

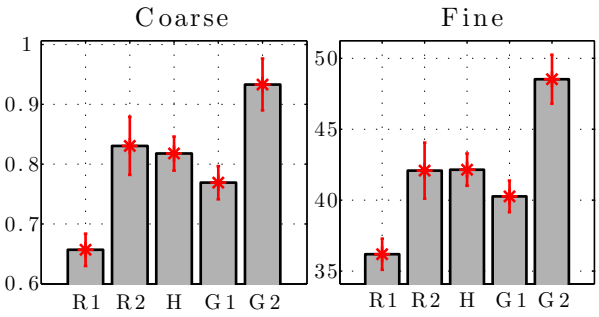


Figure 3. Mean coarse and fine ratings by algorithm. The red cross hairs show standard error about the mean.

Secondly we observe that all models retrieve groove-similar songs better than random selection when the retrieval space is unrestricted by tempo. We have learned from Janata et al. that humans are able to reliably detect the presence of groove. Our results support the hypothesis that humans may also reliably detect groove similarity.

We find that *G1* and *H* perform competitively. When adjusting for multiple comparisons using the Tukey-Kramer method each performs significantly better than *R1* (at 95% confidence), but they share similar statistical distributions with each other for coarse and fine ratings.

We notice that groove similarity ratings jump upward for random retrieval when the space is limited to an 8 BPM tempo difference from the query. Humans are more likely to rate two arbitrary songs to have similar groove if they are close in tempo.

The only model that was evaluated with a restricted tempo space was *G2*. As can be seen in Figure 3 and Table 2, this model performed significantly better than the tempo-restricted random set on fine evaluations. We do not know whether the increased performance of *G2* is due to a (pitch-free) low-level feature or the gaussian smoothing of the groove kernel. Our intuition leads us to believe that smoothing had a significant impact. The kernels are fairly high-dimensional. By smoothing them, neighbors that were once distant due to fine differences in temporal structure become less distant (cf. Figure 2).

5. CONCLUSIONS

Groove is associated with the often pleasurable induction of bodily movement to music. There are an increasing number of rhythm similarity and classification algorithms in the literature, yet groove encompasses a higher-level construct involving sensorimotor interaction stemming from repeated acoustic patterns. We presented a new groove kernel feature based on shift and tempo invariance. We asked humans to evaluate the groove kernel and another rhythm similarity model in a groove similarity retrieval task using a diverse collection of real-world music recordings spanning 10 base genres.

The *H* and *G1* models give groove similarity rankings that are significantly better than random retrieval. The *G2* model performs significantly better than random retrieval when the retrieval space is limited to an 8 BPM absolute difference from the query.

We note that all three models—the temporal *H* model and our proposed timbre-inclusive SI-PLCA based models—are constructed in an unsupervised manner. Building human-annotated collections of music is expensive. Hence there is higher value associated with models that do not rely on human annotation.

Our models rely on beat-synchronous features derived from the automatically estimated bar and beat estimations. As noted in Section 3 beat and downbeat estimation is not a solved problem. We may assume that there is error in the estimated beat and bar locations. Unfortunately, this error is necessarily propagated forward through every stage of the groove kernel models. We expect that our proposed models will perform better as beat and downbeat detection improves.

The groove kernel activation templates were restricted to 3/4 and 4/4 meters. While the dataset included a large selection of world music, it is possible that the learned kernels did not fit a significant portion of the dataset. We expect that the algorithm could be improved with enhanced meter detection.

The experimental design did not allow for a direct comparison between the *H* and *G2* or *G1* and *G2* methods. We therefore cannot draw conclusions regarding the impact of the tempo-restricted space on these methods. We also cannot state conclusively the contribution that smoothing has on the *G2* model since this effect was not studied.

Our human subject study had a limited number of human participants with respect to the number of song queries. There were 100 queries each associated with 3 models and two random baselines. An improved study would include the effects of pairwise perceived genre similarity, song preference, and other potential biases. Further investigation is needed into the relationship between *how much* groove is perceived and groove similarity. Future work will include a larger scale human evaluation with the intent to address these important issues.

To the best of our knowledge, this paper provides the first human-based groove similarity retrieval task. Experimental results suggest that the groove kernel presents a promising direction for exploration of groove metrics.

6. ACKNOWLEDGMENTS

This project has been supported by a Google Faculty Research Award and by a Neukom Institute for Computational Science Graduate Fellowship.

7. REFERENCES

[1] J.A. Bilmes. Timing is of the essence: perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive rhythm. Master's thesis, Massachusetts Institute of Technology, 1993.

[2] M.E.P. Davies and M.D. Plumbley. A spectral difference approach to downbeat extraction in musical audio. In *Proc. EUSIPCO*, 2006.

[3] O. de Manzano, T. Theorell, L. Harmat, and F. Ullen. Psychophysiology of flow during piano playing. *Emotion*, 10:301–311, 2010.

[4] S. Dixon, F. Gouyon, and G. Widmer. Towards characterisation of music via rhythmic patterns. In *Proc. ISMIR*, volume 5, 2004.

[5] J. Foote, M. Cooper, and U. Nam. Audio retrieval by rhythmic similarity. In *Proc ISMIR*, volume 3, pages 265–266, 2002.

[6] A. Holzapfel and Y. Stylianou. A scale transform based method for rhythmic similarity of music. In *Proc. ICASSP*, pages 317–320. IEEE, 2009.

[7] A. Holzapfel and Y. Stylianou. Scale transform in rhythmic similarity of music. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(1):176–185, 2011.

[8] V. Iyer. Embodied mind, situated cognition, and expressive microtiming in african-american music. *Music Perception*, 19:387– 414, 2002.

[9] P. Janata, S.T. Tomic, and J.M. Haberman. Sensorimotor coupling in music and the psychology of the groove. *Journal of Experimental Psychology: General*, 141(1):54–75, 2012.

[10] J.H. Jensen, M.G. Christensen, and S.H. Jensen. A tempo-insensitive representation of rhythmic patterns. In *Proc. EUSIPCO*, 2009.

[11] C. Keil and S. Feld. *Music grooves*. University of Chicago Press, Chicago, IL, 1994.

[12] G. Madison. Experiencing groove induced by music: Consistency and phenomenology. *Music Perception*, 24:201–208, 2006.

[13] J. Paulus and A. Klapuri. Measuring the similarity of rhythmic patterns. In *Proc. ISMIR*, volume 2, 2002.

[14] G. Peeters. Spectral and temporal periodicity representations of rhythm for the automatic classification of music audio signal. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(5):1242–1252, 2011.

[15] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Proc. ISMIR*, volume 9, 2009.

[16] J. Pressing. Black atlantic rhythm: Its computational and transcultural foundations. *Music Perception*, 19:285–310, 2002.

[17] E. Scheirer. Tempo and beat analysis of acoustic musical signals. *Journal of the Acoustical Society of America*, pages 588–601, 1998.

[18] P. Smaragdis, B. Raj, and M. Shashanka. Sparse and shift-invariant feature extraction from non-negative data. In *Proc. ICASSP*, pages 2069–2072, 2008.

[19] A.M Stark, M.E.P Davies, and M.D. Plumbley. Real-time beat-synchronous analysis of musical audio. In *Proc. DAFx*, 2009.

[20] R.J. Weiss and J.P. Bello. Unsupervised discovery of temporal structure in music. *Selected Topics in Signal Processing, IEEE Journal of*, 5(6):1240–1251, oct. 2011.

INSTRUMENT IDENTIFICATION INFORMED MULTI-TRACK MIXING

Jeffrey Scott and Youngmoo E. Kim
Music and Entertainment Technology Laboratory (MET-lab)
Electrical and Computer Engineering, Drexel University
{jjscott, ykim}@drexel.edu

ABSTRACT

Although digital music production technology has become more accessible over the years, the tools are complex and often difficult to navigate, resulting in a large learning curve for new users. This paper approaches the task of automated multi-track mixing from the perspective of applying common practices based on the instrument types present in a mixture. We apply basic principles to each track automatically, varying the parameters of gain, stereo panning, and coarse equalization. Assuming all instruments are known, a small listening evaluation is completed on the mixed tracks to validate the assumptions of the mixing model. This work represents an exploratory analysis into the efficacy of a hierarchical approach to multi-track mixing using instrument class as a guide to processing techniques.

1. INTRODUCTION

The pervasive use of digital tools for creating, recording, producing and editing audio has led to a desire for increased automation and efficiency of these tools. Although there is a wide variety of digital audio workstations (DAW) and plug-in suites available, the level of expertise required to operate them proficiently necessarily inhibits many newcomers from obtaining reasonable results even with a significant amount of effort. This has led to an exploration in the audio signal processing community for methods of automatically analyzing audio and improving the perceived quality. Several significant difficulties arise when attempting this task. The qualitative difference between the preference of individuals, the wide range of timbre, dynamics and instrumentation and the multitude of production techniques available present ample hurdles to overcome.

This paper attempts to exploit some of the commonalities between processing chains for a specific instrument class, namely the drum kit. Figure 1 shows the general framework for the task. Given the identities of individual drum tracks (kick, snare, tom and overhead), we apply some basic guidelines to make the kit sound more balanced using spatial and spectral modifications.

In order to apply the processing techniques, the labels of each drum track must be known. To this end, we provide a simple classification experiment to evaluate the difficulty of classifying these tracks in a real world situation. A listening test is conducted to determine how well the model mixes the individual drum tracks. For the listening test, we use the ground truth instrument labels to evaluate how well the instrument based model mixes the tracks.

The remainder of the paper is organized as follows. Prior work in automatic mixing and instrument identification is presented in Section 2. Details about the dataset are outlined in Section 3 and the processing employed to form a drum mix is in Section 4. Sections 5 and 6 detail the classification experiment and listening evaluation, respectively.

2. BACKGROUND

Interest in automating multi-track production tasks has increased in recent years, focusing on both real-time and offline models based on perceptual information, acoustic features and best practices [1, 2].

Significant work on cross-adaptive methods for multi-track mixing for specific parameters has been explored in [3–7]. The implementation of these models relies on computing time domain and spectral features on the tracks present in a mixture and devising models to control mixing parameters based on the temporal and inter-track relations between the features. Psychoacoustic models of loudness and frequency masking are leveraged to inform the relationships on a perceptual level and more closely approximate the signal characteristics that are processed by humans.

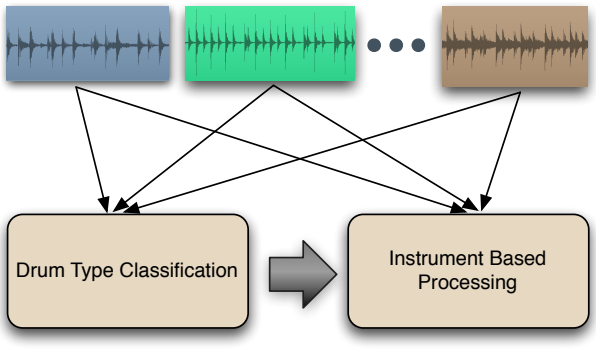


Figure 1: Identification informed mixing.

Related work focuses on estimating production techniques from user examples as well as higher level constructs of musical timbre. Variation in the vocabulary used to describe music as well as individual artist and listener preference makes universal rules difficult to derive. Observing and recording user interaction with mixing tools creates data that can be used to model individual preferences [8–10].

There have also been developments in learning mixing parameters from data. The authors in [11] use regression to estimate fader gains, equalization and compression from mixed audio signals. In a previous paper, we attempted to learn dynamic mixing models directly from data [12].

Instrument identification is a problem that has been popular in Music-IR for many years. State of the art performance on datasets of individual samples of both pitched and un-pitched instruments has reached classification accuracies in the upper 90th percentile [13–15]. However, most of these systems evaluate libraries of single, isolated instrument tones. There has been some work that shows good performance on longer excerpts of pitched instruments [16], but there are not comparable results to individual sample classification.

3. DATASET

The dataset used throughout this paper consists of 135 songs across a variety of genres. The genres include Acoustic, Alternative, Country, Dance, Electronic, Hip-Hop, Indie, Jazz, Rock and Metal. The songs were obtained from three primary sources: Weathervane Music [1], Sound on Sound [2] and a multi-track dataset used for song structure segmentation [17]. Each track is converted to a monaural source at 44.1kHz sampling rate and labeled with the instrument present in the track.

The tracks in every song are labeled with the instrument present by three individuals and the majority label for each track was retained as ground truth. The labelers are students in the music industry program at Drexel University. The filenames for each audio track are used when possible and normalized to a standard label for a single instrument class. Instrument classes are differentiated on a fine level (clean/distorted electric guitar) and may be combined into superclasses (electric guitar) if desired. The electric guitar is a specific example where fine level labels are desired since the distorted and clean versions are treated very differently by engineers and have much different roles in the mix. The dataset is publicly available online [3].

4. INSTRUMENT BASED PROCESSING

In this approach we attempt to codify some common practices and apply them to multi-track drum audio. Several professional and student mixing engineers were interviewed about the process of mixing audio and it was unsurprising to find that all of them specified that their approach is dependent upon the source material (i.e. genre, instrumentation). It is quite difficult to define a set of hard and fast rules for mixing audio yet there do exist some commonalities that many agree upon. We apply some basic techniques to improve the balance and quality of the drums via stereo panning, filtering and level adjustment. The motivation for the processing techniques employed in the following subsections are derived from the engineer interviews as well as authoritative sources on mixing [18,19].

There are several concerns when combining the signals from multiple drum microphones to produce a mixture. Problems with phase coherence between the different microphones can often occur and result in a comb filtering effect applied to the instruments [18]. This is the case with bleed (leakage) between microphones on different instruments as well as multiple microphones on a single instrument (as in the top/bottom heads of a snare drum). In properly recorded material this effect is usually anticipated for and dealt with during signal capture and therefore not considered in this paper.

We consider three proccessing areas: level balancing, stereo panning and equalization. Two basic approaches for level adjustment are serial (faders down) and parallel (faders up) [18,19]. The serial approach invovles adding in layers one at a time and the parallel approach starts with all layers active and adjusts levels accordingly. We opt for the parallel approach where the level of each instrument track is evaluated individually against the rest of the mix. There are also two main approaches to using the ambient (overhead/room) mics. One primarily uses the overheads as the main drum signal and uses the individual instrument mics as reinforcement when needed. The alternate approach is to use the close microphones as the primary signal source and use the overhead microphones to increase the amount of cymbals and add 'air' to the mix. We opt to use the latter approach in this work.

For panning, one may start with a stereo spread of the overhead mics and pan the close microphones according to their position in that signal. Another common approach is to pan the kick and snare dead center since they are the driving force of the rhythm section. This is the option we choose in our model.

The equalization applied is minimal and was obtained from the interviews of engineers. The interviewees expressed reservation about making gereralizations without hearing the source material and knowing what other instruments are in the mixture, yet these are the same issues they expressed with nearly all aspects of mixing, namely that each session is different and must be approached individually. Nevertheless, a filtering scheme was developed to boost frequency ranges that often need boosting and cut frequency ranges that often need attenuating. Ideally, this would be done adaptively through comparing bandwise energy ratios and making adjustments accordingly.

Before processing, each track is analyzed to determine where the instrument is playing on each track. We only want to compare signal characteristics where there is an ac-

[1] http://weathervanemusic.org/
[2] http://www.soundonsound.com/
[3] http://music.ece.drexel.edu/research/AutoMix

tive instrument in a track, not where there is just the noise floor. Figure 2 depicts the computation of the active regions in each track. The first four steps, full-wave rectification, low pass filtering, downsampling and smoothing with a moving average filter produce the temporal envelope of the signal and the threshold determines active regions. After thresholding, any segments less than 150ms long are discarded.

4.1 Stereo Panning

In [5, 20] a dynamic cross-adaptive model is used to actively pan tracks as they come in and out of the instrument mixture based on several constraints related to spectral and spatial balance and masking. Here we attempt to leverage common practices in drum kit panning and apply them to the individual tracks of a drum kit. This results in a static value being applied to the entire track for the duration of the song regardless of the presence or absence of instrument playing at any given time. Panning a drum kit is one aspect of mixing that is fairly consistent between engineers. Qualitatively, the stereo balance of the drum mix is as follows:

1. Kick drum panned center
2. Snare drum panned center
3. Toms panned from left to right
4. Overhead microphones panned left and right

Panning is accomplished by applying the sine-cosine panning law

$$L_{pan} = cos(45^\circ - \theta) \quad (1)$$

$$R_{pan} = sin(45^\circ - \theta). \quad (2)$$

Here $\theta \in [-45^\circ, 45^\circ]$ and represents the angle offset from the center of the stereo field with -45° being panned fully to the left and $+45^\circ$ panned fully right. This method of panning maintains the perceived loudness of the signal as it is varied from left to right. Table 1 shows the parameter values used to pan the tracks.

The kick and snare drums are panned in the center of the stereo field. The toms are spaced linearly from left to right with 25° being the maximum offset from the center position. The overhead tracks are panned alternating left and right at the specified value in Table 1.

4.2 Relative Levels

After panning, the loudness of each track is computed and compared against the loudness of the rest of the tracks to determine any boost or attenuation that is desired for each track. The loudness of each track is calculated by filtering the signal using the inverse of the ISO 226 normal equal-loudness-level contours (at 75 phons) and then computing the RMS energy over a 23ms window [21]. The level of 75 phons was chosen based on preferred listening levels shown in [22]. The loudness of the target track ($\mathbf{x}_{loud}$) is compared to the loudness of the sum of the remaining tracks ($\mathbf{y}_{loud}$) and a loudness ratio is computed,

$$r_{loud} = \frac{1}{T}\sum_{\tau}^{T} \frac{x_{loud}^{(\tau)}}{y_{loud}^{(\tau)}}, \quad (3)$$

where x and y are in dB and T is the total number of short time frames in the current song being analyzed. The loudness ratio is then used to attenuate or boost the level of the track in question. The gain of the track is determined using the following equation

$$g = 10^{(-\frac{1}{\lambda} log(r_{loud}))}. \quad (4)$$

Equation 4 offers control over the amount of level correction that is applied to each instrument through the parameter λ. As λ increases, the amount of level correction is reduced as shown in Figure 3.

Loudness is computed on each channel (L/R) after panning and the average of the loudness ratios is used to determine the gain of the instrument. There are three parameters $\{\alpha, \beta, \lambda\}$ for each instrument type that determine how the loudness ratio affects the gain, g, applied to the track. The α and β parameters define thresholds for the loudness ratio necessary to apply loudness correction. For example, if we require $r_{loud} < \alpha$ or $r_{loud} > \beta$ where $\alpha = 0.8$ and $\beta = 1.2$ before applying gain g, then the track will have no level correction if $r_{loud} \in [0.8, 1.2]$ and will have loudness correction specified in Equation 4 otherwise. The parameters in Table 1 are specified to err on the side of more kick and snare drum than overhead and tom microphones since the kick and snare instruments are generally more prominent in rock music.

Instrument Class	Panning Value (θ)	Gain Values $\{\alpha, \beta, \lambda\}$
Kick Drum	0 (center)	{0.9, 1.2, 2}
Snare Drum	0 (center)	{0.9, 1.2, 2}
Toms	Spaced {-25, 25}	{0.8, 1.3, 4}
Overhead/Room	{-35, 35}	{0.8, 1.3, 4}

Table 1: Mixing parameter values for individual drum tracks.

4.3 Equalization

The desired frequency content for a specific instrument is very genre dependent. For example in an electronic track the kick drum generally contains more low frequency content and may be prominent even into the sub-bass range. In heavy metal, the sound of the beater striking the kick drum is often desirable and the signal may need to be boosted in

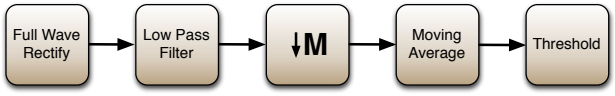


Figure 2: Processing chain to calculate the active areas of an instrument track.

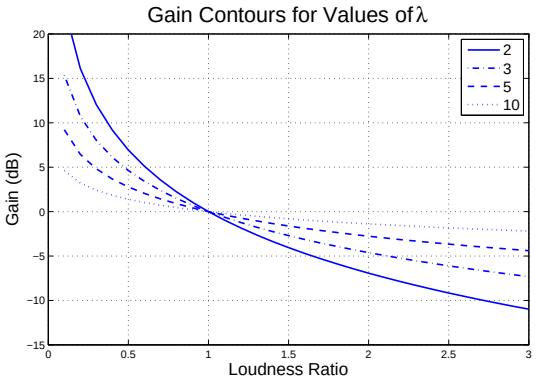


Figure 3: Contours of gain attenuation for various γ.

the high-mid frequency range. For these reasons we chose to apply only subtle equalization based on some common operations. The kick drum has a 2dB boost from 1kHz-6kHz, a 2dB cut from 400Hz-900Hz and a 2dB boost of 100Hz with a quality factor of 4.5. The snare drum has a 3dB high shelving boost starting at 10kHz. These modifications are designed to give the kick drum slightly more punch and the snare drum more brilliance.

5. DRUM TYPE CLASSIFICATION

For an unknown set of tracks, the drums would need to be identified to apply the common practices outlined above. Here we explore a preliminary experiment to classify a track in terms of the drum content it contains. The approach is fairly standard for supervised learning and is meant to serve as a benchmark of the difficulty of this particular dataset.

A support vector machine (SVM) classifier with radial basis function (RBF) kernel is trained and evaluated via 5-fold cross validation using LIBSVM [23] . This is a four class problem ($C \in \{1, 2, 3, 4\}$) with the four classes being kick drum, snare drum, tom-tom and overhead. The features used in the experiment are listed in Table 2 and include mel-frequency cepstral coefficients (MFCC) (20 dimensions), spectral features and time domain features as well as information about the amount of time active audio is present in the track. The first and second derivatives of each feature (non-singleton) is also included in the dataset. This results in 138 total feature dimensions which is then reduced through principle components analysis (PCA). The classifier achieved an average accuracy across all folds of 0.504. For a four class problem, this

Features	Features (cont.)
MFCC	RMS
Centroid	Bandwidth
Flux	Zero-Crossing Rate
Number of Segments	Inter Onset Interval
Segment Length	

Table 2: Features used in drum type classification.

Production Familiarity	Participants
None	4
Novice	4
Intermediate	6
Expert	1

Table 3: Listening test participant familiarity with audio mixing and production.

result is not particularly promising, but the model and features used are not as advanced as those in [13–16]. Although the data is in multi-track format, there are still several instruments present via the bleed of the microphones. For the tom-tom drums, the majority of the track resembles an overhead microphone signal of low amplitude until the drum is (with relative infrequency) struck. This type of real-world situation increases the difficulty of performing classification.

6. LISTENING EVALUATION

To evaluate the ability of the model to appropriately mix the drum tracks together, a listening test is performed where participants noted their preference for the individual monaural tracks summed versus the mix generated with the model. The ground truth instrument labels are used for generating the mixes using the model. Ten songs were selected at random from the dataset and a 15 second clip for each song was selected so that as many of the individual drum tracks were active as possible. Most songs in the dataset do not have drum stems associated with them, only the raw unmixed multi-track session and the final professional mix. The majority of songs that do have mixed drum stems are from the same studio and use very similar processing chains. Therefore to avoid over-representing that subset we only included the summed mix and the automatic mix across a larger number of sources.

The clip pairs were presented with the summed version and the automatically mixed version appearing in random order. Each participant was presented clip pairs one at a time and asked which clip sounds more balanced. They

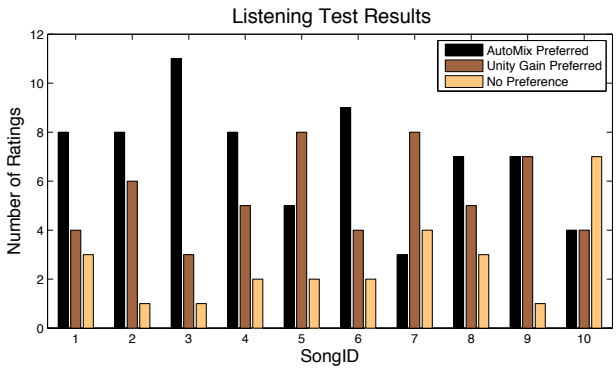


Figure 4: Listening test results showing the number of ratings for each clip pair.

could choose Clip A, Clip B or No Preference. The participants were asked to provide their level of experience with audio mixing and production, the distribution is shown in Table 3. Subjects are graduate and undergraduate students at Drexel in the music industry and engineering programs. Most subjects are male, with only two participants being female. There were 15 total participants in the study with about half having little experience working with audio production and the other half having significant experience.

Figure 4 shows the results of the listening test. For six of the ten songs, the model is preferred over the summed mix and listeners prefer two of the ten monaural summed mixes. Songs 7, 8 and 10 contain some drum loops from a library and do not adhere to the 'standard' recording technique of having kick, snare, tom and overhead microphones. The dataset represents a variety of material from various sources and varying quality. Some material is recorded professionally and sounds reasonably balanced through just summing the tracks.

7. DISCUSSION AND FUTURE WORK

We present an approach to mixing multi-track audio in an automated fashion by incorporating common techniques based on prior knowledge of instrumentation. The method obtains fair performance on a certain class of song in the dataset but is not able to gracefully handle inconsistencies in recording quality present the dataset. One caveat of working with multi-track audio is the lack of standardization for recording sessions. This makes obtaining well labeled consistent datasets to train models a difficult task in itself. The work here demonstrates the possible potential of a hierarchical system that combines both best practices and common techniques of mixing engineers with more sophisticated models of instrument identification, however there is significant room for improvement.

For the classification task, there exist more advanced methods in the literature, yet most apply to individual instrument samples and not full recorded tracks. Including more information about the temporal evolution of a signal as well as taking advantage of the audio in multiple drum tracks while classifying each track could improve results significantly.

Genre information plays a significant role in the desired drum sound for a given song. A jazz kit requires much different treatment than a dance or house drum beat, however genre recognition is not a solved problem and the definitions of genres are constantly evolving. This is an aspect of automatic mixing where it would make sense to expose a parameter to the user and offer 'presets' similar to most audio plugins.

More adaptive methods can be used on the track level processing that computes the active segments and loudness as in [7]. Perhaps the most important aspect is further user evaluation and iteration based on listening test results. The ultimate goal of automated mixing systems is to make the mix sound better to the user. Mixing audio often demands an iterative coarse-to-fine approach where the engineer is constantly making changes and then evaluating those decisions in the context of the mix [18, 19].

This is an introductory work that explores the potential of a hierarchical approach to multi-track mixing using instrument class as a guide to processing techniques. While the classification and listening evaluation results have room for improvement, a system basing mixing decisions on the instruments in the mixture warrants further investigation.

8. REFERENCES

[1] J. D. Reiss, "Intelligent systems for mixing multichannel audio," in *Proceedings of the 17th International Conference on Digital Signal Processing (DSP)*, Jul. 2011, pp. 1–6.

[2] E. Perez-Gonzalez and J. D. Reiss, "Automatic mixing," in *DAFX: Digital Audio Effects*, 2nd ed., U. Zölzer, Ed. John Wiley & Sons, Ltd, 2011, pp. 523–549.

[3] E. Perez-Gonzalez and J. D. Reiss, "Automatic gain and fader control for live mixing," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2009.

[4] ——, "Automatic equalization of multichannel audio using cross-adaptive methods," in *127th AES Convention*, 2009.

[5] S. Mansbridge, S. Finn, and J. D. Reiss, "An autonomous system for multitrack stereo pan positioning," in *133rd AES Convention*, 2012.

[6] ——, "Implementation and evaluation of autonomous multi-track fader control," in *132nd AES Convention*, 2012.

[7] D. Ward, J. D. Reiss, and C. Athwal, "Multitrack mixing using a model of loudness and partial loudness," in *133rd AES Convention*, Oct. 2012.

[8] A. T. Sabin and B. Pardo, "A method for rapid personalization of audio equalization parameters," *Proceedings of ACM Multimedia*, pp. 769–772, 2009.

[9] Z. Rafii and B. Pardo, "Learning to control a reverberator using subjective perceptual descriptors," in *Proceedings of the 10th International Society for Music Information Retrieval Conference*, Kobe, Japan, October 26-30 2009, pp. 285–290.

[10] B. Pardo, D. Little, and D. Gergle, "Building a personalized audio equalizer interface with transfer learning and active learning," in *Proceedings of the 2nd International ACM Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies*. New York, USA: ACM, 2012, pp. 13–18.

[11] D. Barchiesi and J. Reiss, "Reverse engineering of a mix," *Journal of the Audio Engineering Society*, vol. 58, no. 7, pp. 563–576, 2010.

[12] J. Scott, M. Prockup, E. M. Schmidt, and Y. E. Kim, "Automatic multi-track mixing using linear dynamical systems," in *Proceedings of the 8th Sound and Music Computing Conference*, Padova, Italy, 2011.

[13] S. Scholler and H. Purwins, "Sparse approximations for drum sound classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 933–940, 2011.

[14] A. Tindale, A. Kapur, G. Tzanetakis, and I. Fujinaga, "Retrieval of percussion gestures using timbre classification techniques," in *Proceedings of the 5th International Society for Music Information Retrieval Conference*, 2004.

[15] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, 2011.

[16] S. Essid, G. Richard, and B. David, "Musical instrument recognition by pairwise classification strategies," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 4, pp. 1401–1412, 2006.

[17] S. Hargreaves, A. Klapuri, and M. Sandler, "Structural Segmentation of Multitrack Audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 10, pp. 2637–2647, 2012.

[18] M. Senior, *Mixing Secrets for the Small Studio*, 1st ed. Focal Press, 2011.

[19] R. Izhaki, *Mixing Audio: Concepts, Practices and Tools*, 1st ed. Elsevier Ltd., 2008.

[20] E. Perez-Gonzalez and J. D. Reiss, "A real-time semi-autonomous audio panning system for music mixing," *EURASIP Journal on Advances in Signal Processing*, 2010.

[21] International Standards Organization, *ISO226: Normal equal-loudness-level contours*, 2003.

[22] W. E. Hodgetts, J. M. Rieger, and R. A. Szarko, "The effects of listening environment and earphone style on preferred listening levels of normal hearing adults using an mp3 player," *Ear and Hearing*, vol. 28, no. 3, pp. 290–297, 2007.

[23] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

TEMPO DETECTION OF URBAN MUSIC USING TATUM GRID NON-NEGATIVE MATRIX FACTORIZATION

Daniel Gärtner
Fraunhofer Institute for Media Technology IDMT
daniel.gaertner@idmt.fraunhofer.de

ABSTRACT

High tempo detection accuracies have been reported for the analysis of percussive, constant-tempo, Western music audio signals. As a consequence, active research in the tempo detection domain has been shifted to yet open tasks like tempo analysis of non-percussive, expressive, or non-western music. Also, tempo detection is included in a large range of music-related software. In DJ software, features like beat-synching or tempo-synchronized sound effects are widely accepted in the DJ community, and their users rely on correct tempo hypothesis as their basis. In this paper, we are evaluating both academic and commercial tempo detection systems on a typical dataset of an urban club music DJ. Based on this evaluation, we identify octave errors as a problem that has not yet been solved. Further, an approach based on non-negative matrix factorization is presented. In its current state it can compete with the state of the art. It further provides a foundation to tackle the octave error issue in future research.

1. INTRODUCTION

Tempo detection on percussive music with constant tempo has been extensively investigated by the music information retrieval community throughout the last 30 years, and high accuracies have been reported. Therefore, researchers have moved on to related tasks like tempo detection of non-percussive music, dealing with soft onsets, or tempo/beat tracking of expressive performances, that are more difficult to analyze correctly.

Several comparative evaluations of tempo detection algorithms have been published. The results of the ISMIR tempo induction contest of 2004 are summarized in [9]. In a recent study [19], another 12 algorithms are investigated. In both studies, [11] outperforms the competing algorithms in most of the cases. Another comparative study is presented in [14], where algorithms of seven groups are analyzed. It is shown, that the genre has an effect on the tempo detection performance. Also, algorithms performed quite differently within different tempo ranges. Furthermore, some algorithms performed much worse on songs with ternary meter compared to songs with binary meter, while in general, percussive music returned higher scores than non-percussive music.

To motivate our work, a pre-study has been conducted, in which several algorithms have been evaluated on a urban club music dataset of 1000 songs size (more details can be found in Section 3). This study revealed, that the leading academic tempo-detection algorithms reach up to 70% of accuracy on urban club music, which is less than expected (100% on reggae, soul, and rap are reported in [1], [8] achieved over 95% on constant rock and pop music).

Several metrics are commonly used for tempo detection evaluation. The fraction of songs, for which the tempo has been correctly identified, is an intuitive measure. Often, an additional metric is used [9, 19], in which tempo estimates that are an integer multiple or divisor of the ground-truth tempo are also counted as correct estimates. This metric is motivated by the fact that even human listeners will not agree on a single tempo (another approach dealing with this fact is the metric used in [14]). This is surely true for a large quantity of music from several styles. However, as reported in [13], there is a high agreement in tempo perception of urban club music amongst listeners, and it can be assumed that the agreement is even higher amongst urban club music DJs, since it is their job to mix songs with the same tempo. However, this assumption remains to be proven.

From the perspective of the user of a DJ software, it is absolutely mandatory that the tempo is annotated correctly. The so called octave errors are unacceptable. Although not really related to the origin of the word octave in music, they refer to tempo estimates that are different from the correct tempo by a factor that is a power of two. Songs need to have the same tempo in order to be mixed with clean transitions, which is requested by the dancing audience. Further, many audio effects also rely on correct tempo hypotheses. And of course, additional processing like beat tracking, which can be used for automatic mixing, also strongly depends on a correct tempo estimation.

[13] shows that there is a significant effect of music genre on the most salient tempo, which is consolidated in [16], where a style detection method is used to improve tempo detection of ballroom dance songs. Further, [4] lists several cues that beat-tracking might profit from a style-specific analysis.

Besides a limited tempo range of about 60 to 140 bpm,

urban club music is percussive, has a constant tempo, and is composed of repeating drum patterns. Further, the drums are often sampled and triggered by a sequencer. This reduces tempo fluctuations and the variety in sound of multiple occurrences of a certain drum.

The remainder of this paper is organized as follows. In Section 2, the proposed system is explained, followed by the experimental setup and results section (3). The paper concludes with a summary and outlook in Section 4.

2. APPROACH

As [3] states, "Perceptually, musical metric structure comprises beats and hierarchies. Beats constitute the framework in which successive musical events are perceived". The number of beats per minute is used to quantify the tempo of a musical piece, the beat grid specifies the position of the beats in a musical piece. The beat grid can be further subdivided, leading to the tatum grid (or just tatum), which is the "lowest regular pulse train that a listener intuitively infers from the timing of perceived musical events" [10]. Also, beats can be grouped in bars. In urban music, a bar mostly consists of four beats. On each metrical level we will use the term grid to address the positions of the pulses, and the term period for the time or frame interval between two successive pulses.

An overview of our approach is shown in Figure 1. From the spectrogram of the audio data, quantized event bands are calculated from which the bar period is estimated, that directly leads to the tempo hypothesis. In order to extract the quantized event bands, a tatum tracker is applied to an onset detection function. Next, the spectrogram is sampled at tatum grid positions and then factorized in event bands. This is done by means of non-negative matrix factorization (NMF [12]), aiming at also isolating different drum classes (bass drum, snare drum) in separate bands. Calculating NMF on just a subset of spectrogram frames is a major difference to our approach in [7].

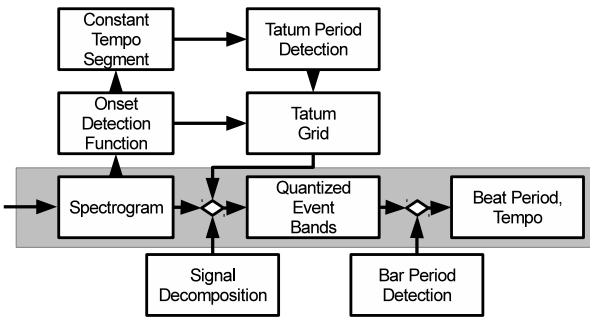


Figure 1. System Overview.

2.1 Onset Detection Function

At the beginning of the analysis, events that are supposed to contribute to the perception of beats in the musical piece are determined. As this study focuses on urban music, percussive drum events are investigated, and an onset detection function that is sensitive to percussive events is chosen. An absolute spectrogram S is calculated from the 44.1 kHz audio input data, using a window-size of 4096 samples size with a hop-size of 512 samples, which corresponds to a spectrogram sample rate of 86.1 Hz.

From S, an onset detection function d_o is calculated. The onset components detection function developed in [8] is used as onset detection function.

2.2 Constant Tempo Segment

In this component, the longest segment with approximately constant tempo is identified. Analyzing audio data with several distinct tempi will harm the tatum period detection.

d_o is convolved with a set of 601 comb grids, which correspond to pulse trains of 20 s length from quarter notes between 40 bpm and 640 bpm (sixteenth notes at 160 bpm). Next, each convolved function is cut into segments of 1 s length, and the maximum value inside each segment is stored in a so called comb response matrix.

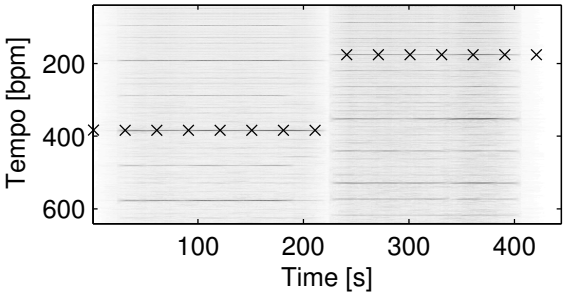


Figure 2. Comb response matrix of a song with multiple tempi, including the Viterbi path.

An example of a comb response matrix is shown in Figure 2. The tempo is constant over the first half of the song, and then changes to another tempo. A strong response for the grid corresponding to about 380 bpm, which equals a pulse train of 16th notes at 95 bpm, can be observed in the first half. The second half is a little slower.

Next, the entries in the comb response matrix are normalized to unit sum for each column. Now, each entry in the matrix can be seen as likelihood for observing a bpm class at a given time in a hidden Markov model (HMM). The Viterbi algorithm [18] is then used to track the most likely path through the comb response matrix of observations. From the path, the segment where the tempo stays roughly constant for the longest time is determined and further investigated during tatum period detection.

2.3 Tatum Period Detection

Since d_o is an onset detection function sensitive to percussive events, it can be used for tatum period detection in the following way. An accent function g_o is calculated from the constant-tempo excerpt of d_o by keeping only the local maxima of d_o while setting all other values to zero, and convolving the resulting signal with a hann window (11 samples width). The tatum period detection function g_t is

calculated: $g_t = R(R(g_o))$, where $R(f)$ is the autocorrelation of function f. The first 400 values of g_t can be seen in Figure 3. Now, for each local maximum position on the lag

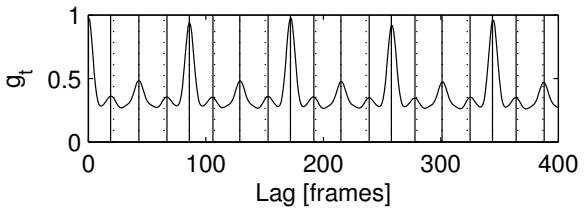


Figure 3. The tatum period detection function g_t. Values close to 1 indicate a high periodicity with a period of the corresponding lag axis index. Solid vertical lines denote local maxima positions, dotted vertical lines denote multiples of the tatum period.

axis as tatum period candidate c_i, we determine the smallest multiple m_i of c_i that does not correspond to a local maximum in g_t. The tatum period candidate c_i with the largest m_i is chosen as tatum period if it is also the local maximum with the smallest lag. Otherwise, it's lag value is divided by 2 until it is smaller than 14, which still allows tracking sixteenth notes at 180 bpm. This additional step is required, since not even all the rhythmic events necessarily fall exactly on the tatum grid. If a certain drum event in a bar is systematically played early or late, this will lead to slightly shifted locations of local maxima in g_t. This can also be observed in Figure 3, where the odd local maxima (solid lines) do not exactly overlap with the tatum period and its multiples (dotted lines).

2.4 Tatum Grid

After having determined the tatum period, the onset detection function d_o is convolved with a comb grid where the combs are tatum period spaced. This will strengthen accents in d_o that lie on the not yet determined tatum grid, and extenuate all other accents. Now, all dominant local maxima positions and their pairwise differences are determined. Only those pairs, for which the difference is approximately 1, 2, or 3 times the tatum period, are kept. Then, dynamic programming [2] is used to find the longest path over the remaining pairs. Using this approach, variations around a center tempo can be compensated. However, if a song contains multiple different tempi, the path will only cover the region that has a beat period that is approximately a multiple of the tatum period hypothesis. In [6], dynamic programming is used for beat tracking. The approaches differ mostly in the fact that we only allow local maxima in the underlying detection function as potential tatum grid anchors.

2.5 Signal Decomposition

The determined tatum grid positions are refined to the locations of close local maxima in d_o. This way, also early and late played events can be incorporated. Then, the spectrogram is subsampled at the tatum grid positions, and the resulting matrix is factorized using NMF [12]. In NMF, a non-negative matrix V is factorized in matrix factors W and H, $V \approx WH$. Applied to an absolute spectrogram, it is factorized in a set of components where the characteristic spectrum of each component is stored in W and the activation of each component is stored in H. From factorization, we expect to separate different drum classes in different bands, and also separate additional, sources in additional bands (e.g., separate bass drum and bass even though they might be overlapping in the spectrogram). Since the drum tracks in urban club music are often generated using drum machines, samplers, and sequencers, NMF seems to be a good choice for decomposition, since drum sounds are not supposed to vary over time. In [7] we observed better results in tempo recognition on urban music using NMF based decomposition compared to a filter bank approach.

NMF is used to factorize the subsampled spectrogram in 24 components. Both bases and activations are randomly initialized, and 50 iterations of multiplicative update rules using Kullback-Leibler divergence are performed. These parameters have not yet been quantitatively optimized in any way.

NMF has been shown to be able to separate drum events in, e.g., [15], where NMF is used for drum transcription in polyphonic music.

Factorizing a spectrogram that is subsampled at tatum grid positions instead of the full spectrogram reduces the computational load. A more general advantage of working on a tatum grid level lies in removing tempo variations.

However there are disadvantages as well. The decomposition is dependent on the quality of the tatum grid. If the tatum grid does not contain the drum event onsets, the drums will not be analyzed at all. Further, in cases where different drum events fall, for example, on the same beat but at slightly different times, the spectral frame that is meant to capture these events might not capture them all.

2.6 Bar Period Detection

The 24 bands obtained by the NMF analysis are again analyzed using a comb grid. Each of the bands is convolved with a set of comb grid with combs spaced 1, 2, ... , n, where n corresponds to the largest possible distance between 4 beats at 65 bpm (see Section 3.5.2), considering the underlying tatum grid. For each band, from the corresponding n filtered functions, the index of the one with the largest variance is determined.

From all 24 collected indices, a histogram is calculated and the most frequent index is chosen as tatum-period-to-bar-period factor z. The bar period is calculated by multiplying Z and the tatum period. As final steps, the determined bar period is first divided by four to retrieve the beat period, since 4/4 is the most common measure in urban club music. As a final step, the retrieved beat period is transformed in the bpm range of 65 bpm to 130 bpm (see Section 3.5.2) by doubling or halving.

3. EVALUATION

In this section we describe the experiments, present the results, and discuss them.

3.1 Measures

For each evaluation run, we report the relative number of songs with a tempo hypothesis that differs less then 4% from the ground-truth tempo (Acc1). To gather further insights on the performance of the evaluated algorithms, we further report the relative number of songs with two times or three times the correct tempo (Acc2), and one half or one third of the correct tempo (Acc3). The fraction of the remaining songs is denoted Acc4.

3.2 Dataset

2324 songs have been collected from an urban music promotion platform exclusively for DJs. Labels provide songs to DJs over this kind of platforms at no charge, in return the DJs will help to promote these songs and make them popular by playing them in their sets in the club. It is an authentic set that well represents the kind of music, urban club music DJs are working with. Each of the songs has been tempo-annotated by an experienced urban music DJ. The set has been randomly split into a development set (1000 songs, denoted *dev*) and a test set (1324 songs, denoted *test*).

Dev has been used for the development of the algorithm, which includes the design of the components and parameter setting. The final algorithm has then been evaluated using *test*, the results for *dev* are also reported.

Figure 4 shows a histogram over the observed tempi in *dev*.

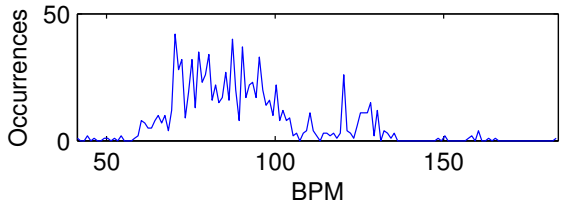


Figure 4. Histogram over the ground truth tempi in *dev*

Although it has been argued, that urban club music in general is strongly percussive and songs have a constant tempo, there are exceptions. A few songs in the dataset do not contain any or only soft percussions. Some are played by live bands (e.g., The Roots) which leads to varying tempo. Another source for varying tempo is the use of sampling in the music, where the samples vary according to their tempo. And there are even a few songs that are cut together in a way that the segments are not concatenated on beat, or even the tempo is different for the segments. Further, a few songs in the database do not belong to the urban music genre, like some pop rock songs, that are also responsible for the upper tempo outliers in Figure 4. We decided to keep them in the database, since we wanted to keep it exactly the way it was obtained. However, genre detection algorithms could be used to identify songs like that prior to tempo analysis.

3.3 Preprocessing

The data consists of MP3 files of different sample rates and bit rates. FFmpeg has been used to convert the files to 44.1 kHz mono PCM wav files.

3.4 Algorithms

The approaches from Dixon [5], Ellis [6], and Klapuri [11] have been selected to represent the academic systems. [6] returns two tempo hypotheses, from which the stronger one is selected. [11], and [5] return a beat grid. We calculated histograms over inter-beat-intervals, and then determined the mean of all inter-beat-intervals that contributes to the most frequent inter-beat-interval, from which the tempo in bpm can be derived. All implementations were obtained from the authors.

In addition, several commercial DJ systems have been evaluated. Each one offers ways to parameterize the tempo analysis algorithm. Cross 1.7.0, denoted Cross [1], offers three different tempo ranges for bpm analysis, each one covering exactly one octave. 75-150 has been selected for analysis. Scratch Live 2.4.1, denoted SL [2], offers five different tempo ranges for analysis, each one covering one octave. 68 - 135 has been selected for analysis. Torq 2.0.3, denoted Torq [3], offers several genre-specific tempo ranges. In our experiments, the default settings have been used, returning bpm values from 60 to 160 bpm. Traktor Pro 2.6.0, denoted TraA+B [4], offers 9 different octaves, from which 68-135 (TraB) has chosen. Further, Traktor also offers a single range covering more than a octave (60-200), which will be denoted TraA. Virtual DJ Home 7.0.5, denoted VDJ, offers an option to allow also bpm values smaller than 80 bpm, which was activated. It returned tempi between 60 and 170 bpm. It is worth noticing, that some of the investigated tools only offer tempo ranges of exactly one octave, which is a simple but (as can be seen in the Section 3.5) working approach to reduce octave errors at least for urban club music, since a large amount of urban club music is located inside a single octave.

3.5 Results

In this section the results of the conducted experiments are presented and discussed. All experiments have been performed in Matlab.

3.5.1 Evaluation of the reference systems

Table 1 lists the results returned from the evaluation of the state of the art.

Directly comparing the results for *dev* and *test*, one can see that the performances are similar, which indicates that both sets are comparably difficult. This is also true for the

[1] http://www.mixvibes.com
[2] http://serato.com/scratchlive
[3] http://www.torq-dj.com/
[4] http://www.native-instruments.com/traktor

	Acc1 1	Acc2 2+3	Acc3 1/2+1/3	Acc4 other
Cross	73.2/75.6	23.2/22.9	1.2/0.5	2.4/1.0
SSL	89.4/89.4	8.2/8.5	1.3/1.2	1.1/1.0
Torq	85.6/84.3	2.9/4.4	4.3/3.4	7.2/7.9
VDJ	81.0/78.5	17.0/20.4	1.1/0.9	0.9/0.2
TraA	77.6/79.1	15.3/14.7	5.1/4.5	2.0/1.7
TraB	90.3/90.7	6.1/6.2	1.5/1.4	2.1/1.7
Dixon	25.3/24.5	69.8/70.1	0.0/0.0	4.9/5.4
Ellis	57.5/51.5	4.5/5.4	19.3/26.5	18.7/16.5
Klapuri	68.7/71.7	28.8/27.2	1.8/0.8	0.7/0.3

Table 1. Results for the state of the art algorithms on *dev* / *test*, accuracies in %.

results of the proposed approach, listed in Table 2, which shows, that even though it has been optimized using *dev* it still generalizes well.

With an accuracy of 73.2%, Cross is the worst of the commercial algorithms. This is mainly caused by the limited choices of the tempo range, of which no one really fits our data well. Since all commercial algorithms do a pretty good job in choosing the correct tempo (as long as octave errors are still accepted as correct), the performance mainly depends on the prior choice of the bpm-analyzing octave. Traktor offers both a large tempo range (60-200, TraA) and a suitable octave tempo range (68-135, TraB). TraB has an accuracy of about 12.7% higher than TraA, which shows, that automatically picking the right tempo octave is still an open issue.

Dixon often returns twice the correct tempo (69.8%) and could be simply tuned by halving the returned tempo estimate. For Ellis, almost 20% of the songs in *dev* are neither correct nor do they belong to one of the octave error classes. The most common cases in "other" of Ellis are 4/3 (15%) and 2/3 (4%). In accordance with the mentioned MIREX benchmarks, Klapuri is the best performing academic algorithm.

In the first category (Acc1), the commercial approaches outperform the academic ones. In this category, the commercial approaches can benefit by the fact that most of the data is located in a single octave. However, in the fourth category (other), Klapuri's algorithm is among the best.

3.5.2 Best Tempo Octave

Based on the data from *dev*, an experiment has been conducted to determine the best tempo octave settings, assuming that an algorithm performs perfectly but transforms its results in a specified tempo range of exactly one octave by doubling or halving the tempo several times. The accuracy depending on the given tempo range is plotted in Figure 5. The best performance (92.2%) is achieved for a bpm range of 65 - 130 bpm. Therefore, in the presented approach, a tempo hypothesis is transformed in this range by doubling and halving.

3.5.3 Evaluation of the proposed approach

Table 2 contains the results for the evaluation of the presented algorithm. For both *dev* and *test*, the algorithm returns the highest accuracies retrieved in the whole study.

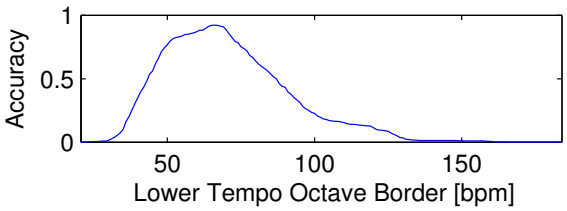


Figure 5. Accuracy of a perfect algorithm with octave restriction depending on the allowed range.

	Acc1 1	Acc2 2+3	Acc3 1/2+1/3	Acc4 other
Own rel.	91.9/92.5	4.9/4.5	2.9/2.7	0.3/0.2
Own abs.	919/1225	49/60	29/36	3/3

Table 2. Results for the presented algorithm on *dev*/*test*, accuracies in %, and absolute number of songs in each category.

For both *dev* and *test*, only three songs return a tempo that is not 2^n times the correct tempo. The three failing songs in *dev* all have a 3/4 measure, but a 4/4 measure is assumed when going from bar period to beat period. Two of the failing songs in *test* have no or almost no drum tracks, the third one has also a measure of only 3 beats length.

The chosen tempo range (that has been determined from the bpm distribution in *dev*) induces the assignment to one of the accuracies Acc1, Acc2 and Acc3. An Acc1 of 92.5% confirms the choice of the bpm range.

	1	2	3	4	6	8	9	12	other
dev	1	188	8	761	6	32	1	3	0
test	0	279	7	992	5	41	0	0	0

Table 3. Distribution of the songs in tatum classes.

Table 3 shows the performance of the tatum period detection component. The songs are differentiated into tatum classes, where the class name denotes the ratio of ground truth beat period and determined tatum period. For all instances, the beat period is an multiple of the tatum period. The most common multiple is 4, which corresponds to 16th notes. Assuming 16th notes as tatum for each song, and therefore estimating the beat period as four times the tatum period, the tempo could be correctly determined for 76.1% (*dev*) and 74.9% (*test*) of the songs without making any assumptions on a bpm range, which outperforms any of the academic approaches.

The distribution of the songs in bar classes is listed in Table 4. For both sets, for most of the songs a bar-length of 4 beats is returned. Assuming the determined bar period to be four times the beat period, the tempo could be determined correctly for 81.5% (*dev*) and 82.0% (*test*) respectively, without making any assumptions on a bpm range, which again outperforms any of the academic approaches.

Both tables reveal the strength of the algorithm, but at the same time also show limits regarding to octave determination, as also shown in [7].

	1	2	3	4	6	8	other
dev	4	166	3	815	0	12	0
test	0	214	0	1085	1	22	2

Table 4. Distribution of the songs in bar classes.

4. CONCLUSION AND OUTLOOK

In this paper, it is shown that finding the correct octave is still an issue for even urban club music. This claim is consolidated by evaluating several academic and commercial tempo detection algorithms on a urban club music data set. The presented algorithm, developed specifically for urban music, outperforms all the other algorithms evaluated in this study, estimating the correct tempo for 92.5% of the *test* set. The remaining songs are a 2^n multiple of the correct tempo except for 3 out of 1324 songs.

Therefore, the proposed algorithm provides a good basis for further processing, in which the correct octave has to be determined. In the current approach, all tempo values are forced to be in the range of 65 to 130 bpm. Further experiments will be conducted where the octave is chosen based on the musical structure. A first investigation on the tatum-quantized activations indicated, that they still capture the dominant drum events, contributing to the rhythm of a song. Therefore, drum pattern analysis, as as performed in, e.g., [17] could be carried out on the activations, and then be incorporated in tempo detection. Finding the characteristic drum pattern of a song also offers additional opportunities like drum pattern similarity or urban music sub-genre classification.

5. ACKNOWLEDGEMENT

We want to thank Simon Dixon, Daniel P.W. Ellis, and Anssi Klapuri for providing implementations of their algorithms.

6. REFERENCES

[1] Miguel Alonso, Bertrand David, and Gael Richard. Tempo and beat estimation of musical signals. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, 2004.

[2] Richard Ernest Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[3] Jeff A. Bilmes. *Timing is of essence*. PhD thesis, Massachusetts Institute of Technology, 1993.

[4] Nick Collins. Towards a style-specific basis for computational beat tracking. In *Proceedings of the 9th International Conference on Music Perception & Cognition*, 2006.

[5] Simon Dixon. Evaluation of the audio beat tracking system beatroot. *Journal of New Music Research*, (36), 2007.

[6] Daniel P. W. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[7] Daniel Gärtner. Tempo estimation from urban music using non-negative matrix factorization. In *Proceedings of the 42th AES International Conference*, pages 208–215, Ilmenau, 2011.

[8] Masataka Goto. A real-time beat tracking system for audio signals. In *Proceedings of the International Computer Music Conference*, 1995.

[9] 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 Speech and Audio Processing*, 14:1832–1844, 2006.

[10] Tristan Jehan. *Creating Music by Listening*. PhD thesis, Massachusetts Institute of Technology, 2005.

[11] Anssi Klapuri, Antti 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.

[12] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 2001.

[13] Martin F. McKinney and Dirk Moelants. Ambiguity in tempo perception: What draws listeners to different metrical levels? *Music Perception: An Interdisciplinary Journal*, 24(2):155–266, 2006.

[14] Martin F. McKinney, Dirk Moelants, Matthew E. P. Davies, and Anssi Klapuri. Evaluation of audio beat tracking and music tempo extraction algorithms. *Journal of New Music Research*, 36(1):1–16, 2007.

[15] Jouni K. Paulus and Anssi Klapuri. Drum transcription with non-negative spectrogram factorisation. In *Proceedings of the 13th European Signal Processing Conference (EUSIPCO)*, 2005.

[16] Björn Schuller, Florian Eyben, and Gerhard Rigoll. Tango or waltz?: Putting ballroom dance style into tempo detection. *EURASIP Journal on Audio, Speech, and Music Processing*, 2008(6):1–12, 2008.

[17] Christian Uhle. *Automatisierte Extraktion rhythmischer Merkmale zur Anwendung in Music Information Retrieval-Systemen*. PhD thesis, Technische Universität Ilmenau, Ilmenau, 2008.

[18] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

[19] Jose R. Zapata and Emilia Gómez. Comparative evaluation and combination of audio tempo estimation approaches. In *Proceedings of the 42th AES International Conference*, Ilmenau, 2011.

A STUDY OF CULTURAL DEPENDENCE OF PERCEIVED MOOD IN GREEK MUSIC

Katerina Kosta, Yading Song, György Fazekas, Mark B. Sandler
Centre for Digital Music, Queen Mary University of London
firstname.lastname@eecs.qmul.ac.uk

ABSTRACT

Several algorithms have been developed in the music information retrieval community for predicting mood in music in order to facilitate organising and accessing large audio collections. Little attention has been paid however to how perceived emotion depends on cultural factors, such as listeners' acculturation or familiarity with musical background or language. In this study, we examine this dependence in the context of Greek music. A large representative database of Greek songs has been created and sampled observing predefined criteria such as the balance between Eastern and Western influenced musical genres. Listeners were then asked to rate songs according to their perceived mood. We collected continuous ratings of arousal and valence for short song excerpts and also asked participants to select a mood tag from a controlled mood vocabulary that best described the music. We analysed the consistency of ratings between Greek and non-Greek listeners and the relationships between the categorical and dimensional representations of emotions. Our results show that there is a greater agreement in listener's judgements with Greek background compared to the group with varying background. These findings suggest valuable implications on the future development of mood prediction systems.

1. INTRODUCTION

A large body of research [3, 5, 16] supports that music can either evoke emotions in listeners, a phenomenon known as felt emotion, or express emotion, known as perceived mood. For this reason, understanding emotion or mood [1] is useful in daily experiences listening to music. Several Music Information Retrieval (MIR) related studies [2, 19, 20] have demonstrated the importance of mood-based organisation when accessing music catalogues. For instance, Lee [20] showed that participants of a survey conducted to assess the relevance of different modalities in music searching and browsing would use emotional or mood states in their queries. More recently, Bischoff [2] reported that over 15% of the song queries on the Web-based music service Last.fm were made using mood tags. MIR systems that facilitate the use of mood to interact with music rely on specific mood models. For instance, categorical organisation, or dimensional representation of emotion stated in a mathematical space associates mood with musical audio, tags or other information sources using selected machine learning algorithms [3, 9]. In this paper, a dimensional representation of emotion is demonstrated.

A wealth of research focuses on the above problems in the context of Western musical culture [5, 29], assuming that generic models can be built independently from musical style, culture, genre or listeners' acculturation. This study investigates these issues in the context of Greek musical culture. In particular, we focus on the following research questions:

1. Do Greek and non-Greek people agree with each other on the ratings of perceived arousal and valence?
2. Are there any differences in the music perception of Western and Eastern influenced musical genres between Greek and non-Greek people?
3. Are specific mood labels originated from Western musical studies suitable for Greek music?

The rest of the paper is organised as follows: In Section 2, related work on the field is presented as well as cases that show why Greek music is particularly relevant for study. Due to the unique properties of Greek musical culture, we investigate the culture dependence of valence-Arousal (VA) ratings and tagging in the context of Greek music. Section 3 outlines the mood classification models and music perception background considered. In Section 4, we analyse the database collection that we created and used. In Section 5 we present the design of the study, while in Section 6 we show our results. In Section 7 we conclude and outline some directions for the future work.

2. BACKGROUND

2.1 Cross-Cultural Mood classification

It is well known that music emotion perception is influenced by cultural background. For instance several studies have attempted to explore the emotion perception of pop songs between American and Chinese listeners [33], the music emotion classification between Chinese and English songs [14], and perceived complexity of Western and

[1] Albeit we acknowledge the difference between emotion and mood, in this work, the terms will be used interchangeably.

African folk melodies by Western and African listeners [6]. To our knowledge, no previous studies have been performed using Greek music which holds particular interest as it features compositions influenced by both Eastern and Western culture.

2.2 Why do we use Greek Music?

Greek music databases do exist, and many of them can be found on the Internet. However they are either related to specific music labels or come from individuals, hence they are limited in their scope. A broader database is provided by the Greek Music Information Centre of the Institute for Research on Music and Acoustics [2]. This database however lacks popular and modern Greek music metadata.

A study for using principles of MIR Systems in Greek Music is described in [17], where a case study for the usage of the "Music Control" system by Nielsen - VNU is presented. This system provides an airplay monitoring service based on MIR principles, but it struggles with two kinds of issues. Firstly, although it could be efficiently used in the context of musical genres such as pop and rock, it could not be utilised for Greek musical genres such as "Entehno" (described as "Alternative" in Section 3). Secondly, the system was misleading due to the "secondary importance" of the international musical menu, counting their airplay time. The above indicate the uniqueness of Greek music, which provides the motivation for our further exploration.

3. MOOD CLASSIFICATION AND MUSIC PERCEPTION

3.1 Representation of Emotions

Several representations of emotion or mood have been proposed in psychology and related disciplines. For recent thorough reviews in the context of music see for instance [5] or [3]. Among these representations, the *categorical model*, which assumes that emotion may be represented as a set of distinct categories, and the *dimensional model*, which characterises emotions using a small number of dimensions corresponding to the internal human representations are paramount. Categorical approaches have been criticised since they constrain emotions to be represented as a set of predefined families or landmarks [23]. As a result, dimensional models have emerged as dominant, with perhaps the most influential work being Russell's circumplex model of affect [25]. This consists of a two-dimensional circular structure involving the core affect dimensions of *arousal* and *valence*. A space corresponding to these dimensions is often referred to as the VA space. The relevance of these dimensions in music, where emotions may be defined in terms of arousal or energy (i.e. how exciting or calming musical pieces are) and valence or stress, (i.e. a dimension of positive or negative emotions), were experimentally validated by Thayer [30].

[2] http://www.musicportal.gr/?lang=en

3.2 Music Emotion Recognition

Early music emotion recognition (MER) studies focused on categorical models and approached emotion recognition from audio as a classification or auto-tagging problem (see e.g. [21, 28]). However, due to the limitations of categorical models, (e.g. ambiguity in the meaning of adjectives associated with emotion categories, and the potential heterogeneity of the taxonomical organisation) recent studies have focussed on continuous valued dimensional models. In the first study that addresses these issues [34], MER is formulated as a regression problem in order to map high-dimensional features extracted from audio to the VA space directly. However, the dimensional model also presents some barriers, for instance, it is difficult to use in browsing and searching, where interfaces based on generic or music related emotion tags dominate.

3.3 Music mood tags

In order for dimensional representations to be ultimately useful in conventional browsing interfaces, it is necessary to establish connections between continuous valued emotion states and mood tags. As a solution, one may use the Affective Norm for English Words (ANEW) database [1] which contains VA values for a large number of English words derived from psychological experiments. However, these values are not validated in the context of music. More recently, Saari and Eerola [26] proposed a technique called Affective Circumplex Transformation (ACT) which uses Vector Space Modelling, Latent Semantic Analysis (LSA), Multi Dimensional Scaling (MDS) and Procrustes analysis [10] to derive dimensional mood models from music related social tags (such as those available form Last.fm) whose axes, among other possible dimensions, may be interpreted in terms of VA. This method has been validated against human ratings using a listening test, and found to be robust in the context of 600 songs drawn predominantly from Western music genres. It was also shown to be useful in the context of curated editorial tags used by production music libraries [27].

These studies, as well as most recent works in MIR and music emotion recognition have attempted to use generic mood models assuming that the relation between continuous emotion states and mood words is independent from musical style, genre or culture. A notable exception is the work of Eerola [7] demonstrating the genre specificity of mood modelling. In this present work, we investigate the last issue further.

4. GREEK MUSIC DATABASE AND ANALYSIS

In order to perform our study, a music database was created which includes tracks from the Greek musical scene from the 1970's until recent years, containing the most critically acclaimed compositions. It is a continuously growing resource, containing 2087 songs at present, all taken from personal collections [3].

[3] The Greek songs metadata can be found online at http://greek_music_data.isoftcloud.gr

In order for efficient categorisation, certain descriptions such as genre titles have been taken from the Western musical vocabulary, however, new genre titles have also been introduced to best describe the music. This is crucial because of the principles described in [31] and also because the structural categorisation that musical genres provide is important for MIR systems. The final number of the categories as shown in Table 1, is 25. Out of these, 16 genres identified using grey cells in Table 1 were used for our experiment.

Western Influenced	
Alternative Pop (AP)	Pop (P)
Ballad (B)	Progressive Rock
Blues	Rap (Ra)
Dance (D)	Reggae (Re)
Electronic	Rock (R)
Hip Hop (HH)	Ska
Latin	Tango
Lounge	Trip Hop
New Age (NA)	
Eastern Influenced	
Alternative (A)	Laiko-Pop (LP)
Cretan	Laiko-Rock (LR)
Experimental (E)	Tsifteteli (T)
Laiko (L)	Zeibekiko (Z)

Table 1. Greek Database Musical Genres. Samples from the genres indicated using grey background have been used for the experiment.

The primary demarcation between Western and Eastern influenced musical groups depends on the song's rhythm pattern and the origination of the instruments used. The genres presented as Western influenced are defined in similar terms as corresponding music in standard Western culture.

The Eastern influenced genres consist of the following categories: "Alternative" includes compositions which either have specific tempo variations, or are influenced by different Eastern musical styles within the same piece. "Cretan" is folk music originated from the Greek island of Crete. The music pieces in "Experimental" comprise experimental music with a strong presence of Eastern culture. "Laiko", or otherwise "General folk" includes compositions inspired by popular old folk song arrangements, usually following the rhythm known as "Syrtos". It is a rhythm of the round dance, originating from ancient Greece that was claimed by ancient Greek theory "for the heroic hexameter, that is, the rhythm of Homer". It consists of three counts, in which "the first is longer by one-half than the second and third" [11]. "Laiko-Pop" and "Laiko-Rock" are defined as large sub-categories of "Laiko", where the influence of rhythm patterns and instrumentation from "Pop" and "Rock" respectively are present. The two last genres are "Tsifteteli" and "Zeibekiko" named for their respective dance styles. Tsifteteli is constituted by pieces whose rhythm contains two counts, while Zeibekiko has similarities with the turkish dance "Zeybek" following the rhythm of nine counts [24].

5. EXPERIMENT DESIGN

5.1 Database Sampling

108 songs have been selected in total from a sub-category of our database with their lyrics in Greek. We used pseudo-random sampling observing several criteria to ensure balanced genres and artists. The output of the algorithm selects songs such that there is a maximum 7 songs per genre, with each artist appearing no more than once. The most representative 30 second musical excerpt was selected for each song manually such that it has a lyrical part, except for 3 excerpts from the "Experimental" genre.

5.2 Listening test

Our experiment includes an online listening test. Its first page welcomes the user and introduces the subjects with crucial instructions. After selecting either the English or Greek-translated version of the test, the system presents a form to fill for personal information, including name, age range, gender and nationality. Moreover, it presents the user with a questionnaire about his/her musical culture and about the type of music that he is most familiar with.

The second part consists of the questions detailed below, following the Goldsmiths Musical Sophistication Index, about the musical training level [22]. Here, the answers to the questions 1-3 may be given as discrete ratings on a 7 point Likert scale: (1: Completely disagree, 2: Strongly disagree, 3: Disagree, 4: Neither agree nor disagree, 5: Agree, 6: Strongly agree, 7: Completely agree).

1. *I have never been complimented for my talents as a musical performer*
2. *I can't read a musical score*
3. *I would not consider myself a musician*
4. *For how many hours I engage in regular, daily practice of a musical instrument?*
5. *At the peak of my interest, how many hours per day did I practice on my primary instrument?*
6. *For how many years have I played or sung in a group, band, choir, or orchestra?*
7. *For how many years did I have formal training in music theory?*
8. *For how many years did I have formal training on a musical instrument?*
9. *How many musical instruments can I play?*

The following listening test procedure is repeated for each song; the subject is presented with an audio excerpt and rates the mood suggested by the music in terms of valence and arousal level by selecting a location on a 2D plane representation. This user interface was inspired by that of the MoodSwings game developed to collect music emotion labels from human listeners [32]. Then, a list of the twenty closest emotion tags to the selected position is proposed. The subject may choose the one that best describes the suggested mood. The tag positions were derived from Last.fm data using the ACT transformation [26] mentioned in Section 3.3.

An optional post-experiment questionnaire was sent to all subjects after completing the listening test. The possible answers were designed to use the same rating scale as before:

Rate the following 2 sentences:
a. I could find a tag in the list that was exactly (or really close to) what I wanted to label for the perceived mood of every song.
b. I listened to a song from my favourite genre, at least once.

5.3 Participants

Twenty-two Greek participants (10 male and 12 female) and twenty non-Greek participants (15 male and 5 female) took part in the study. Their ages ranged between 18 to 54. They were recruited without consideration of their musical training. Their musical training level was assessed using the questionnaire shown in Section 5.2. The overall results are shown in Table 2.

	Min	Max	Median	SD
Non-Greek music score	21	43	30	6.07
Greek music score	21	41	30	5.42

Table 2. Musical training level score for Greek and non-Greek participants: min. and max. value, median and Standard Deviation. Ascending scale from 9 to 63.

6. RESULTS

To answer the research questions as described in Section 1, the following four experiments were carried out. The results were aggregated separately in Eastern-influenced and Western-influenced musical genres.

6.1 Effects of Eastern-influenced and Western-influenced musical genres across Greek and non-Greek listeners on valence and arousal (VA) ratings

To find out the effects of culture-biased musical genres between Greek and non-Greek listeners on the VA ratings, we used the two-way analysis of variance (factor 1: musical genre, Western-influenced and Eastern-influenced, factor 2: musical culture, Greek and non-Greek). Significant results were found in the ratings of valence for nationalities ($F(1,0.08) = 72.56$, $p<0.001$) and Eastern-Western influenced genres ($F(1,0.005) = 4.5$, $p<0.05$). Similar results were revealed in the ratings of arousal on nationalities ($F(1,0.19) = 167.16$, $p<0.001$), Eastern-Western influenced musical genres ($F(1,0.006) = 5.7$, $p<0.05$) as well as the interaction of these two factors ($F(1,0.00524) = 4.67$, $p<0.05$). However, no significant result was found for the interaction between culture and nationality on valence ratings.

6.2 Comparison of Greek and non-Greek responses on VA

To compare the difference in non-Greek and Greek responses on VA ratings, the Wilcoxon Signed Rank test was carried out across all the listeners on the mean value of VA ratings. A significantly higher standard deviation value was found in the ratings of non-Greek people for both valence ($p<5.1427e-13$) and arousal ($p<6.6826e-17$). This suggests that the ratings of Greek listeners are more consistent than those of non-Greek listeners, since Greek listeners are more familiar with these musical excerpts.

6.3 Comparison of emotion perception across Western influenced and Eastern influenced musical genres between Greek and non-Greek participants

As mentioned in section 6.1, (Western-influenced or Eastern-influenced) musical genres have an effect on VA ratings. Previous studies also showed a link between particular emotions and specific musical genres [8]. Therefore, we investigated whether in Greek music, a certain musical genre was linked to a category of emotions. The Wilcoxon Signed Rank test was conducted to compare the mean VA ratings between non-Greek and Greek listeners.

		Greek		Non-Greek		*P-Value*
Genre		**Mean**	**SD**	**Mean**	**SD**	
Eastern Infl.						
A	V	0.38	0.10	0.33	0.06	0.08
	A	0.55	0.06	0.43	0.06	**0.02**
E	V	0.36	0.06	0.32	0.04	0.31
	A	0.42	0.10	0.37	0.07	0.31
L	V	0.68	0.08	0.50	0.10	**0.02**
	A	0.63	0.07	0.55	0.06	**0.05**
LP	V	0.64	0.10	0.66	0.06	0.47
	A	0.51	0.11	0.61	0.07	**0.02**
LR	V	0.57	0.09	0.60	0.11	0.30
	A	0.42	0.09	0.44	0.05	0.69
T	V	0.68	0.06	0.61	0.07	0.08
	A	0.54	0.09	0.53	0.07	1.00
Z	V	0.61	0.09	0.47	0.07	**0.02**
	A	0.40	0.05	0.47	0.07	0.22
Western Infl.						
AP	V	0.58	0.07	0.58	0.06	**0.58**
	A	0.59	0.08	0.56	0.06	0.47
B	V	0.51	0.16	0.47	0.14	0.22
	A	0.42	0.06	0.42	0.05	**0.81**
D	V	0.73	0.07	0.70	0.07	0.16
	A	0.56	0.06	0.62	0.07	0.16
HH	V	0.66	0.08	0.68	0.05	0.47
	A	0.51	0.13	0.48	0.08	**0.58**
NA	V	0.39	0.12	0.32	0.07	0.05
	A	0.49	0.10	0.46	0.14	0.11
P	V	0.64	0.08	0.63	0.09	**0.94**
	A	0.59	0.13	0.63	0.08	0.30
Ra	V	0.61	0.06	0.69	0.07	0.06
	A	0.41	0.04	0.41	0.10	**0.84**
Re	V	0.58	0.11	0.54	0.10	0.16
	A	0.66	0.11	0.63	0.09	0.44
R	V	0.73	0.09	0.69	0.08	0.08
	A	0.57	0.08	0.54	0.05	0.22

Table 3. Valence (V) - Arousal (A) Mean and SD values for each genre (abbreviations from Table 1). Genres Z and P are chosen as representatives of low and high P - value respectively.

Significant differences in the ratings of either valence and/or arousal were found in the following genres: "Alternative (A)", "Laiko (L)", "Laiko-Pop (LP)" and "Zeibekiko

(Z)". Interestingly, all these genres belong to Eastern-influenced musical styles. It substantiates the VA rating results, since most Western listeners are especially not familiar with these types of compositions. Likewise, a greater consistency in ratings between Greek and non-Greek listeners are presented in Western-influenced genres. Mean ratings and standard deviations (SD) of Greek and non-Greek users in Eastern-influenced and Western-influenced genres are shown in the Table 3.

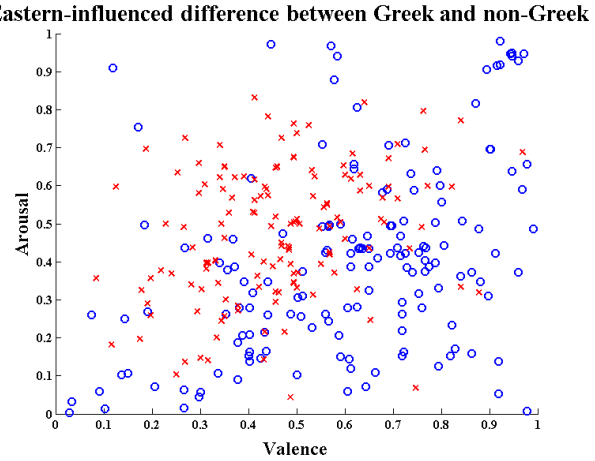


Figure 1. VA Ratings of Greek (circles) and non-Greek (crosses) for the Eastern-influenced genre "Zeibekiko"; the clusters created by both groups are separable.

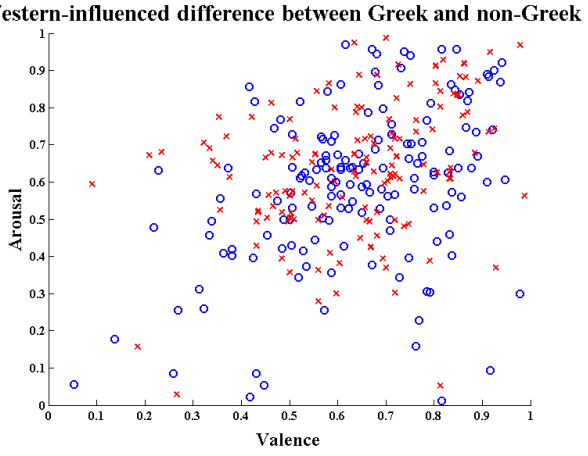


Figure 2. VA Ratings of Greek (circles) non-Greek (crosses) for the Western-influenced genre "Pop"; the clusters created by both groups overlap.

Figure 1 shows an example of the Eastern-influenced genre "Zeibekiko". The ratings of Greek listeners were closer to high valence and low arousal. However, the ratings of non-Greek listeners were closer to the upper-left part of the valence-arousal space. The cluster from Greek Listeners' ratings indicates that certain emotions are better expressed by certain genres. In addition, the ratings distribution of the Western-influenced genre "Pop" is shown in Figure 2. Although no significant differences were found between Greek and non-Greek listeners, the cluster of user ratings in genre "pop" tends to be in the right top quadrant in two-dimensional model.

6.4 Examination of the Tag Labels

Ten responses from each group (Greek and non-Greek) were collected to the optional post-experiment questionnaire. An interesting question to investigate is the conformity between their perceived emotion and the proposed tag labels in our system for every song.

When answering the question if they could find a tag in the list that was exactly or really close to what they wanted to label for the perceived mood of every song, the mean response from Greek participants is close to "Neither agree nor disagree", while non-Greek participants' mean response is close to "Disagree" (see Table 4). To some extent, it suggests that our proposed tags can represent Greek listeners but doesn't work at all for non-Greek listeners. However, the results remain tentative.

	Greek	Non-Greek
Mean	4.2	3.3
SD	1.32	1.41

Table 4. Mean and SD for level of tag agreement (scale 1-7).

7. CONCLUSIONS AND FUTURE WORK

This study investigates the cross-cultural applicability of mood categories and classification models of perceived emotion through different musical genres of Greek music. The responses by Greek people on the rating of music perception tend to agree with each other, while this does not appear to be the case as strongly with non-Greek people. A likely reason for this is differences in acculturation. Indeed significant difference can be observed in the perception of Eastern influenced musical genres between these two groups.

We have also examined whether specific metadata that is designed for Western music are applicable to Greek music. The negative results we obtained concerning tag positions derived from predominantly Western music seem to indicate that either the VA positions for the tags were less meaningful in the context of Greek music, or the tag selection mechanism was too constrained.

In future work we will compare mood classification results of perceived emotion from the Western influenced musical genres with those of Western music tracks. Additionally, an experiment with participants of specific non-Greek origination could be designed. Another alternative is to include the translation of lyrics for non-Greek participants.

In addition, expansion of the existed database is foreseen. In the new version, the genre "Rebetiko" (cf. [13] for details) will be included, and a comparison with other databases of Greek music containing folk songs from different geographical areas will be made.

Finally, it will be interesting to examine the validity of the results from western designed automatic audio feature extraction and mood estimation techniques on the less-western Greek database. This work may be enhanced using the system described in [15] for detecting similar phrases in music of the Eastern Mediterranean.

8. ACKNOWLEDGEMENTS

We thank Charalampos Tampakopoulos, Ph.D. student from University of Athens, who designed and organized the Greek database metadata and made it public accessible.

9. REFERENCES

[1] M. Bradley and P. Lang, "Affective norms for english words (anew): Instruction manual and affective ratings." *Technical Report C-2. University of Florida, Gainesville, FL.*, 2010

[2] K. Bischoff, C.S. Firan, W. Nejdl, R. Paiu, "Can all tags be used for search? " *in Proceeding of the ACM Conference on Information and Knowledge Management (CIKM)*. pp. 193–202, 2008

[3] M. Barthet, G. Fazekas, M. Sandler, "Music Emotion Recognition: From Content to Context-Based Models", *Lecture Notes in Computer Science, CMMR 2012 Post-proceedings* (in press)

[4] T. Eerola, "A comparison of the discrete and dimensional models of emotion in music" *Psychology of Music*, 39(1), 18–49 (2010)

[5] T. Eerola and J. K. Vuoskoski. "A review of music and emotion studies: Approaches, emotion models and stimuli" *Music Perception*, 30(3):307–340, 2012

[6] T. Eerola, T. Himberg, P. Toiviainen and J. Louhivuori, "Perceived complexity of Western and African folk melodies by Western and African listeners," *Psychology of Music, Vol. 34, No. 3*, pp. 337-371, 2006

[7] T. Eerola, O. Lartillot, and P. Toiviainen, "Prediction of multidimensional emotional ratings in music from audio using multivariate regression models" *in Proceeding of International Society for Music Information Retrieval (ISMIR)*, pp. 621–626, 2009

[8] T. Eerola, "Are the emotions expressed in music genre-specific? An audio-based evaluation of datasets spanning classical, film, pop and mixed genres" Journal of New Music Research, Vol. 40, No. 4, pp. 349-366, 2011

[9] Z. Fu, G. Lu, et al., "A survey of audio-based music classification and annotation." *IEEE Transactions on Multimedia*, 13(2):303 –319, april 2011

[10] J. Gower, "Generalized procrustes analysis", *Psychometrika*, vol. 40, pp. 3351, 1975

[11] T. Georgiades: "Greek music, verse and dance,"*Da Capo Press, New York*, pp. 134-141, 1973

[12] A.H. Gregory, N. Varney, "Cross-cultural comparisons in the affective response to music," *Psychology of Music*, Vol. 24, pp. 47-52, 1996

[13] G. Holst-Warhaft, "Road to Rembetika: music of a Greek sub-culture, songs of love, sorrow and hashish" *Athens, Denise Harvey*, 1989

[14] X. Hu and J.H. Lee, "A cross-cultural study of music mood perception between American and Chinese listeners," *in Proceedings of International Society for Music Information Retrieval (ISMIR)*, pp. 535-540, 2012

[15] A. Holzapfel, Y. Stylianou, "Parataxis: morphological similarity in traditional music," *in Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2010

[16] P.N. Juslin, S. Liljeström, D. Västfjäll, L.O. Lundqvist, "How does music evoke emotions? Exploring the underlying mechanisms." In: P.N. Juslin, J. Sloboda (eds.) Handbook of Music and Emotion: Theory, Research, Applications, pp. 605642. Oxford University Press (2011)

[17] G. Kapetsis, "Music information retrieval systems: An investigation on business relation impacts and user information needs of an airplay monitoring service in the Greek music industry," *Ph.D. Thesis, City University Press*, 2006

[18] M. Levy and M. Sandler, "A semantic space for music derived from social tags." *in Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2007

[19] M. Lesaffre, M. Leman, J.P. Martens, "A user oriented approach to music information retrieval" *in Proceeding of the Content-Based Retrieval Conference*, Daghstul Seminar Proceedings, 2006

[20] J.A. Lee, J.S. Downie, "Survey of music information needs, uses, and seeking behaviors: preliminary findings." *in Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2004

[21] T. Li, M. Ogihara, "Detecting emotion in music." *in Proceeding of International Society for Music Information Retrieval*, 2003

[22] D. Müllensiefen, B. Gingras, L. Stewart, and J.J. Musil. Goldsmiths Musical Sophistication Index (Gold-MSI) Technical report, 2012

[23] M. Mortillaro, B. Meuleman, R. Scherer, "Advocating a componential appraisal model to guide emotion recognition" *International Journal of Synthetic Emotions, Special Issue on Benefits and Limitations of Continuous Representations of Emotions*, 2012

[24] Oxford Music Online Dictionary, Grove Music Online, Greece, IV: Traditional music, pp. 3-4

[25] J.A. Russell, "A circumplex model of affect." *Journal of personality and social psychology* 39(6), 1161–1178, 1980

[26] P. Saari, T. Eerola, "Semantic computing of moods based on tags in social media of music.", *IEEE Transactions on Knowledge and Data Engineering*, (in press, manuscript submitted), 2013.

[27] P. Saari, M. Barthet, G. Fazekas, T. Eerola, M. B. Sandler "Semantic models of mood expressed by music: Comparison between crowd-sourced and curated editorial annotations." In *IEEE International Conference on Multimedia and Expo (ICME 2013): International Workshop on Affective Analysis in Multimedia (AAM)*.

[28] Y. Song, S. Dixon, M. Pearce, "Evaluation of musical features for emotion classification." *in Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2012.

[29] Y. Song, S. Dixon, M. Pearce, G. Fazekas, "Using tags to select stimuli in the study of music and emotion". In *The 3rd International Conference on Music & Emotion*, 2013.

[30] J. F. Thayer, Multiple indicators of affective responses to music. (Dissertation Abstracts International 47(12) , 1986

[31] G. Tzanetakis et al., "Automatic musical genre classification of audio signals," *in Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2001

[32] Y. Kim, E. M. Schmidt, et al., "Moodswings: A collaborative game for music mood label collection," in *Proc. of the International Society for Music Information Retrieval (ISMIR) Conference*, 2008.

[33] Y.H. Yang, X. Hu, "Cross-cultural music mood classification: a comparison on English and Chinese songs" *in Proceedings of International Society for Music Information Retrieval (ISMIR)*, pp. 15-24, 2012

[34] Y.H. Yang, Y.C. Lin, et al. "A regression approach to music emotion recognition." *IEEE Trans. on Audio, Speech, and Language*. 16(2), 448–457, 2008

EVALUATION ON FEATURE IMPORTANCE FOR FAVORITE SONG DETECTION

Yajie Hu, Dingding Li and Ogihara Mitsunori
Department of Computer Science
University of Miami
yajie.hu@umail.miami.edu, d.wang1@miami.edu
ogihara@cs.miami.edu

ABSTRACT

Detecting whether a song is favorite for a user is an important but also challenging task in music recommendation. One of critical steps to do this task is to select important features for the detection. This paper presents two methods to evaluate feature importance, in which we compared nine available features based on a large user log in the real world. The set of features includes song metadata, acoustic feature, and user preference used by Collaborative Filtering techniques. The evaluation methods are designed from two views: i) the correlation between the estimated scores by song similarity in respect of a feature and the scores estimated by real play count, ii) feature selection methods over a binary classification problem, i.e., "like" or "dislike". The experimental results show the user preference is the most important feature and artist similarity is of the second importance among these nine features.

1. INTRODUCTION

In the recent digital world, millions of digital songs are available online and it is difficult for users to manually search favorite songs. A music recommender system is the solution that helps users find their possible favorite songs in the song ocean, and makes a personalized journey for the user to listen these songs one by one. Essentially, the expected recommender system must be able to answer the very ask, i.e., give me my favorite songs. Automatically detecting favorite songs is therefore an important part in a music recommender system. Basically, recommender systems apply for content-based methods, Collaborative Filtering (CF) techniques, or both to detect favorite songs.

Audio content-based methods use favorite songs to predict other songs users may like [6], based on their similarity to the favorite songs. In order to recommend favorite songs to users, a music recommender system using content-based analysis depends on manual or automatic audio content description to compute the similarity among songs.

For instance, Pandora [1] employs a team of musician analysts to listen to music and give each recording its descriptions, which includes melody, harmony, instrumentation, rhythm, vocals, lyrics and etc. A weighted Euclidean distance is used to find similar songs [3].

Z. Cataltepe et al. music recommender system is based on audio similarity and users' listening history [2]. An industrial-strength music recommender system introduces the Euclidean distance of two tracks over a reduced space using Principal Component Analysis [1]. The system uses several audio features, including Mel-Frequency Cepstral Coefficients (MFCC), tempo, key mode and others.

Music was one of the first forms of content to be addressed by collaborative filtering recommender systems such as Ringo, Firey and HOMR [9, 10]. This technique finds users with similar music preferences and recommends items liked by these similar users to the target user [7]. It could be seen as a method to measure the similarity based on the user preference. Hence, the user preference is considered as one type of feature in our work. Some hybrid methods combined with collaborative filtering techniques and content-base methods are proposed to solve the cold-start problem [4, 11]. B. McFee et al. presented a method for deriving item similarity from a sample of collaborative filter data, and use the sample similarity to train a distance metric over acoustic features. The trained distance metric is used to improve the shortcoming of CF techniques, i.e., incomplete data hamper.

This paper focuses on systemic analysis regarding how important both audio content features and collaborating filtering techniques are for favorite songs detection in the real world. The two evaluation methods are described in Section 2. In Section 3, we introduce the data and present the evaluation results. We make the conclusions and discuss the future work in Section 4.

2. METHODS

We evaluate the importance of a type of feature (In the following paragraphs, "a feature" means "a type of feature" instead of a value in a feature.) by the correlation between

[1] http://www.pandora.com/about/mgp

the predicted scores using this feature and the scores by real play count. Moreover, the recommendation problem is converted to a binary classification problem. As a result, the feature selection results are used to evaluate the feature importance.

2.1 Correlation Evaluation

Some content-based recommender systems assume that a song with high similarity to the favorite songs would be liked by the user [1, 2]. Thus, the systems measure the similarity among songs and recommend the song with the highest similarity to the favorite songs. Euclidean distance and Dynamic Time Warping (DTW) algorithm are used to measure the similarity between two songs.

Euclidean distance sees a vector as a point in Euclidean space, and is given by the Pythagorean formula. This metric works for non-sequential features, like tempo.

For sequential feature, like pitch, DTW algorithm is good metric to measure the distance between two sequences, which may vary in time, since DTW algorithm is able to find the optimal alignment between two time series [8]. One time series may be non-linearly "warped" to the other one by stretching or shrinking it along its time series.

If a certain feature is important, the song similarity in terms of the feature should be highly effective to detect whether a song is favorite for a user. The highly effective detection is expected to give high relevant rating score, and the predicted rating score should correlate or anti-correlate the true one. Thus, correlation between the predicted scores and the true scores, which is estimated by play count, is therefore used to evaluate the importance of the feature. The correlation result is normalized to $[-1.0, 1.0]$, which -1.0 is in the case of a perfect negative (decreasing) linear relationship, and +1.0 means a perfect positive (increasing) linear relationship. The greater absolute value is, more important the feature is.

In a user's log, the predicted rating score is given by the following equation.

$$\hat{r}_i^u = \frac{\sum_{s_j \in \mathbf{R}^u, j \neq i} r_j^u \cdot sim_f(s_i, s_j)}{\sum_{s_j \in \mathbf{R}^u, j \neq i} sim_f(s_i, s_j)}, \tag{1}$$

where $\hat{r}_i^u$ denotes the predicted score for song s_i, and $\mathbf{R}^u$ is the set of rated songs. Rating score r_j^u is estimated by the play count of song s_j. $sim_f(s_i, s_j)$ is the similarity between song s_i and song s_j in terms of feature f.

We apply this approach to predict the rating score for each song listened by user u, and get the predicted scores $\{\hat{r}_1^u, \hat{r}_2^u, ..., \hat{r}_n^u\}$. The correlation between $\{r_1^u, r_2^u, ..., r_n^u\}$ and $\{\hat{r}_1^u, \hat{r}_2^u, ..., \hat{r}_n^u\}$ are used to evaluate the importance of feature f.

Furthermore, CF technique predicts the interest of a user in a song by collecting preferences or taste information from other users. This technique is based on the assumption that a user A is more likely to have a user B's opinion on a song x than to have the opinion on x of a user chosen randomly, if A has the same opinion as B on many songs. A user's preference or taste is represented by the opinion on the songs the user listened.

We consider a user as a document and treat the songs which the user listened as terms in the document. Then, the user could be represented by a vector of songs using TF-IDF values. The cosine distance between two vectors are used to measure the similarity between the two corresponding users.

Then, we measure the similarity between two songs considering user similarity and rating scores by the following equation.

$$\begin{aligned}
&\text{if } |\mathbf{U}_i| \leq |\mathbf{U}_j| \\
&\quad sim(s_i, s_j) = \\
&\frac{\sum_{u_m \in \mathbf{U}_i} \left| \lambda \max_{u_n \in \mathbf{U}_j} sim(u_m, u_n)^2 - \eta \left[\left(r_i^m - r_j^n \right) / V \right]^2 \right|}{|\mathbf{U}_i|}, \\
&\text{otherwise} \\
&\quad sim(s_i, s_j) = sim(s_j, s_i),
\end{aligned} \tag{2}$$

where $\mathbf{U}_i$ is the set of users who played song s_i, and u_m denotes the user m. V is the rating scale. λ and η are two regulatory factors.

This measurement allows the similarity of two songs to be high, if two users have high similar tastes, and rate the two songs at the same score. On the contrary, if two users have high similar tastes, and rate two songs at totally different scores, or the two users' tastes are different but the ratings are similar, the two songs must be different in terms of user preference.

This type of similarity is also used to predict song ratings by Equation 1, in order to evaluate the importance of user preference.

2.2 Feature Selection Methods Evaluation

Furthermore, this problem is about classifying a song to a label, i.e., "like" or "dislike", so it is a binary classification problem. We compare several feature selection methods, and select χ^2 Statistic (Chi), Information Gain (IG), Information Gain Ratio (IG Ratio) and Uncertainty to evaluate the importance of features, respectively.

χ^2 Statistic is used to investigate how much a feature depends on the category by the following equation.

$$\chi^2 = \frac{(ad - bc)^2 (a + b + c + d)}{(a + b)(c + d)(b + d)(a + c)}, \tag{3}$$

where a, b, c and d are the number of instances in different cases as shown in Table 1. If the feature is categorical, "Data Type" is the category. If the feature is numerical, "Data Type" is the data range.

IG evaluates the importance of a feature from the view of information theory. Generally speaking, the expected IG is the change in information entropy from a prior state to a posterior state that takes some information as given:

$$\mathrm{IG}(T, f) = \mathrm{H}(T) - \mathrm{H}(T|f), \tag{4}$$

Feature	Data Type 1	Data Type 2	Total
Category 1	a	b	a+b
Category 2	c	d	c+d
Total	a+c	b+d	a+b+c+d=N

Table 1. General notation for a 2 x 2 contingency table

where $\mathrm{IG}(T, f)$ denotes the IG of the feature f on the training data T. $\mathrm{H}(T)$ is the entropy of the training data, and $\mathrm{H}(T|f)$ is the average conditional entropy of T.

IG Ratio introduces Intrinsic Value (IV) to evaluate the feature importance, and it is the ratio between the information gain and the intrinsic value.

$$\mathrm{IGR}(T, f) = \mathrm{IG}(T, f)/\mathrm{IV}(T, f), \tag{5}$$

where $\mathrm{IGR}(T, f)$ is the IG Ratio of feature f and $\mathrm{IV}(T, f)$ is the intrinsic value given by:

$$\mathrm{IV}(T, f) = -\sum_{v} \frac{|\{t \in T|t_f = v\}|}{|T|} \log\left(\frac{|\{t \in T|t_f = v\}|}{|T|}\right) \tag{6}$$

Uncertainty measures the relevance of a feature by calculating the symmetrical uncertainty with respect to the class.

$$\mathrm{Uncentainty}(f) = 2 \cdot \frac{(p(c) - p(c|f))}{p(c) + p(f)} \tag{7}$$

These four methods evaluate the feature importance from different views, and Chi-square statistic and Information Gain have been proved that they surpass among feature selection methods in a text classification task [5]. We therefore apply each of them to evaluate the feature importance.

3. ANALYSIS

An ideal analysis is expected to run on a large song dataset and a big user set. Furthermore, the users should explicitly rank every song in the dataset. However, regrading to the private information protection, the exposed information is limited. This section will compare several dataset, select one dataset and evaluate several types of features.

3.1 Dataset

There are three possible and available dataset online, i.e., Taste Profile Dataset by the Echo Nest, Last.fm dataset 360K and Yahoo! Music User Ratings of Songs by Yahoo! Research.

The Taste Profile Dataset [2] is a huge collection of real world anonymous listener data in the form of Echo Nest Taste Profiles. Considering the protection of individuals private information, the data includes a shuffled hash of persistent session identifiers from a very small random selection of the musical universe and only play counts associated with Echo Nest song IDs that overlap with the Million

[2] http://labrosa.ee.columbia.edu/millionsong/tasteprofile

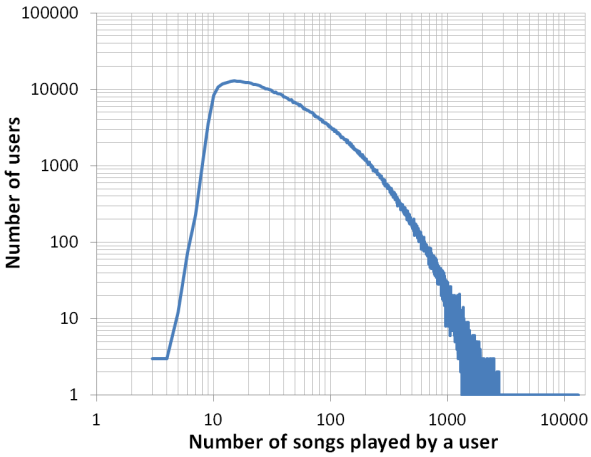


Figure 1. Distribution of users

Song Dataset (MSD) [3]. No usernames, listener details, original IDs, dates, IPs, locations or anything but random user string, Echo Nest song ID, and play count are being released [4]. The Taste Profile dataset has 1,019,318 unique users, 384,546 unique MSD songs and 48,373,586 <*user ID*, *song ID*, *play count*> triplets. The triplets don't have any time stamps and they are not in chronological order. However, the Echo Nest song ID overlaps with the MSD, which is a freely-available collection of audio features and metadata for a million contemporary popular music tracks. The MSD therefore enables researchers look into the songs by many acoustic features and metadata, it doesn't provide the audio file though.

Last.fm dataset 360K [5] collects <*user*, *artist ID*, *plays*> triplets from Last.fm API. *Plays* has two parts: *song title* and *play count*. The triplets don't have any time stamps and not in chronological order. The dataset contains 359,347 unique users and 294,015 artists and 17,559,530 <*user*, *artist ID*, *plays*> triplets. Moreover, more information about users is available, including *gender*, *age*, *country* and *date* (The "date" description is not found). The song metadata and social tags are searchable on Last.fm by *song title* and *artist ID*, but the acoustic features are not available on Last.fm.

Yahoo! Music User Ratings of Songs [6] represents a snapshot of the Yahoo! Music community's preferences for various songs. It contains over 717 million ratings of 136 thousand songs given by 1.8 million users of Yahoo! Music services. The rating triplet is <*user ID*, *song ID*, *rate*>, which *rate* is an integer from 1 to 5. All information of each song in the dataset is artist, album and genre. The users, songs, artists and albums are represented by randomly assigned numeric id's so there is no identifying information revealed. Consequently, it is impossible to accompany more information to the song.

The dataset comparison is summarized in Table 2. Al-

[3] http://labrosa.ee.columbia.edu/millionsong/

[4] http://blog.echonest.com/post/11992136676/taste-profiles-get-added-to-the-million-song-dataset

[5] http://mtg.upf.edu/node/1671

[6] http://webscope.sandbox.yahoo.com/catalog.php?datatype=r

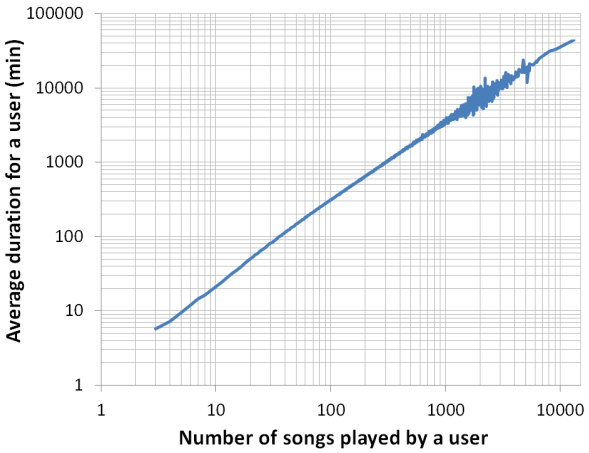


Figure 2. Distribution of estimated duration

though the Taste Profile Dataset doesn't provide explicit rating values, features and metadata provided by MSD supplies us for the probability of evaluating feature importance, and play counts could implicitly represent rating scores. Hence, we select the Taste Profile Dataset to do the experiment.

The distribution of users by the number of songs is shown in Figure 1. The user logs, of which the number of played songs are less than three, are removed by the Taste Profile dataset. The distribution mainly locates in [10, 1,000], and it has a long tail.

Users' preferences would be changed by different contexts in a long period so the overall listening duration of a user should be taken into account. The Taste Profile dataset doesn't show the actual listening duration of a song. We roughly estimate how long a user listens to the songs in the user log by Equation 8.

$$T = \sum_{i \in S}^{i} n_i \cdot t_i \cdot \left(1 - \frac{1}{n_i + 1}\right), \tag{8}$$

where n_i is the play count of song i by the user. t_i is the duration of song i, which can be obtained from MSD. The estimated duration of playing songs by users is shown in Figure 2.

3.2 Evaluation

MSD provides many types of features for a song as shown in Table 3. We select eight features from the MSD, i.e., *loudness*, *pitches*, *tempo*, *duration*, *song hotness*, *artist similarity*, *artist hotness* and *artist familiarity*, and these features cover the three categories in Table 3. Furthermore, user preference is counted in. The similarity is able to indirectly represent the performance of CF techniques in favorite song detection.

Euclidean distance is applied to measure the distance of songs for non-sequential features, while DTW method measures the distance of songs for sequential features, like pitches. DTW has a quadratic time and space complexity that limits its use to only small time series data. Thus, we apply FastDTW [8] to accelerate the similarity measurement to O(n) time.

Category	Features
Acoustic features	bars, beats, sections, segment, **loudness**, **pitches**, timbre, tatums, key, mode and **tempo**
Song metadata	sample rate, **duration**, release, **song hotness**, title and year
Artist metadata	terms, **similar artists**, **artist hotness**, **artist familiarity**, artist location, artist name and musicbrainz tags

Table 3. Features provided by MSD (Bold features are selected)

Play count is considered as an implicit rating based on the assumption that the rating is positive if the song is played many times, and vice versa. Considering the bias of the ratings, the normalized rating $r'_{i,j}$ from -1.0 to 1.0 is given by:

$$r_i^{j\prime} = r_i^j - \frac{1}{2}\left(\bar{r}_{i\cdot} + \bar{r}_{\cdot}^{j}\right), \tag{9}$$

where $\bar{r}_{i\cdot}$ is the average rating of all ratings to song s_i and $\bar{r}_{\cdot}^{j}$ denotes the average rating by user u_j.

We evaluate the feature importance by the correlation between predicted scores and normalized scores as described in Section 2.1. As all non-sequential features but user preference could represent a song to a vector, they are evaluated by feature selection methods mentioned in Section 2.2. The numeric value of a feature is assigned into categories by predefined window to apply feature selection methods. The normalized rating less than -0.1 is seen as "dislike" while the song is beloved if the rating greater than 0.1. Because the ratings between -0.1 and 0.1 is blur to classify the song to like or dislike, these ratings are ignored. The evaluation result is normalized to [0.0, 1.0] in order to compare the results among different selection methods.

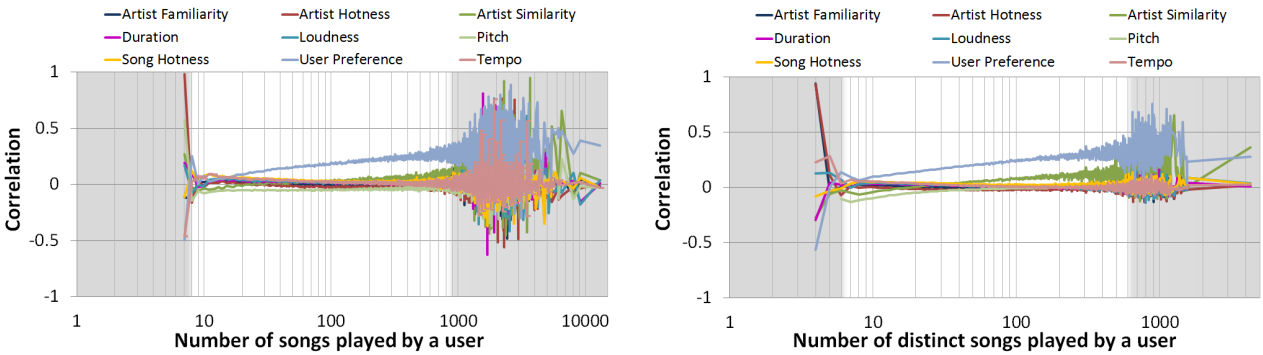
3.3 Result And Discussion

The value of each plot in Figures 3 and 4 is the mean of the results by different users who play the same number of songs. Figure 3 shows the correlation between the predicted ratings and the normalized ratings by play count in two views, including nine features. Figure 3(a) shows the distribution of the correlation by the overall played songs. In Figure 3(b), the correlation varies by the distinct songs, which means that the available information about songs increases as the distinct songs increases.

In Figure 3, the gray shadow covers the number of played songs by which there are fewer than 30 users so that the correlation results in the shadow are not statistically reliable. In the remaining part, user preference similarity is the most important feature except at the cold start, namely, CF technique is remarkable for favorite song detection. Basically, artist similarity is the secondary important feature. The other curves fluctuate around 0.0, which means

Dataset	Taste Profile Dataset	Last.fm dataset 360K	Yahoo! Music
Number of songs	**384,546**	294,015 artists	∼136,000
Number of users	1,019,318	359,347	**∼1,800,000**
Number of triplets	48,373,586	17,559,530	**∼717,000,000**
Rating type	play count	play count	**rate**
Other information	**features and metadata provided by MSD**	metadata provided by Last.fm	genre

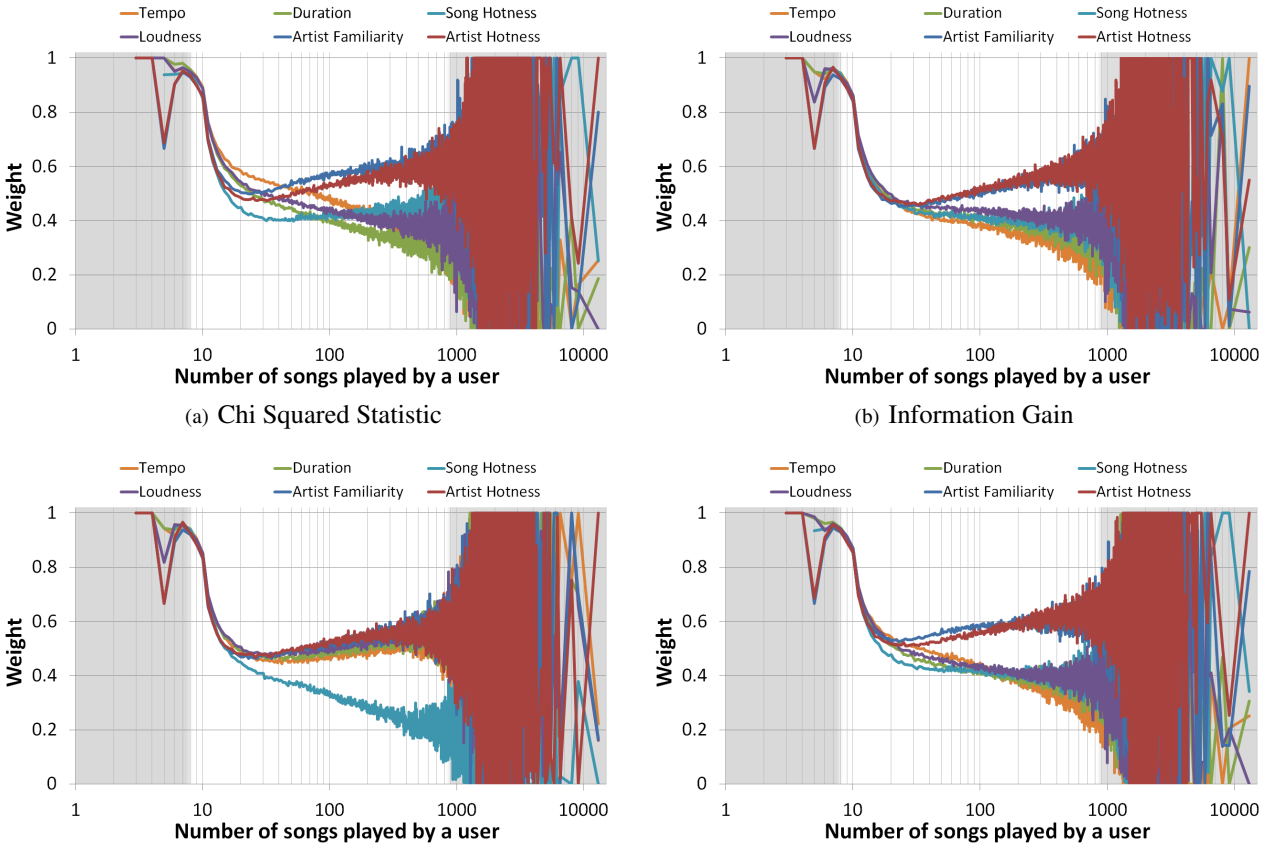
Table 2. Available dataset comparison



(a) Based on overall songs

(b) Based on distinct songs

Figure 3. Correlation between the predicted ratings and the normalized ratings



(a) Chi Squared Statistic

(b) Information Gain

(c) Information Gain Ratio

(d) Uncertainty

Figure 4. Feature selection results by different feature selection methods

the corresponding features are not critical for favorite song detection.

At the cold start period, it is frustrating that the selected features don't make a great contribution to favorite song detection. As the number of played songs increases, the number of users reduces, and the fluctuation of curves becomes vast but the main trend doesn't change significantly. Thus, in a long playing period, the importance of user preference is basically stable.

We leverage feature selection methods to measure the feature importance, and the feature selection results are normalized to [0.0, 1.0]. Zero means the feature doesn't play an important role and one means the feature is crucial. Figure 4 presents the evaluation results. The gray shadow also covers the region that the number of users are less than 30.

When the number of played songs is less than 10, most features are highly effective to distinguish "like" and "dislike". Then, the curves remarkably separate from each other. As the play counts increase, artist hotness and artist familiarity rise and other features descend more or less except Figure 4(c). Referring to Figure 2, the importance of features varies by the overall duration. In a long period, the user preferences would be changed by different contexts. As a result, the importance of tempo, duration, song hotness and loudness becomes weak while that of artist familiarity and artist hotness grows. Therefore, artist familiarity and artist hotness are more stable in a long listening period than other features.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we compare nine features by correlation and feature selection methods. Among these features, user preference and artist similarity play important roles in favorite song detection. Artist familiarity and artist hotness are stable in a long listening period. The evaluation result shows that collaborative filtering technique has high performance for favorite song detection. Song hotness is not as important as people thought.

If audio files of songs are available, we will employ a feature extraction tool to gain more features, and compare them in the future. If more information on the user log is exposed, such as time stamps and sequence information, the preference variation model is expected to be analyzed.

5. REFERENCES

[1] Pedro Cano, Markus Koppenberger, and Nicolas Wack. An industrial-strength content-based music recommendation system. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 673–673, 2005.

[2] Z. Cataltepe and B. Altinel. Music recommendation based on adaptive feature and user grouping. In *22nd international symposium on Computer and information sciences.*, pages 1–6, 2007.

[3] Oscar Celma. *Music recommendation and discovery, the long tail, long fail, and long play in the digital music space*. Springer, Berlin, 2010.

[4] Justin Donaldson. A hybrid social-acoustic recommendation system for popular music. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 187–190, 2007.

[5] George Forman. An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, 3:1289–1305, March 2003.

[6] Keiichiro Hoashi, Kazunori Matsumoto, and Naomi Inoue. Personalization of user profiles for content-based music retrieval based on relevance feedback. In *Proceedings of the eleventh ACM international conference on Multimedia*, MULTIMEDIA '03, pages 110–119, 2003.

[7] Ioannis Konstas, Vassilios Stathopoulos, and Joemon M. Jose. On social networks and collaborative recommendation. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 195–202, 2009.

[8] Stan Salvador and Philip Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.

[9] U. Shardanand. Social information filtering for music recommendation. Master's thesis, Massachusetts Institute of Technology, 1994.

[10] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 210–217, 1995.

[11] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. G. Okumo. Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. *12th International Society for Music Information Retrieval Conference*, pages 103–108, 2011.

QBT-EXTENDED: AN ANNOTATED DATASET OF MELODICALLY CONTOURED TAPPED QUERIES

Blair Kaneshiro **Hyung-Suk Kim** **Jorge Herrera**
Jieun Oh **Jonathan Berger**
CCRMA, Stanford University
{blairbo,hskim08,jorgeh,jieun5,brg}@ccrma.stanford.edu

Malcolm Slaney
Microsoft Research
CCRMA
malcolm@ieee.org

ABSTRACT

Query by tapping remains an intuitive yet underdeveloped form of content-based querying. Tapping databases suffer from small size and often lack useful annotations about users and query cues. More broadly, tapped representations of music are inherently lossy, as they lack pitch information. To address these issues, we publish QBT-Extended—an annotated dataset of over 3,300 tapped queries of pop song excerpts, along with a system for collecting them. The queries, collected from 60 users for 51 songs, contain both time stamps and pitch positions of tap events and are annotated with information about the user, such as musical training and familiarity with each excerpt. Queries were performed from both short-term and long-term memory, cued by lyrics alone or lyrics and audio. In the present paper, we characterize and evaluate the dataset and perform initial analyses, providing early insights into the added value of the novel information. While the current data were collected under controlled experimental conditions, the system is designed for large-scale, crowdsourced data collection, presenting an opportunity to expand upon this richer form of tapping data.

1. INTRODUCTION

Query by tapping (QBT) is the process of identifying a musical excerpt based upon a tapped representation. QBT is a canonical Music Information Retrieval (MIR) task and an intuitive query to perform [16], yet the literature on this topic remains small relative to other query forms such as singing and humming [9]. This task is also among the least attempted in recent years of MIREX [3]. A number of retrieval systems and databases using rhythm or tapping have been published to date (Table 1). Some cite the need for larger datasets for testing and validation [4–6]. Some lack annotations, such as musical ability of the performer, or the performer's familiarity with the excerpt being queried; such annotations could prove useful in developing improved systems [11, 12, 16]. It is also not always clear how the performer was cued to perform the query (from short-term or long-term memory; or from score, lyrics, or audio), nor how the type of cue affected performance.

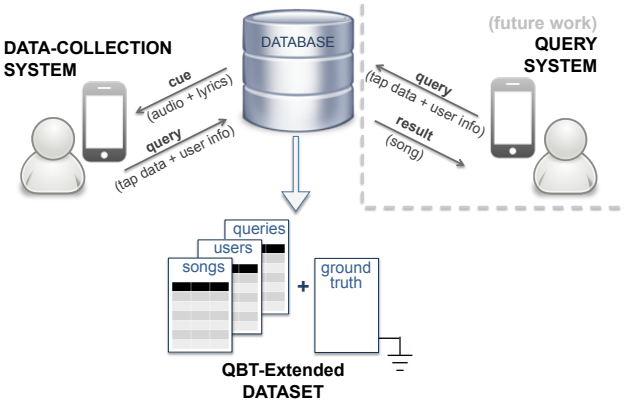


Figure 1: QBT-Extended system and dataset overview.

Study	Songs	Performers	Queries
Chen & Chen, 1998 [1]	102	NA	NA
Jang et al., 2001 [8]	NA	9	269
Eisenberg et al., 2004a [4]	9	3	27
Eisenberg et al., 2004b [5]	9	4	144/288
Peters et al., 2005 [15]	30	NA	NA
Peters et al., 2006 [16]	30	NA	518
Hanna & Robine, 2009 [6]	103	6	533
MIREX: MIR-QBT [1]	136	NA	890
MIREX: QBT_symbolic [1]	**143**	NA	410
Current study	51	**60**	**3,365** [2]

Table 1: Size of published QBT datasets to date. NA indicates that information was not available.

Another possible hurdle in QBT research is that tapping data are inherently lossy. A performer is likely replaying the pitches of a melody in his head as he taps its rhythm, but this experienced information is not captured in the output [14]. Confounding the issue further is the fact that musical excerpts can share similar or identical rhythms while being vastly different melodically (consider "Happy Birthday" and "The Star-Spangled Banner" as examples). On

[1] http://www.music-ir.org/mirex/wiki/Query_by_Tapping
[2] Numbers reported for the current dataset reflect those queries for which at least one tap event was registered. With zero-tap queries (for skipped tasks) included, the dataset comprises 3,943 queries.

a perceptual level, too, human recognition of musical excerpts is generally less successful using rhythm alone than melody alone, regardless of a listener's level of musical training [7]. Therefore, melodic information could be a useful addition to the QBT signal.

With the goal of facilitating future QBT research, we publish a dataset of annotated queries, and a system for collecting them. The queries were collected from 60 unique participants performing excerpts from a set of 51 songs. As the queries were performed on the 2D touchscreen of a mobile device, we were able to collect not only the timestamps of tap onset (touch on) and release (touch off) events, but also a rough melodic contour based upon the position of each tap on the screen. Participants were cued from either long-term memory (lyrics only) or short-term memory (lyrics and audio) for a given query task. Finally, the tapped queries are annotated with information about the performer, including musical training and the level of familiarity with each excerpt.

The current dataset was collected under controlled experimental conditions. However, the system, being mobile and open source, is easily extendable to crowdsourced data collection.

The remainder of this paper is structured as follows. We first explain how we devised the system and collected data (§2). We then describe the data (§3) and perform illustrative analyses for QBT application (§4). We conclude with a discussion of implications and next steps (§5).

2. DATA COLLECTION

2.1 Stimulus Set

We wished to maximize the number of queries that could be performed from long-term memory (cued by lyrics only). To assemble a set of songs that would be maximally familiar based upon lyrics alone, we conducted a survey to assess familiarity of lyrics excerpts from 120 top British and American pop songs from 1950–2010. Songs were chosen based upon their presence in a variety of Top 10 lists from each year, and lyrics were drawn from songs' main themes and choruses. Cued by lyrics, participants rated on a 3-point scale whether they knew the accompanying melody (Yes, Maybe/parts of it, No). We targeted the same demographic for the survey as we would for subsequent QBT data collection.

Fifty-five participants (born between 1929–1994; mean birth year 1985; 27 female) completed the survey. Each song was scored (# Yes responses - # No responses), and the 49 highest-scoring songs were retained for the QBT-Extended stimulus set. We additionally included "Happy Birthday" and "The Star-Spangled Banner" to illustrate the same-rhythm/different-melody phenomenon. The resulting 51 audio excerpts ranged in length from 9 to 28 sec (mean length 16.96 sec).

2.2 Participants

Tapping data were collected from 60 participants; participant information is summarized in Table 2. All participants were fluent in English, at least 18 years old, and reported normal hearing and no cognitive or decisional impairments. Informed consent was obtained from each participant at the start of his session. Both the survey and tapping study were approved by the local Institutional Review Board (IRB).

Data	Value Stored	Range Collected
age	integer	18-64 (mean = 30.8)
gender	male/female	33/27
native language	text	English = 53
music listening	0 (never) to 5 (all the time)	mean = 3.91
instrument training	0 (none) to 5 (professional)	mean = 2.53
theory training	0 (none) to 5 (professional)	mean = 2.08
handedness	left/right/both	3/55/2
tone-deafness	yes/no/don't know	1/58/1
arrhythmic	yes/no/don't know	0/59/1
specific training	time and instrument	varies

Table 2: Participant information collected from in-app questionnaire. Values for *music listening*, *instrument training*, and *theory training* are continuous in the given range. Multiple answers (instruments) per user were allowed for *specific training*.

2.3 System

The system comprises a front end for data collection and a back end for data storage and processing/analysis. An iOS application was developed for the front end so that participants could leverage the 2D touchscreen for queries, tapping higher on the screen for higher pitches, and lower for lower pitches. A screenshot of the tap screen is shown in Figure 2. Our internal tests show a tap-to-timestamp latency of 40 ms on average, with standard deviation of 10 ms. Because the start time of each recorded query is set to the onset of the first tap event, mean latency is not a factor. The front-end application was also used to obtain informed consent, and for the participant questionnaire (§2.4). The mobile implementation facilitated data collection and provided an ecologically valid apparatus that would extend easily to real-life use of a QBT system.

Figure 2: The data-collection application's tapping interface. Users can tap anywhere in the shaded gray area of the screen, using the vertical position of the tap to denote pitch height.

The back-end application was written in Ruby-on-Rails to receive and store the collected data. The data are stored in an SQLite3 database. The back end also stores the audio and lyrics files that are fetched by the front end when a new experiment is instantiated.

2.4 Data Collection Procedure

All data were collected using 4th Generation iPod Touch devices and Sony MDR-V6 headphones. Participants started the session by giving informed consent and filling out the questionnaire (Table 2). Following this, the participant completed 3 practice trials in order to learn how to use the application and perform queries, with the experimenter on hand to provide instruction and clarification. Once the participant was comfortable with the interface, he performed up to 51 trials in random order for the remainder of the 45-minute session. [3] Participants were given a $10 gift card at the end of the session. No authors contributed to the dataset.

A single trial is described as follows:

1. A lyrics excerpt, along with the song title, performer, and year, is presented on screen.
2. Long-term memory task: The participant is asked to tap the melody accompanying the lyrics if it is familiar, using the vertical axis to denote approximate pitch positions. If the user cannot recall the melody, he skips this step.
3. The participant is asked "How familiar was the song presented?" The answer is encoded as a continuous value from 0 to 5.
4. The participant listens to the audio accompanying the excerpt. The audio plays only once, and must be heard in its entirety. The lyrics and metadata are shown on screen while the audio plays.
5. Short-term memory task: The participant is taken back to the lyrics/metadata screen (described in Step 1) and taps the melody (regardless of whether he was able to do so from long-term memory).
6. The participant is asked "Did hearing the music help? Tap on the answer that fits best." The answer options, and distribution of responses, can be found in Table 3.

3. DATASET

3.1 Ground Truth

A ground truth was created for each excerpt by a single performer with 14 years of piano training. MIDI keyboard renditions of the sung melodies in the excerpts were converted to comma-separated value (CSV) files with MIDI note number, note-on, and note-off times. Both MIDI and CSV formats are included in the dataset.

[3] Some participants requested to perform queries for all of the songs, taking more than 45 minutes to complete the set.

3.2 Statistics of the Collected Data

A total of 3,365 queries were collected—1,412 tapped from long-term memory (cued by lyrics only) and 1,953 from short-term memory (cued by lyrics and audio). For each song, an average of 27.69 long-term memory queries (min = 16, max = 37) and 38.29 short-term memory queries (min = 31, max = 47) were collected. Each participant performed on average 23.53 queries from long-term memory (min = 0, max = 51) and 32.55 queries from short-term memory (min = 20, max = 51).

3.3 Structure

The database contains 3 tables: *songs*, *users* (participants), and *tasks*. The fields of each table are summarized in Table 4. The *songs* table contains information about the songs in the stimulus set. Due to possible copyright issues, the dataset does not include lyrics or audio, and only specifies the start and end times within the original song, as well as the song part from which the lyrics are derived (main theme, chorus, or other). The *users* table contains the participant information summarized in Table 2. The *tasks* table contains the information for each query including `user_id` and `song_title`, which can be used to identify the participant and song of a given query.

songs (7 fields)		
filename	song_title	artist
year	start_time	end_time
song_section		
users (11 fields)		
age	gender	listening_habits
instrument_training	theory_training	handedness
tone_deaf	arrhythmic	user_id
native_language	specific_training	
tasks (16 fields)		
version_number	song_title	user_id
session_id	experimenter_id	task_order
device_type	song_familiarity	with_music
audio_helpful	tap_data	tap_off_data
tap_x_data	tap_y_data	tap_off_x_data
tap_off_y_data		

Table 4: Database table fields. The `song_title` field connects the *songs* and *tasks* tables, while the `user_id` field connects the *users* and *tasks* tables.

The dataset is available as 1) an SQLite3 db-file; 2) comma-separated value (CSV) files; and 3) space-separated .onset files compatible with previous QBT datasets. In addition to .onset files for a given task, we provide in separate files the *x* and *y* screen coordinates and release times corresponding to each onset.

The dataset is distributed using the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 license, and is available for download at the following URL:

`https://ccrma.stanford.edu/groups/qbtextended`

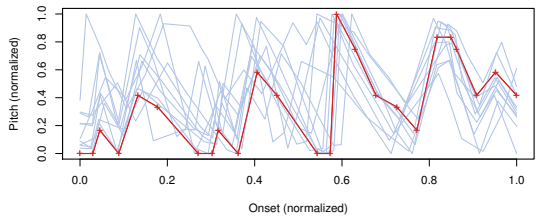
3.4 Observations of the Data

We present two example visualizations of the tapped queries. First, a rough melodic contour can be recon-

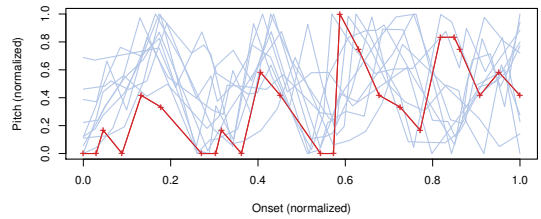
Did hearing the music help?	With long-term	Without long-term
Yes—it helped me remember more details of the song	70.0 %	30.6 %
Yes—I thought the lyrics were from a different song, but now I know which song it is	0.9 %	4.0 %
Yes—I had no idea of the song from just the lyrics, but listening made me recognize it	2.3 %	26.1 %
No—I already knew the song really well	24.7 %	1.5 %
No—this song is totally unfamiliar, so hearing it once didn't help	0.4 %	10.1 %
Yes—I didn't know the song at all, but I could tap it out after hearing it	1.6 %	27.7 %

Table 3: Distribution of answers to the question asked at the end of the short-term memory task (§2.4). The second column shows the distribution for short-term memory queries where the participant also did the long-term memory task; the third column presents the distribution for cases where the participant performed the query from short-term, but not long-term, memory.

structed by plotting each tap position as a function of its onset time. Figure 3 shows the short-term memory queries for "Happy Birthday" from participants in the top and bottom quartiles of musical instrument training for the song. The ground truth is overlaid in red. Both the query lengths (x-axis) and tap positions (y-axis) have been normalized to the length of each query and total vertical range of the screen used, respectively.



(a) Highest quartile of instrument training [3.5–5.0]



(b) Lowest quartile of instrument training [0.0–2.0)

Figure 3: Tapped pitch contours of "Happy Birthday" queries from short-term memory (blue) with ground truth (red). Pitch contours and timestamps have been normalized to the vertical range of the screen used and the total length of the query, respectively. Variance among queries appears to be lower for the highly trained participants, especially in the second half of the query.

In contrast, Figure 4 focuses solely on the temporal dimension of the queries. More variability in the temporal patterns is evident when absolute tap times are presented (Figure 4a), but the phrase structure becomes easier to discern across the set when each query is normalized by its length (Figure 4b).

We include some anecdotal observations from the experiment sessions:

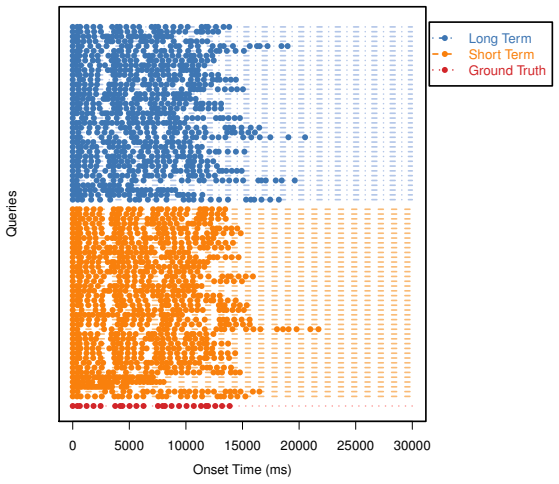
1. *The tapping queries were fun to do.* This observation is supported by some participants requesting to finish the full set of songs; but even participants who did not finish the full set reported that they had fun.
2. *The pitch positions are not exact representations of the melody being performed.* For example, repeated pitches in a melody were likely not tapped at the exact same position on the screen; similarly, recurring taps in a given position do not necessarily reflect the same pitch. In addition, the range of the melody relative to the range of the screen may have changed over the course of a query, as participants could encounter notes outside of the range they had accounted for initially.
3. *Participants with less musical training reported difficulty with the pitch dimension.* Some participants reporting this problem preferred to tap in the same general area of the screen, while others tapped monotonically up the screen for each line of lyrics. As evidenced in Figure 3, degradation of the pitch contour is observable among participants with lower reported levels of instrument training.
4. *No participants reported trouble with the rhythm dimension.* More analysis is needed to confirm this observation.
5. *Many participants reported often not knowing a melody from the lyrics alone, but recognizing it once the audio started playing.* Evidence of this can be seen in Table 3, as 26.1% of participants who could not do the long-term memory task actually did know the song once they heard the audio.
6. *Some participants reported that their instrument experience (e.g., guitar, drums) distracted them from the vocal line in the audio, and that they wanted to tap their instrument's part instead.* More analysis is needed to confirm this observation.

4. PRELIMINARY ANALYSIS

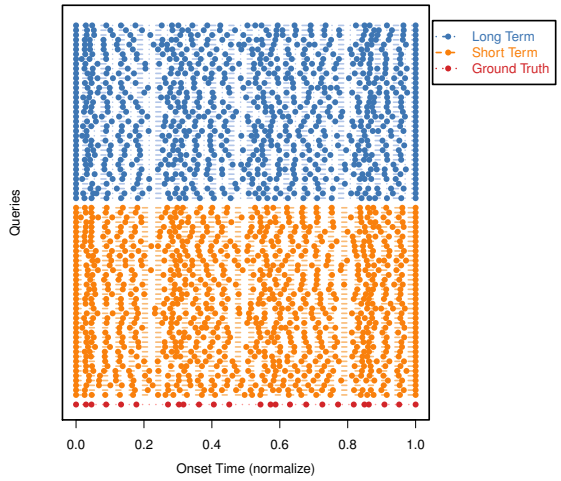
Although the main focus of this paper is to introduce QBT-Extended, in this section we perform some preliminary analyses of the dataset to build a rudimentary QBT system. We acknowledge that these analyses are very basic, and they are intended primarily for illustrative purposes and to validate the data.

We validated the temporal dimension of the dataset using Rhythmic Contour String encoding [16] for both the ground truth and the queries.[4] All queries, as well as

[4] This encoding normalizes onset-to-onset durations relative to mean inter-onset duration, then converts each inter-onset duration to S ("same"), U ("up"), or D ("down") using a threshold of "sameness" to define S. We used a threshold of 0.2.



(a) Not normalized



(b) Normalized

Figure 4: Temporal dimension of all queries collected for "Happy Birthday".

the ground truth, are represented as separate strings. We then used edit distance (or Levenshtein distance) between strings, computed by the approximate matching procedure [16], to rank the distance between a query and each of the ground-truth strings. The distances were computed over the full set of 51 candidate excerpts, and results were ranked by increasing order of distance. Table 5 contains the classification accuracy for the entire dataset against the ground-truth data. A query result is considered correct if the actual song was among the top N sorted matches.

We adapted the Rhythmic Contour String method to analyze the novel pitch dimension. Because the pitch information expressed in the queries is not exact (§3.4), we encoded pitch contours as separate strings based upon note-to-note variance relative to the overall position range of the query. Differences in position were computed between successive tap events, and the set of differences for a given query were normalized between 1 and -1. Following that, the same 0.2 thresholding was used. We then applied the approximate matching procedure to compare distances between a query and each ground truth, and ranked the results.

Each query therefore comprises two strings; one for rhythm and one for melody. As a preliminary attempt to make use of both representations, we computed the rhythmic and melodic distances separately and averaged them, giving equal weighting to each.

4.1 Assessing Performance with Added Pitch Dimension

The accuracy of our simple system using each dimension alone, and rhythm and melody combined, is shown in Table 5. These results were computed using the entire set of queries, and no filtering was applied based on user ability or familiarity with each song excerpt. Rhythm alone outperforms melody alone for all three ranking ranges.

	Accuracy (%)		
Condition	Top 1	Top 5	Top 10
Rhythmic contour	51.92	70.82	78.46
Melodic contour	37.68	54.27	64.67
Both contours	53.67	70.70	78.28
	Significance analysis		
$\chi^2(1, N = 3,365)$	4.69	0.017	0.060
p-value	0.030	0.90	0.81

Table 5: Accuracy of the simple QBT classification system using all 3,365 tapped queries (51-class problem). χ^2 and p-values, comparing classification using rhythmic contour alone versus rhythmic and melodic contours combined, were computed using McNemar's test.

To quantitatively assess the effect of adding the pitch dimension, we used McNemar's test [13], an established method of comparing two classifications of a single dataset [2]. We compared performance when the classifier used only rhythmic contour, versus rhythmic and melodic contours together. The tests show that adding melodic contour significantly improved accuracy for the Top 1 case, but did not significantly affect classifier performance in the Top 5 and Top 10 cases.

A specific case in which melodic information should boost accuracy is the 2-class problem of "Happy Birthday" versus "The Star-Spangled Banner", which share similar rhythms. As shown in Table 6, using rhythmic and melodic contour together significantly improved classifier accuracy over using rhythmic contour alone.

Condition	Accuracy (%)
Rhythmic contour	75.54
Melodic contour	72.66
Both contours	89.93
	Significance analysis
$\chi^2(1, N = 139)$	10.03
p-value	0.0015

Table 6: Comparison of accuracies in the 2-class problem classifying "Happy Birthday" and "The Star-Spangled Banner" queries. McNemar's test measures the significance of the change in classifier performance when both rhythmic and melodic contour are used, versus rhythmic contour alone.

5. DISCUSSION

The QBT-Extended dataset and system presents new possibilities for QBT research. By appropriately applying the new pitch position information, and by understanding how user background and memory cue affect performance, more effective QBT systems may be implementable. In addition, the touchscreen-based interface for data collection may prove useful for users who are not comfortable singing or humming, or who wish to query in situations where use of the microphone for acoustic input is not ideal (for instance, in a library or a noisy bar).

We acknowledge limitations of the current dataset. First, our choice of device for data collection imposed constraints upon the range of vertical space available, and users may have run out of room or needed to tap over the lyrics for some queries. In addition, the act of expressing the pitch dimension was confusing for some users, especially those who did not have musical training or could not read music. Therefore, it may be the case that having to focus on both timing and pitch degraded the quality of output along both dimensions. As we noticed that some sliders were not moved, nor instrument training fields filled out in the questionnaire, some users may not have entered their information completely. Finally, we acknowledge the demographic skew of the current participant population for this dataset, given the community that we targeted for the study [10].

5.1 Future Work

Many opportunities for future work are present. First, more analysis can be done to evaluate the usefulness of both the added pitch dimension and the annotations accompanying each query. For example, it may be useful to weight the temporal versus pitch dimensions of a query based upon users' musical expertise, experience, and familiarity with the specific excerpt. Alternate representations of queries, such as the normalized signals shown in Figure 3, could also provide feasible feature vectors for classification. Beyond the domain of query, the dataset is potentially useful for research on musical memory, expertise, and other aspects of music cognition.

Because the data-collection system is open source, a natural extension of the current implementation would be to port it to other platforms (e.g., Android, web [16]) and devices (e.g., tablets). The system is also well positioned for crowdsourced data collection, and the current dataset could then serve as a control to validate the quality of data collected via crowdsourcing.

6. REFERENCES

[1] James CC Chen and Arbee LP Chen. Query by rhythm: An approach for song retrieval in music databases. In *Research Issues In Data Engineering, 1998. Proceedings of Eighth International Workshop on Continuous-Media Databases and Applications*, pages 139–146. IEEE, 1998.

[2] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, December 2006.

[3] J Stephen Downie, Andreas F Ehmann, Mert Bay, and M Cameron Jones. The music information retrieval evaluation exchange: Some observations and insights. In *Advances in music information retrieval*, pages 93–115. Springer, 2010.

[4] Gunnar Eisenberg, Jan-Mark Batke, and Thomas Sikora. BeatBank – an MPEG-7 compliant query by tapping system. In *Audio Engineering Society Convention 116*, 2004.

[5] Gunnar Eisenberg, Jan-Mark Batke, and Thomas Sikora. Efficiently computable similarity measures for query by tapping systems. In *Proceedings of the Seventh International Conference on Digital Audio Effects (DAFx'04), Naples, Italy, October*, pages 189–192, 2004.

[6] Pierre Hanna and Matthias Robine. Query by tapping system based on alignment algorithm. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1881–1884. IEEE, 2009.

[7] Sylvie Hébert and Isabelle Peretz. Recognition of music in long-term memory: Are melodic and temporal patterns equal partners? *Memory & cognition*, 25(4):518–533, 1997.

[8] Jyh-Shing Roger Jang, Hong-Ru Lee, and Chia-Hui Yeh. Query by tapping: A new paradigm for content-based music retrieval from acoustic input. In *Advances in Multimedia Information ProcessingPCM 2001*, pages 590–597. Springer, 2001.

[9] Alexios Kotsifakos, Panagiotis Papapetrou, Jaakko Hollmén, Dimitrios Gunopulos, and Vassilis Athitsos. A survey of query-by-humming similarity methods. In *Proceedings of the 5th International Conference on PErvasive Technologies Related to Assistive Environments*, page 5. ACM, 2012.

[10] Jin Ha Lee and Sally Jo Cunningham. The impact (or non-impact) of user studies in music information retrieval. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, pages 391–396, 2012.

[11] Micheline Lesaffre, Koen Tanghe, Gaëtan Martens, Dirk Moelants, Marc Leman, Bernard De Baets, Hans De Meyer, and Jean-Pierre Martens. The MAMI query-by-voice experiment: Collecting and annotating vocal queries for music information retrieval. In *Proceedings of the International Society for Music Information Retrieval Conference*, 2003.

[12] Mark Levy. Improving perceptual tempo estimation with crowd-sourced annotations. *Proceedings of the International Society for Music Information Retrieval Conference*, pages 317–322, 2011.

[13] Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, June 1947.

[14] Elizabeth Louise Newton. *The rocky road from actions to intentions*. PhD thesis, Stanford University, 1990.

[15] Geoffrey Peters, Caroline Anthony, and Michael Schwartz. Song search and retrieval by tapping. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 1696. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[16] Geoffrey Peters, Diana Cukierman, Caroline Anthony, and Michael Schwartz. Online music search by tapping. In *Ambient Intelligence in Everyday Life*, pages 178–197. Springer, 2006.

AUDIO CHORD RECOGNITION WITH RECURRENT NEURAL NETWORKS

Nicolas Boulanger-Lewandowski, Yoshua Bengio and Pascal Vincent
Dept. IRO, Université de Montréal
Montréal, Québec, Canada H3C 3J7
{boulanni, bengioy, vincentp}@iro.umontreal.ca

ABSTRACT

In this paper, we present an audio chord recognition system based on a recurrent neural network. The audio features are obtained from a deep neural network optimized with a combination of chromagram targets and chord information, and aggregated over different time scales. Contrarily to other existing approaches, our system incorporates acoustic and musicological models under a single training objective. We devise an efficient algorithm to search for the global mode of the output distribution while taking long-term dependencies into account. The resulting method is competitive with state-of-the-art approaches on the MIREX dataset in the major/minor prediction task.

1. INTRODUCTION

Automatic recognition of chords from audio music is an active area of research in music information retrieval [16, 21]. Existing approaches are commonly based on two fundamental modules: (1) an *acoustic* model that focuses on the discriminative aspect of the audio signal, and (2) a musicological, or *language* model that attempts to describe the temporal dependencies associated with the sequence of chord labels, e.g. harmonic progression and temporal continuity. In this paper, we design a chord recognition system that combines the acoustic and language models under a unified training objective using the sequence transduction framework [7, 12]. More precisely, we introduce a probabilistic model based on a recurrent neural network that is able to learn realistic output distributions given the input, that can be trained automatically from examples of audio sequences and time-aligned chord labels.

Following recent advances in training deep neural networks [1] and its successful application to chord recognition [19], music annotation and auto-tagging [15], polyphonic music transcription [24] and speech recognition [17], we will exploit the power of deep architectures to extract features from the audio signals. This pre-processing step will ensure we feed the most discriminative features possible to our transduction network. A popular enhancement that we also employ consists in the use of multiscale aggregated features to describe context information [4, 10, 14]. We also exploit prior information [13] in the form of pitch class targets derived from chord labels, known to be a useful intermediate representation for chord recognition (e.g. [9]).

Recurrent neural networks (RNN) [26] are powerful dynamical systems that incorporate an internal memory, or *hidden state*, represented by a self-connected layer of neurons. This property makes them well suited to model temporal sequences, such as frames in a magnitude spectrogram or chord labels in a harmonic progression, by being trained to predict the output at the next time step given the previous ones. RNNs are completely general in that in principle they can describe arbitrarily complex long-term temporal dependencies, which made them very successful in music applications [5–7, 11, 23]. While RNN-based musical language models significantly surpass popular alternatives like hidden Markov models (HMM) [6] and offer a principled way to combine the acoustic and language models [7], existing inference procedures are time-consuming and suffer from various problems that make it difficult to obtain accurate predictions. In this paper, we propose an inference method similar to Viterbi decoding that preserves the predictive power of the probabilistic model, and that is both more efficient and accurate than alternatives.

The remainder of this paper is organized as follows. In Section 2, we present our feature extraction pipeline based on deep learning. In Sections 3 and 4 we introduce the recurrent neural network model and the proposed inference procedure. We describe our experiments and evaluate our method in Section 5.

2. LEARNING DEEP AUDIO FEATURES

2.1 Overview

The overall feature extraction pipeline is depicted in Figure 1. The magnitude spectrogram is first computed by the short-term Fourier transform using a 500 ms sliding Blackman window truncated at 4 kHz with hop size 64 ms and zero-padded to produce a high-resolution feature vector of length 1400 at each time step, L^2 normalized and square root compressed to reduce the dynamic range. Due to the following pre-processing steps, we found that a mel scale conversion was unnecessary at this point. We apply

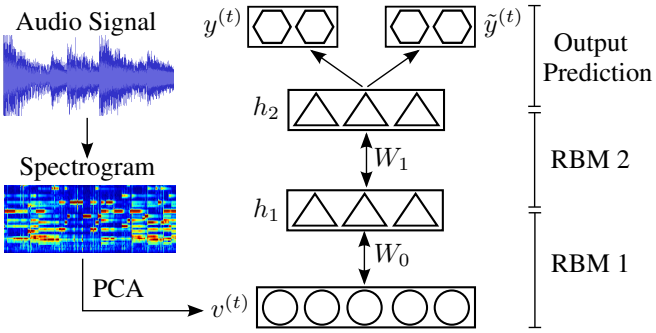


Figure 1. Pre-processing pipeline to learn deep audio features with intermediate targets $z^{(t)}, \tilde{z}^{(t)}$. Single arrows represent a deterministic function, double-ended arrows represent the hidden-visible connections of an RBM.

PCA whitening to retain 99% of the training data variance, yielding roughly 30–35% dimensionality reduction. The resulting whitened vectors $v^{(t)}$ (one at each time step) are used as input to our DBN.

2.2 Deep belief networks

The idea of deep learning is to automatically construct increasingly complex abstractions based on lower-level concepts. For example, predicting a chord label from an audio excerpt might understandably prerequire estimating active pitches, which in turn might depend on detecting peaks in the spectrogram. This hierarchy of factors is not unique to music but also appears in vision, natural language and other domains [1].

Due to the highly non-linear functions involved, deep networks are difficult to train directly by stochastic gradient descent. A successful strategy to reduce these difficulties consists in pre-training each layer successively in an unsupervised way to model the previous layer expectation. In this work, we use restricted Boltzmann machines (RBM) [27] to model the joint distribution of the previous layer's units in a deep belief network (DBN) [18] (not to be confused with a dynamic Bayesian network).

The observed vector $v^{(t)} \equiv h_0$ (input at time step t) is transformed into the hidden vector h_1, which is then fixed to obtain the hidden vector h_2, and so on in a greedy way. Layers compute their representation as:

$$h_{l+1} = \sigma(W_l h_l + b_l) \tag{1}$$

for layer l, $0 \leq l < D$ where D is the depth of the network, $\sigma(x) \equiv (1+e^{-x})^{-1}$ is the element-wise logistic sigmoid function and W_l, b_l are respectively the weight and bias parameters for layer l. The whole network is finally fine-tuned with respect to a supervised criterion such as the cross-entropy cost:

$$L(v^{(t)}, z^{(t)}) = -\sum_{j=1}^{N} z_j^{(t)} \log y_j^{(t)} + (1-z_j^{(t)}) \log(1-y_j^{(t)}) \tag{2}$$

where $y^{(t)} \equiv h_D$ is the prediction obtained at the topmost layer and $z^{(t)} \in \{0,1\}^N$ is a binary vector serving as a target at time step t. Note that in the general multi-label framework, the target $z^{(t)}$ can have multiple active elements at a given time step.

2.3 Exploiting prior information

During fine-tuning, it is possible to utilize prior information to guide optimization of the network by providing different variables, or *intermediate targets*, to be predicted at different stages of training [13]. Intermediate targets are lower-level factors that the network should learn first in order to succeed at more complex tasks. For example, chord recognition is much easier if the active pitch classes, or *chromagram targets*, are known. Note that it is straightforward to transform chord labels $z^{(t)}$ into chromagram targets $\tilde{z}^{(t)}$ and vice versa using music theory. Our strategy to encourage the network to learn this prior information is to conduct fine-tuning with respect to $\tilde{z}^{(t)}$ in a first phase then with respect to $z^{(t)}$ in a second phase, with all parameters W_l, b_l except for the last layer preserved between phases.

While a DBN trained with target $z^{(t)}$ can readily predict chord labels, we will rather use the last hidden layer $h_{D-1}^{(t)}$ as input $x^{(t)}$ to our RNN in order to take temporal information into account.

2.4 Context

We can further help the DBN to utilize temporal information by directly supplementing it with tap delays and context information. The retained strategy is to provide the network with aggregated features $\bar{x}, \tilde{x}$ [4] computed over windows of varying sizes L [14] and offsets τ relative to the current time step t:

$$\bar{x}^{(t)} = \left\{ \sum_{\Delta t=-\lfloor L/2 \rfloor}^{\lfloor (L-1)/2 \rfloor} x^{(t-\tau+\Delta t)}, \forall (L,\tau) \right\} \tag{3}$$

$$\tilde{x}^{(t)} = \left\{ \sum_{\Delta t=-\lfloor L/2 \rfloor}^{\lfloor (L-1)/2 \rfloor} (x^{(t-\tau+\Delta t)} - \bar{x}_{L,\tau}^{(t)})^2, \forall (L,\tau) \right\} \tag{4}$$

for mean and variance pooling, where the sums are taken element-wise and the resulting vectors concatenated, and L, τ are taken from a predefined list that optionally contains the original input ($L = 1, \tau = 0$). This strategy is applicable to frame-level classifiers such as the last layer of a DBN, and will enable fair comparisons with temporal models.

3. RECURRENT NEURAL NETWORKS

3.1 Definition

The RNN formally defines the conditional distribution of the output z given the input x:

$$P(z|x) = \prod_{t=1}^{T} P(z^{(t)}|\mathcal{A}^{(t)}) \tag{5}$$

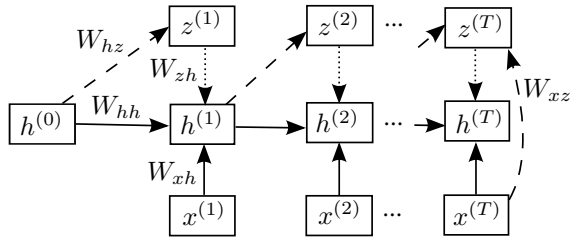


Figure 2. Graphical structure of the RNN. Single arrows represent a deterministic function, dotted arrows represent optional connections for temporal smoothing, dashed arrows represent a prediction. The $x \to z$ connections have been omitted for clarity at each time step except the last.

where $\mathcal{A}^{(t)} \equiv \{x, z^{(\tau)} | \tau < t\}$ is the sequence history at time t, $x \equiv \{x^{(t)}\}$ and $z \equiv \{z^{(t)} \in C\}$ are respectively the input and output sequences (both are given during supervised training), C is the dictionary of possible chord labels ($|C| = N$), and $P(z^{(t)}|\mathcal{A}^{(t)})$ is the conditional probability of observing $z^{(t)}$ according to the model, defined below in equation (9).

A single-layer RNN with hidden units $h^{(t)}$ is defined by its recurrence relation:

$$h^{(t)} = \sigma(W_{zh} z^{(t)} + W_{hh} h^{(t-1)} + W_{xh} x^{(t)} + b_h) \quad (6)$$

where the indices of weight matrices and bias vectors have obvious meanings. Its graphical structure is illustrated in Figure 2.

The prediction $y^{(t)}$ is obtained from the hidden units at the previous time step $h^{(t-1)}$ and the current observation $x^{(t)}$:

$$y^{(t)} = s(W_{hz} h^{(t-1)} + W_{xz} x^{(t)} + b_z) \quad (7)$$

where $s(a)$ is the softmax function of an activation vector a:

$$(s(a))_j \equiv \frac{\exp(a_j)}{\sum_{j'=1}^{N} \exp(a_{j'})}, \quad (8)$$

and should be as close as possible to the target vector $z^{(t)}$. In recognition problems with several classes, such as chord recognition, the target is a one-hot vector and the likelihood of an observation is given by the dot product:

$$P(z^{(t)}|\mathcal{A}^{(t)}) = z^{(t)} \cdot y^{(t)}. \quad (9)$$

3.2 Training

The RNN model can be trained by maximum likelihood with the following cost (replacing eq. 2):

$$L(x, z) = -\sum_{t=1}^{T} \log(z^{(t)} \cdot y^{(t)}) \quad (10)$$

where the gradient with respect to the model parameters is obtained by backpropagation through time (BPTT) [26].

While in principle a properly trained RNN can describe arbitrarily complex temporal dependencies at multiple time scales, in practice gradient-based training suffers from various pathologies [3]. Several strategies can be used to help reduce these difficulties including gradient clipping, leaky integration, sparsity and Nesterov momentum [2].

It may seem strange that the $z^{(t)}$ variable acts both as a target to the prediction $y^{(t)}$ and as an input to the RNN. How will these labels be obtained to drive the network during testing? In the transduction framework [7, 12], the objective is to infer the sequence $\{z^{(t)*}\}$ with maximal probability given the input. The search for a global optimum is a difficult problem addressed in the next section. Note that the connections $z \to h$ are responsible for temporal smoothing by forcing the predictions $y^{(t)}$ to be consistent with the previous decisions $\{z^{(\tau)} | \tau < t\}$. The special case $W_{zh} = 0$ gives rise to a recognition network without temporal smoothing.

A potential difficulty with this training scenario stems from the fact that since z is known during training, the model might (understandably) assign more weight to the symbolic information than the acoustic information. This form of *teacher forcing* during training could have dangerous consequences at test time, where the model is autonomous and may not be able to recover from past mistakes. The extent of this condition can be partly controlled by adding the regularization terms $\alpha(|W_{xz}|^2 + |W_{xh}|^2) + \beta(|W_{hz}|^2 + |W_{hh}|^2)$ to the objective function, where the hyperparameters α and β are weighting coefficients. It is trivial to revise the stochastic gradient descent updates to take those penalties into account.

4. INFERENCE

A distinctive feature of our architecture are the (optional) connections $z \to h$ that implicitly tie $z^{(t)}$ to its history $\mathcal{A}^{(t)}$ and encourage coherence between successive output frames, and temporal smoothing in particular. At test time, predicting one time step $z^{(t)}$ requires the knowledge of the previous decisions on $z^{(\tau)}$ (for $\tau < t$) which are yet uncertain (not chosen optimally), and proceeding in a greedy chronological manner does not necessarily yield configurations that maximize the likelihood of the complete sequence. We rather favor a global search approach analogous to the Viterbi algorithm for discrete-state HMMs.

4.1 Viterbi decoding

The simplest form of temporal smoothing is to use an HMM on top of a frame-level classifier. The HMM is a directed graphical model defined by its conditional independence relations:

$$P(x^{(t)}|\{x^{(\tau)}, \tau \neq t\}, z) = P(x^{(t)}|z^{(t)}) \quad (11)$$
$$P(z^{(t)}|\{z^{(\tau)}, \tau < t\}) = P(z^{(t)}|z^{(t-1)}) \quad (12)$$

where the emission probability can be formulated using Bayes' rule [17]:

$$P(x^{(t)}|z^{(t)}) \propto \frac{P(z^{(t)}|x^{(t)})}{P(z^{(t)})} \quad (13)$$

where $P(z^{(t)}|x^{(t)})$ is the output of the classifier and constant terms given x have been removed. Since the resulting

joint distribution

$$P(z^{(t)}, x^{(t)} | \{z^{(\tau)}, \tau < t\}) \propto \frac{P(z^{(t)}|x^{(t)})}{P(z^{(t)})} P(z^{(t)}|z^{(t-1)}) \tag{14}$$

depends only on $z^{(t-1)}$, it is easy to derive a recurrence relation to optimize z^* by dynamic programming, giving rise to the well-known Viterbi algorithm.

4.2 Beam search

An established algorithm for sequence transduction with RNNs is beam search (Algorithm 1) [7, 12]. Beam search is a breadth-first tree search where only the w most promising paths (or nodes) at depth t are kept for future examination. In our case, a node at depth t corresponds to a subsequence of length t, and all descendants of that node are assumed to share the same sequence history $\mathcal{A}^{(t+1)}$; consequently, only $z^{(t)}$ is allowed to change among siblings. This structure facilitates identifying the most promising paths by their cumulative log-likelihood. Note that $w = 1$ reduces to a greedy search, and $w = N^T$ corresponds to an exhaustive breadth-first search.

Algorithm 1 BEAM SEARCH

Find the most likely sequence $\{z^{(t)} \in C | 1 \leq t \leq T\}$ given x with beam width $w \leq N^T$.

$q \leftarrow$ priority queue
q.insert$(0, \{\})$
for $t = 1 \dots T$ **do**
  $q' \leftarrow$ priority queue of capacity w ⋆
  for z **in** C **do**
    for l, s **in** q **do**
      q'.insert$(l + \log P(z^{(t)} = z|x, s), \{s, z\})$
  $q \leftarrow q'$
return q.max()

⋆A priority queue of fixed capacity w maintains (at most) the w *highest* values at all times.

4.3 Dynamic programming

A pathological condition that sometimes occurs with beam search is the exponential duplication of highly likely quasi-identical paths differing only at a few time steps, that quickly saturate beam width with essentially useless variations. In that context, we propose a natural extension to beam search that makes a better use of the available width w and results in better performance. The idea is to make a trade-off between an RNN for which $z^{(t)}$ fully depends on $\mathcal{A}^{(t)}$ but exact inference is intractable, and an HMM for which $z^{(t)}$ explicitly depends only on $z^{(t-1)}$ but exact inference is in $O(TN^2)$.

We hypothesize that it is sufficient to consider only the most promising path out of all partial paths with identical $z^{(t)}$ when making a decision at time t. Under this assumption, any subsequence $\{z^{(t)*} | t \leq T'\}$ of the global optimum $\{z^{(t)*}\}$ ending at time $T' < T$ must also be optimal under the constraint $z^{(T')} = z^{(T')*}$. Note that relaxing this last constraint would lead to a greedy solution. Setting $T' = T - 1$ leads to the dynamic programming-like (DP) solution of keeping track of the N most likely paths arriving at each possible label $j \in C$ with the recurrence relation:

$$l_j^{(t)} = l_{k_j^{(t)}}^{(t-1)} + P(z^{(t)} = j | x, s_{k_j^{(t)}}^{(t-1)}) \tag{15}$$

$$s_j^{(t)} = \{s_{k_j^{(t)}}^{(t-1)}, j\} \tag{16}$$

$$\text{with } k_j^{(t)} \equiv \underset{k=1}{\overset{N}{\text{argmax}}} \left[l_k^{(t-1)} + P(z^{(t)} = j | x, s_k^{(t-1)}) \right] \tag{17}$$

and initial conditions $l_j^{(0)} = 0, s_j^{(0)} = \{\}$, where the variables $l_j^{(t)}, s_j^{(t)}$ represent respectively the maximal cumulative log-likelihood and the associated partial output sequence ending with label j at time t (Algorithm 2). It is also possible to keep only the $w \leq N$ most promising paths to mimic an *effective beam width* and to make the algorithm very similar to beam search.

Algorithm 2 DYNAMIC PROGRAMMING INFERENCE

Find the most likely sequence $\{z^{(t)} \in C | 1 \leq t \leq T\}$ given x with effective width $w \leq N$.

$q \leftarrow$ priority queue
q.insert$(0, \{\})$
for $t = 1 \dots T$ **do**
  $q' \leftarrow$ priority queue of capacity w
  for z **in** C **do**
    $l, s \leftarrow \text{argmax}_{(l,s) \in q} \left[l + \log P(z^{(t)} = z | x, s) \right]$
    q'.insert$(l + \log P(z^{(t)} = z|x, s), \{s, z\})$
  $q \leftarrow q'$
return q.max()

It should not be misconstrued that the algorithm is limited to "local" or greedy decisions for two reasons: (1) the complete sequence history $\mathcal{A}^{(t)}$ is relevant for the prediction $y^{(t)}$ at time t, and (2) a decision $z^{(t)*}$ at time t can be affected by an observation $x^{(t+\delta t)}$ arbitrarily far in the future via *backtracking*, analogously to Viterbi decoding. Note also that the algorithm obviously does not guarantee a globally optimal solution z^*, but is referred to as DP due to its strong similarity to the Viterbi recurrence relations.

5. EXPERIMENTS

5.1 Setup

This section describes experiments conducted on the dataset used in the MIREX audio chord estimation task [1]. Ground truth time-aligned chord symbols were mapped to the *major/minor* and *full chord* dictionaries comprising respectively 25 and 121 chord labels:

- $C_{\text{majmin}} \equiv \{\text{N}\} \cup \{\text{maj, min}\} \times S$,
- $C_{\text{full}} \equiv \{\text{N}\} \cup \{\text{maj, min, maj/3, maj/5, maj6, maj7, min7, 7, dim, aug}\} \times S$,

[1] http://www.music-ir.org/mirex/wiki/2012:Audio_Chord_Estimation

where S represents the 12 pitch classes and 'N' is the *no-chord* label [16, 21]. This allows us to evaluate our algorithm at different precision levels. Evaluation at the major/minor level is based on chord overlap ratio (OR) and weighted average OR (WAOR), standard denominations for the average frame-level accuracy [22, 25].

Results are reported using 3-fold cross-validation. For each of the 3 partitions, 25% of the training sequences are randomly selected and held out for validation. The hyper-parameters of each model are selected over predetermined search grids to maximize validation accuracy and we report the final performance on the test set. In all experiments, we use 2 hidden layers of 200 units for the DBN, 100 hidden units for the RNN, and 8 pooling windows with $1 \leq L \leq 120$ s during pre-processing.

In order to compare our method against MIREX pre-trained systems, we also train and test our model on the whole dataset. It should be noted that this scenario is strongly prone to overfitting: from a machine learning perspective, it is trivial to design a non-parametric model performing at 100% accuracy. The objective is to contrast our results to previously published data, to analyze our models trained with equivalent features, and to provide an upper bound on the performance of the system.

5.2 Results

In Table 1, we present the cross-validation accuracies obtained on the MIREX dataset at the major/minor level using a DBN fine-tuned with chord labels z (DBN-1) and with chromagram intermediate targets $\tilde{z}$ and chord labels z (DBN-2), in addition to an RNN with DP inference. The DBN predictions are either not post-processed, smoothed with a Gaussian kernel ($\sigma = 760$ ms) or decoded with an HMM. The HPA [25] and DHMM [9] state-of-the-art methods are also provided for comparison.

Model	Smoothing	OR	WAOR
DBN-1	None	65.8%	65.2%
	Kernel	75.2%	74.6%
	HMM	74.3%	74.2%
DBN-2	None	68.0%	67.3%
	Kernel	78.1%	77.6%
	HMM	77.3%	77.2%
RNN	DP	**80.6%**	80.4%
HPA [25]	HMM	79.4%	78.8%
DHMM [9]	HMM	N/A	**84.2%**†

Table 1. Cross-validation accuracies obtained on the MIREX dataset using a DBN fine-tuned with chord labels z (DBN-1) and with chromagram intermediate targets $\tilde{z}$ and chord labels z (DBN-2), an RNN with DP inference, and the HPA [25] and DHMM [9] state-of-the-art methods. †4-fold cross-validation result taken from [9].

It is clear that optimizing the DBN with chromagram intermediate targets ultimately increases the accuracy of the classifier, and that the RNN outperforms the simpler models in both OR and WAOR. We also observe that kernel smoothing (a simple form of low-pass filtering) surprisingly outperforms the more sophisticated HMM approach. As argued previously [8], the relatively poor performance of the HMM may be due to the context information added to the input $x^{(t)}$ in equations (3-4). When the input includes information from neighboring frames, the independence property (11) breaks down, making it difficult to combine the classifier with the language model in equation (14). Intuitively, multiplying the predictions $P(z^{(t)}|x^{(t)})$ and $P(z^{(t)}|z^{(t-1)})$ to estimate the joint distribution will count certain factors twice since both models have been trained separately. The RNN addresses this issue by directly predicting the probability $P(z^{(t)}|\mathcal{A}^{(t)})$ needed during inference.

We now present a comparison between pre-trained models in the MIREX major/minor task (Table 2), where the superiority of the RNN to the DBN-2 is apparent. The RNN also outperforms competing approaches, demonstrating a high flexibility in describing temporal dependencies. Similar results can be observed at the full chord level with 121 labels (not shown).

Method	OR	WAOR
Chordino [22]	80.2%	79.5%
GMM + HMM [20]	82.9%	81.6%
HPA [25]	83.5%	82.7%
Proposed (DBN-2)	89.5%	89.8%
Proposed (RNN)	**93.5%**	**93.6%**

Table 2. Chord recognition performance (training error) of different methods pre-trained on the MIREX dataset.

To illustrate the computational advantage of DP inference over beam search, we plot the WAOR as a function of beam width w for both algorithms. Figure 3 shows that maximal accuracy is reached with a much lower width for DP ($w^* \simeq 10$) than for beam search ($w^* > 500$). The former can be run in 10 minutes on a single processor while the latter requires 38 hours for the whole dataset. While the time complexity of our algorithm is $O(TNw)$ versus $O(TNw \log w)$ for beam search, the performance gain can be mainly attributed to the possibility of significantly reducing w while preserving high accuracy. This is due to an efficient pruning of similar paths ending at $z^{(t)}$, presumably because the hypothesis stated in Section 4.3 holds well in practice.

6. CONCLUSION

We presented a comprehensive system for automatic chord recognition from audio music, that is competitive with existing state-of-the-art approaches. Our RNN model can learn basic musical properties such as temporal continuity, harmony and temporal dynamics, and efficiently search for the most musically plausible chord sequences when the audio signal is ambiguous, noisy or weakly discriminative. Our DP algorithm enables real-time decoding in live situations and would also be applicable to speech recognition.

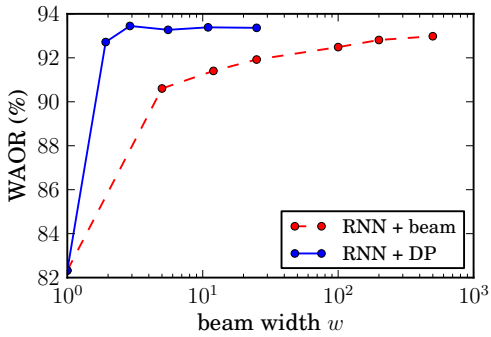


Figure 3. WAOR obtained on the MIREX dataset with the beam search and dynamic programming algorithms as a function of the (effective) beam width w.

7. REFERENCES

[1] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

[2] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu. Advances in optimizing recurrent networks. In *ICASSP*, 2013.

[3] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. on Neural Networks*, 5(2):157–166, 1994.

[4] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and adaboost for music classification. *Machine Learning*, 65(2-3):473–484, 2006.

[5] S. Böck and M. Schedl. Polyphonic piano note transcription with recurrent neural networks. In *ICASSP*, pages 121–124, 2012.

[6] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *ICML 29*, 2012.

[7] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. High-dimensional sequence transduction. In *ICASSP*, 2013.

[8] P. Brown. *The acoustic-modeling problem in automatic speech recognition*. PhD thesis, Carnegie-Mellon University, 1987.

[9] R. Chen, W. Shen, A. Srinivasamurthy, and P. Chordia. Chord recognition using duration-explicit hidden Markov models. In *ISMIR*, 2012.

[10] G. E. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42, 2012.

[11] D. Eck and J. Schmidhuber. Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. In *NNSP*, pages 747–756, 2002.

[12] A. Graves. Sequence transduction with recurrent neural networks. In *ICML 29*, 2012.

[13] Ç. Gülçehre and Y. Bengio. Knowledge matters: Importance of prior information for optimization. *ICLR*, 2013.

[14] P. Hamel, Y. Bengio, and D. Eck. Building musically-relevant audio features through multiple timescale representations. In *ISMIR*, 2012.

[15] P. Hamel and D. Eck. Learning Features from Music Audio with Deep Belief Networks. In *ISMIR*, pages 339–344, 2010.

[16] C. Harte. *Towards automatic extraction of harmony information from music signals*. PhD thesis, University of London, 2010.

[17] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*, 29(6):82–97, 2012.

[18] G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[19] E. J. Humphrey and J. P. Bello. Rethinking automatic chord recognition with convolutional neural networks. In *ICMLA 11*, volume 2, pages 357–362, 2012.

[20] M. Khadkevich and M. Omologo. Time-frequency reassigned features for automatic chord recognition. In *ICASSP*, pages 181–184. IEEE, 2011.

[21] M. Mauch. *Automatic chord transcription from audio using computational models of musical context*. PhD thesis, University of London, 2010.

[22] M. Mauch and S. Dixon. Approximate note transcription for the improved identification of difficult chords. In *ISMIR*, pages 135–140, 2010.

[23] M. C. Mozer. Neural network music composition by prediction. *Connection Science*, 6(2):247–280, 1994.

[24] J. Nam, J. Ngiam, H. Lee, and M. Slaney. A classification-based polyphonic piano transcription approach using learned feature representations. In *ISMIR*, 2011.

[25] Y. Ni, M. McVicar, R. Santos-Rodríguez, and T. De Bie. An end-to-end machine learning system for harmonic analysis of music. *Audio, Speech, and Language Processing*, 20(6):1771–1783, 2012.

[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Dist. Proc.*, pages 318–362. MIT Press, 1986.

[27] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In *Parallel Dist. Proc.*, pages 194–281. MIT Press, 1986.

VIRTUALBAND: INTERACTING WITH STYLISTICALLY CONSISTENT AGENTS

Julian Moreira
Sony CSL
julian.moreira.fr@gmail.com

Pierre Roy
Sony CSL
roy@csl.sony.fr

François Pachet
Sony CSL
pachetcsl@gmail.com

ABSTRACT

VirtualBand is a multi-agent system dedicated to live computer-enhanced music performances. VirtualBand enables one or several musicians to interact in real-time with stylistically plausible virtual agents. The problem addressed is the generation of virtual agents, each representing the style of a given musician, while reacting to human players. We propose a generation framework that relies on *feature-based interaction*. Virtual agents exploit a style database, which consists of audio signals from which a set of MIR features are extracted. Musical interactions are represented by directed connections between agents through these features. The connections are themselves specified as mappings and database filters. We claim that such a connection framework allows to implement meaningful musical interactions and to produce stylistically consistent musical output. We illustrate this concept through several examples in jazz improvisation, beatboxing and interactive mash-ups.

1. INTRODUCTION

Collective improvisation is a group practice in which several musicians contribute their part to produce a coherent musical whole. Each musician typically brings in musical knowledge, taste, and technical skills, more generally a *style*, which makes him or her unique and recognizable. However, good improvisations are not only about putting together the competence of several individuals. Listening and interacting to each other is crucial, as it enables each musician to adapt to the global musical output in terms of rhythm, intensity, harmony, etc. The combination of individual styles with the definition of their interaction defines the quality of a music band. In short, group improvisation can be seen as principled interactions between *stylistically consistent* agents.

Many works have attempted to model and simulate the behavior of a real musician. A MIDI-based model of an improviser's personality is proposed in [6], to build a virtual trio system, but no explicit interactions between virtual agents and real musicians are proposed. Improtek [11] is a system for live improvisation that generates musical sequences built in real-time from a live source using concatenative synthesis. Improtek plays along with a musician improvising on harmonic and beat-based music, in the style of this musician, but interactions with him are based on feature similarity measures, thereby limiting the scope of man-machine interactions. The Jambot [4] infers in real-time various features from an audio signal and then produces percussive responses, following alternatively predefined behaviors, but the style of the Jambot itself is not clearly defined. Beatback [7] generates a MIDI signal based on rhythmic patterns and using Markov models. Besides the limitations of MIDI, interactions are limited to rhythmic elements only. This limitation also applies to the Chimera Architecture [3], a system that infers rhythmical information from an audio input, and generates percussive sounds depending on scenarios that fit with the current musical context. [9] presents a system that learns rhythmic patterns from drum audio recording and synthesizes musical variations from the learnt sequence. In addition to being dedicated to percussive sounds only, this system has no real-time live application. [16] presents an interactive music system driven by syncopation measurements, but also addresses rhythmical features only. [10] describes a system that interacts in real-time with a human musician, by adapting its behavior both on prior knowledge (database of musical situations) and parameters extracted during the current session, but this system forces the musician to manipulate a graphical interface during the improvisation, which makes the system hardly usable in a live performance for a solo musician.

In this paper we revisit the issue of musical interaction with stylistically consistent agents by taking a feature-based approach to the problem. We introduce VirtualBand (VB), a *reactive* music multi-agent system in which human musicians can control and interact with virtual musicians that retain their own style.

In VB, both human and virtual musicians are represented by agents that interact together through feature-based interactions. *Human agents* extract their features at every beat from the audio input of their corresponding musician. *Virtual agents* use features of audio *chunks* stored in a database. Interactions are modeled by *connections*, which are the core contribution of our approach. A connection is a directed link from a *master* (human or virtual) agent to a *slave* virtual agent. In reaction to a feature value pro-

vided by the master agent, the connection specifies to the slave agent which audio chunk to play from its database. VB can be seen as a reactive rather than deliberative multi-agent system [8]. Its potential lies in the seemingly infinite possibilities provided by feature interaction, as illustrated in this paper.

In Section 2, we describe the main components of VB. In Section 3, we illustrate feature interaction with several configurations of the system for jazz improvisation. Section 4 describes applications of VB in two other musical contexts: beatboxing and automatic mash-up.

2. SYSTEM DESCRIPTION

The core of VB consists of 1) a clock that establishes the tempo and sends notifications at each beat to every agent - we consider fixed tempos in the current version, 2) a set of agents that receives these notifications and generate audio and 3) an audio playback engine.

2.1 Agents

There are two types of agents: human and virtual agents, representing respectively actual and virtual musicians. Human agents are responsible for extracting acoustic features from the audio signal of real musicians in real-time. These features are the main controlling device for virtual agents via connections, as explained below.

Virtual agents are designed to play *in the style of* the musicians they represent. To build a virtual agent, we record the musician in a musical situation that fits with the targeted performance context. The recorded audio is stored in a *style database*, organized in musically relevant *chunks* (usually beats and bars). The musician is asked to play so as to fully express his musical style, i.e., cover a large range of musical situations (e.g., playing with different intensities, in different moods, staccato or legato, using various patterns). Of course, it is difficult to express one's style exhaustively, so style databases are limited to specific musical *situations*, e.g., defined by musical genre such as Bossa Nova, Swing, etc. and a tempo. Note that the subject of *individual style capture* is still in its infancy and further studies should refine this concept.

In the database each chunk is associated to a set a feature values. A set of MIR features are automatically extracted from the audio signal of the chunk, e.g., RMS, numbers of onsets, spectral centroid, harmonic to noise ratio, or chroma. Contextual features are also associated to each chunk such as the harmony. Note that the harmony can either be extracted automatically or be imposed by a chord sequence. During performance, each virtual agent uses concatenative synthesis [15] to generate music streams as seamless concatenations of database chunks.

2.2 Connections as a Combination of Mappings

A connection between two agents models the *intention* underlying a musical interaction between two musicians. Connections are directed from a master agent A_M to a slave agent A_S, and relate a master feature F_M of A_M to a slave feature F_S of A_S.

At every beat, the connection selects the audio chunk that A_S is going to play, among the available chunks of its associated database D_S. To perform this task, the connection applies two successive mappings: 1) a *feature-to-feature* mapping from the domain X_M of F_M to the domain X_S of F_S, and 2) a *feature-to-chunk* mapping from X_S to D_S.

2.2.1 The feature-to-feature (f2f) mapping

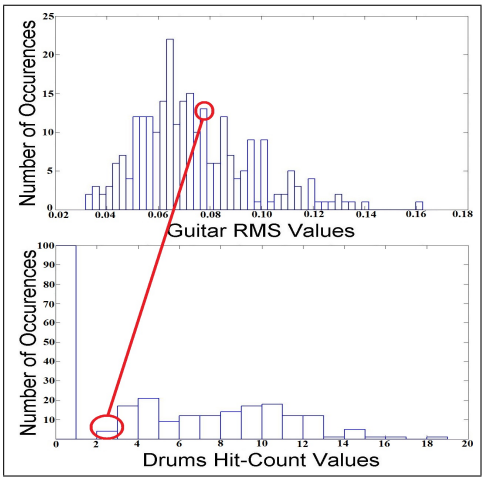


Figure 1. Illustration of a percentile mapping (in red) between two distributions. 1) RMS values extracted from a guitar track (top) and 2) hit-counts extracted from a drum track (bottom). Both tracks come from a recording of the jazz standard *Body and Soul*.

F_M and F_S may take their values in different domains (for instance integers or floating points) and with different value distributions. Normalizing the values of F_M and F_S allows to define a simple feature-to-feature (f2f) mapping, that maps any value x_M of F_M onto the same value x_M of F_S. However, with such a mapping, the distribution of the virtual agent's actual output doesn't match the distribution of the database, i.e., the virtual agent doesn't play consistently in the style of the musician it represents.

Instead we use a percentile-based f2f mapping, which preserves this distribution. If D is the domain of a variable x, the percentile function is defined by:

$$percentile(x, D) = \frac{|\{d \in D \mid d < x\}|}{|D|} \quad (1)$$

The f2f mapping is then defined as:

$$f2f(x_M) = percentile^{-1}(percentile(x_M, D_M), D_S) \quad (2)$$

Suppose a guitarist wants a virtual drummer to adapt its density to the guitar's energy. As a proxy of energy, we use master feature F_M = RMS for the guitar. Drums' density is represented by slave feature F_S = hit-count (i.e., number of onsets in the chunk). Fig. 1 shows the distribution of the RMS values of a typical guitar recording, and of the

hit-count values of a typical drum recording. The distributions strongly differ, by their total number of values (257 values for the RMS, 157 for the hit-count), ranges (RMS values range from 0.03 to 0.16 whereas hit-count values range from 0 to 19) and number of bins (for the RMS, there are 50 different bins, and only 19 for the hit-count). The f2f mapping selects the hit-count value x_S with the same percentile as the RMS value x_M. A value $x_M = 0.076$, corresponds to a percentile of 0.067. The hit-count value with the same percentile is 2, as shown in Fig. 1.

2.2.2 *The feature-to-chunk (f2c) mapping*

Eventually, the connection selects which chunk to play, given x_S. This is implemented by the feature-to-chunk (f2c) mapping, which represents the *musical intention* behind the connection. We specify this mapping *operationally*, by composing various *filters*. Given x_S and a subset $C \subseteq D_S$ of chunks, each filter selects a subset of C that satisfies a specific rule. Filters are defined once for all, as illustrated in the following examples:

- *ClosestChunks*$(C, x_S) \stackrel{def}{=} \underset{c \in C}{\text{argmin}} \mid F_S(c) - x_S|$
- *FarthestChunks*$(C, x_S) \stackrel{def}{=} \underset{c \in C}{\text{argmax}} \mid F_S(c) - x_S|$
- *MatchingChunks*$(C, x_S) \stackrel{def}{=} \{c \in C \mid F_S(c) = x_S\}$
- *UnmatchingChunks*$(C, x_S) \stackrel{def}{=} \{c \in C \mid F_S(c) \neq x_S\}$
- *ChordMatchingChunks*$(C, x_S) \stackrel{def}{=} \{c \in C \mid chord(c)$ is substitutable to $x_S\}$
- At a given beat b,
  AdaptiveClosestChunks(C, x_S)
  $\stackrel{def}{=}$ { *ClosestChunks*(C, x_S) **if** b is the first beat of a bar; $\{c\}$ where c is the chunk that follows in D_S the chunk currently playing **otherwise** }
- *RandomChunk*$(C) \stackrel{def}{=} random(C)$

The f2c mapping of a connection is a composition of filters, such as:

RandomChunk$(f_n(\ldots f_2(f_1(D_S, x_S), x_S) \ldots, x_S))$

where $f_1, \ldots, f_n$ are filters.

In our example of the guitar controlling the drums, we use a *ClosestChunks* filter. A value of $x_M = 0.076$ for the guitar RMS will trigger a drum chunk with a hit-count value closest to 2.

When the composition of filters yields an empty subset, specific procedures are applied, such as using a default style database for the instrument.

2.2.3 *Multiple masters for one slave*

To better approximate the complexity of interaction occurring in a real band, a slave agent may be connected to several master agents. In this case, the f2f mappings are first computed independently and then intersected.

2.3 Representing Harmony as a Connection

In tonal music, such as jazz, musicians improvise on a chord sequence that is known to all beforehand. VB implements such shared harmonic information by a specific agent that represents the chord sequence.

The chord sequence agent A_C provides a unique feature whose value x_C is the name of the next chord in the sequence. A_C is typically connected to harmony-dependent agents, such as pitched instruments (e.g., piano, guitar). For such a virtual agent A, the connection is defined as follows: the master agent is A_C, the master feature value is x_C, and the slave agent is A.

Depending on the expected behavior of the virtual agent, with respect to harmony, various slave features and mappings may be used.

The most typical example is that of a piano comping agent A_P that is expected to play chords in the same harmony as x_C. Each audio chunk of the database D_P is associated to a feature value x_P which represents the corresponding chord name. In this case, the slave feature value is x_P, the f2f mapping is the identity: $x_P = x_C$, and the filter is *ChordMatchingChunks*. Note that the matching is not necessarily strict, but can use substitution rules, as explained in the next section.

2.4 Reflexive Style Databases

VB also features a *memoryless* mode, i.e., which doesn't require pre-recorded style databases. One motivation is to avoid "canned music" effects caused by the audience being unaware of the content of the style databases. In this mode, databases are built on-the-fly, typically like interactive systems such as Continuator [12] or Omax [11]. Reflexive style databases are used to generate various species of *self-accompaniments* such as duos or trios with oneself [13].

Conceptually, reflexive style databases do not differ from pre-recorded ones. Technically, they raise various real-time issues (segmentation and feature extraction must be performed in real-time without interfering with the main VB loop) that are not discussed in this paper.

More interestingly, reflexive style databases raise *sparsity* issues. Because a database contains only what the musician has played so far, its size will typically be much smaller than that of pre-recorded databases. As a consequence, the system has far fewer possibilities for generation. This is particularly annoying in contexts using predetermined chord grids. In principle, the system requires a long *feeding phase*, to accumulate at least one chunk for every chord of the sequence before it can start playing back relevant audio material. Such a feeding phase is usually boring for the musician as well as for the audience.

In order to reduce feeding, we propose to automatically expand style databases by using audio transformations, such as *transpositions*, implemented with pitch shifting algorithms [14]. In VB, the audio chunks of a database can be pitch-shifted so that the transposed bars can be used in new harmonic situations.

We also use so-called *substitution rules* to further expand the style database. Substitution rules consist in re-

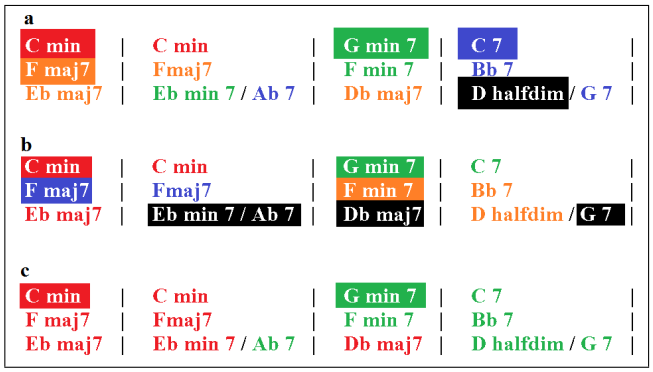


Figure 2. The feeding phase reduction using (a) transpositions, (b) substitutions, and (c) a combination of both.

placing a chord by another one with an equivalent harmonic function. This operation is widely used in jazz to bring variety in performance [1]. VB uses chord substitutions to play in a certain harmonic context audio that was recorded in another harmonic context, provided the two contexts may be substituted.

By combining transpositions and substitutions, the feeding phase is drastically reduced. The three examples of Fig. 2 show how to harmonize the song *Solar* from a limited number of input chords, using transpositions (a), substitutions (b), and a combination of both (c). The input chords (highlighted) generate the remaining chords of Solar (marked in the same color). Without substitutions or transpositions, 12 input chords would be required in the feeding phase. In contrast, (a) requires 5 input chords, (b) requires 8, and (c) only 2. For instance in (c), *G min 7* can be substituted for *C 7* using substitution "G min 7 : C 7", but also *F min 7* using a 1 tone downward transposition, and *Bb 7* with a combination of these two operations.

3. APPLICATION TO JAZZ IMPROVISATION

The following examples illustrate various configurations of VB with one human guitarist and one or two reflexive virtual agents. In the first and second examples, we present duos in which the virtual agent is controlled respectively by a spectral centroid feature and by a rhythm pattern feature extracted from the human guitarist. Then, we extend the example to a guitar trio configuration. An accompanying web site illustrates all these configurations with videos of the corresponding performances [1].

3.1 Spectral Centroid-Based Duo

Two jazz guitarists improvising together commonly use a question-answer interaction scheme: one of the guitarists plays a melody; as soon as he finishes, the other guitarist plays another melody that borrows elements from the first one. Typically the answering melody is in the same pitch range or shares similar rhythm patterns with the original melody. While a guitarist proposes a melody, the other one either stops playing or accompanies the first one with, e.g., chord comping.

[1] http://francoispachet.fr/virtualband/virtualband.html

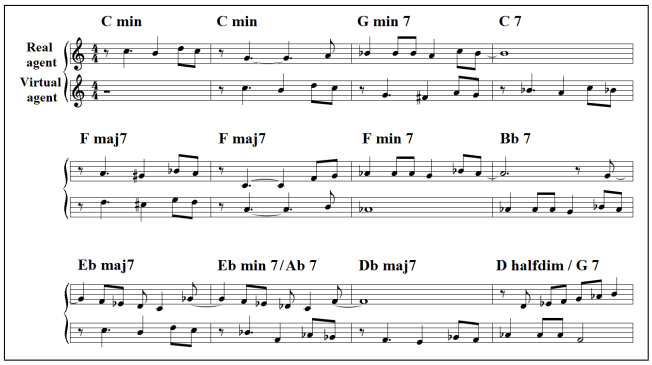


Figure 3. Score of a spectral-centroid based duo: a human agent (A_H) plays the melody of Solar. A virtual agent A_V matches the spectral centroid of A_H, with a one-bar delay. At bar 11 (*Db maj7*) A_V uses a transposition of bar 5 (*F maj7*) of A_H to match the spectral centroid of bar 10 of A_H. At bar 8 (*Bb 7*), substitution "F min 7 : Bb 7" allows A_V to play bar 7 of A_H (*F min 7*).

We simulate these scenarios with various configurations of VB based on pitch features of audio content. A reflexive style database records the incoming audio of a human guitarist A_H and extracts the spectral centroid of each bar. We chose the spectral centroid as an approximation of pitch, but more refined descriptors could be used interchangeably. A virtual agent A_V, associated to the database, is connected to A_H. At each bar, this connection is specified by the following elements:

- master feature F_H = spectral centroid of A_H; slave feature F_V = spectral centroid of A_V;
- f2f = percentile from a value F_H to a value of F_V (details in Section 2.2.1);
- f2c = *ClosestChunks* (details in Section 2.2.2).

Note that in the meantime, another connection ensures that A_V also plays according to the harmonic constraints (see Section 2.3 for details).

In this configuration, the system behaves as a self harmonizer: A_V follows the spectral centroid of A_H with a one-bar delay (see Fig. 3), using music material that sounds like what the guitarist just played.

Replacing the filter in the configuration above by:

- f2c = *FarthestChunks*

implements another musical intention: the agent plays audio that is far away pitch-wise to the human's input. In this configuration, the two outputs (human and virtual guitar) are clearly distinct.

3.2 Rhythm-Based Duo

Question-answer musical dialogues can also be based on rhythmical similarities. A guitarist plays a rhythm pattern for a few bars, and the other one responds by playing a similar pattern. We introduce a feature that represents the rhythm pattern: the *RMS profile*. This profile is obtained

by computing the RMS 12 times per beat over one bar. The agent plays back chunks with a similar profile, with a systematic one-bar delay. Technically, the connection is specified by:

- master feature F_H = RMS profile of A_H; slave feature F_V = RMS profile of A_V;
- f2f = identity, i.e., for a value x_H of F_H and a value x_V of F_V: $x_V = x_H$;
- f2c = *ClosestChunks*. The distance between two RMS profiles is the scalar product of the two vectors.

This mode is fun and lively as the interaction is easily perceived by the musician and the audience. However, it requires larger databases, so it should be used when the system has accumulated enough rhythm samples to play back interesting variations.

As before, changing the filter to:

- f2c = *FarthestChunks*.

implements a very different musical interaction: the virtual agent plays chunks that are dissimilar to the input of the musician, which is more difficult to anticipate for a human than similar patterns.

3.3 Trio

In a typical jazz guitar trio, one of the guitarists improvises melodies on a harmonic grid, while the other two provide respectively chordal and bass accompaniments. These roles (melody, chords, and bass) represent different guitar *playing modes*. During an improvisation, guitarists typically shift roles in turn. When a musician takes the lead (solo), the other guitarists adapt their behavior so that each mode is always played by someone. With VB, we represent this configuration so that a guitarist can be self-accompanied by two reflexive virtual agents, sharing the same reflexive database.

In addition to the RMS profile, we extract the *playing mode* from each recorded beat, among four possible modes: melody, chord, bass and silence [2]. Each virtual agent is associated to a unique playing mode (one to the bass, one to the chords), and agents follow a mutually exclusive rule, i.e., they play only if the human guitarist is not playing in the same mode, following [13]. Given a virtual agent A_V (virtual bass or virtual chords), such a rule is easily modeled by a connection:

- master feature F_H = playing mode of A_H; slave feature F_V = playing mode of A_V;
- f2f = identity;
- f2c = *UnmatchingChunks*.

Furthermore, like in the previous example, each agent follows the rhythmical patterns of the guitarist. For instance, a walking bass, or chord comping (a lot of notes per bar, regularly spaced) can be triggered by playing a fast and regular melody. This configuration provides the feeling of a standard jazz guitar trio to a solo musician.

4. OTHER APPLICATIONS

In this section, we describe applications of VB in two other musical contexts: beatboxing and mash-ups; also illustrated by videos on our web site.

4.1 Reflexive Beatboxing

Beatboxing is a music style where musicians use their mouth to simulate percussion. Beatboxing also involves humming, speech or singing (see [17]). Common beatboxers typically alternate between modes, but some great beatboxers are able to play two modes at the same time (e.g., percussion and humming).

Beatboxing with VB aims at augmenting the performance of a moderately good beatboxer by allowing him or her to play, via reflexive virtual agents, several modes at the same time. Using the same connection settings as in Section 3.3, the system records and stores in a reflexive database the incoming audio of the real beatboxer. From this audio, the database distinguishes between two playing modes: the percussion mode and the humming mode. Then two virtual agents are connected to this database, one representing the percussion mode and the other the humming mode. The classification is performed following a similar scheme to [2], except for the set of features selected by the classifier (here harmonic-to-noise ratio, spectral centroid and Yin). Following a mutually exclusive principle, virtual agents play alternatively, depending on the mode of the human beatboxer, and following his or her rhythmic patterns. Table 4 illustrates a typical session with such an augmented beatboxer.

	1	2	3	4
human	t---|	t-t-|	kk--|	t--t|
A_p	----|	----|	----|	----|
A_h	----|	----|	----|	----|
	5	6	7	8
human	O-O-|	O-O-|	A-A-|	AO--|
A_p	----|	t-t-|	t-t-|	t-t-|
A_h	----|	----|	----|	----|
	9	10	11	12
human	tt--|	tktk|	OAOA|	OAOA|
A_p	kk--|	----|	----|	tktk|
A_h	----|	AO--|	A-A-|	----|
	13	14	15	16
human	tktk|	tktk|	----|	----|
A_p	tktk|	----|	----|	tktk|
A_h	----|	OAOA|	OAOA|	OAOA|

Figure 4. A 16-bar performance of the beatboxing system: `t` and `k` are percussive sounds, `O` and `A` are hummed vowels. A_P is the percussive agents, and A_H is the humming agent. Agents follow a mutually exclusive principle and try to match the human's rhythm with a one-bar delay.

4.2 Mash-up

A mash-up is created by blending two or more songs, usually by overlaying a track of one song over the tracks of another [5]. Mash-ups exploit multi-track songs whose tracks are available as separated audio files. A straightforward way to implement mash-up with VB is to represent each

track of each song as a virtual agent. The database of each agent consists of all the audio chunks obtained by segmenting the corresponding track. The mash-up is obtained by muting and replacing one agent by a track agent representing the same instrument of another song. The virtual agent representing the replacing track is connected to the muted agent of the original song.

For instance, one can replace the drums in song *Roxanne* (*The Police*) by another drummer. The muted agent that represents the original drum track controls the substitute drummer through rhythm patterns. Technically, the settings of the connection are the same as presented in the first example of Section 3.2. On the accompanying web site we provide mash-ups of *Roxanne* with the drummer of 1) *Smells Like Teen Spirit* (*Nirvana*), 2) *Hey* (*Pixies*), 3) Bossa Nova and 4) Funk drums played by Jeff Boudreaux.

We can hear in the provided example each drummer playing in his style while seemingly following the song's structure by, e.g., playing breaks at the right time, or playing more intensively on choruses and bridges.

5. CONCLUSION

We revisit the problem of interacting with stylistically consistent agents from a MIR viewpoint. In VirtualBand, interactions are specified using *features pairs*, taken from the vast library of features developed in MIR. VB agents are reactive but not deliberative, i.e., they do not attempt to exhibit autonomy, make goals, or plan ahead. But the examples show that even with *only* reactive agents, rich and complex interactions can take place, by exploiting the complex correlations that typically occur between pairs of features computed on human audio signals.

However, VB only scratches the surface of the new field of *individual style modeling*. Current work focuses on issues like how to "saturate" a style database, or how to predict the emergence of long-term structure from low-level feature interaction.

6. ACKNOWLEDGEMENT

This research is conducted within the Flow Machines project which received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 291156.

7. REFERENCES

[1] J. Coker. *Elements of the Jazz Language for the Developing Improvisor*. Alfred Music Publishing, 1997.

[2] R. Foulon, F. Pachet, and P. Roy. Automatic classification of guitar playing modes. *Proc. of the CMMR Symposium*, 2013.

[3] T. Gifford and A.R. Brown. Do androids dream of electric chimera? In *Proc. of the ACMC*, pages 56–63, 2009.

[4] T. Gifford and A.R. Brown. Beyond reflexivity: Mediating between imitative and intelligent action in an interactive music system. In *Proc. of the HCI Conference*, 2011.

[5] J. Grobelny. Mashups, sampling, and authorship: A mashupsampliography. *Music Reference Services Quarterly*, 11(3-4):229–239, 2008.

[6] M. Hamanaka, M. Goto, H. Asoh, and N. Otsu. A learning-based jam session system that imitates a player's personality model. In *Proc. of the IJCAI*, volume 18, pages 51–58, 2003.

[7] A. Hawryshkewich, P. Pasquier, and A. Eigenfeldt. Beatback: A real-time interactive percussion system for rhythmic practise and exploration. In *Proc. of the NIME Conference*, pages 100–105, 2011.

[8] L. Iocchi, D. Nardi, and M. Salerno. Reactivity and deliberation: A survey on multi-robot systems. In M. Hannebauer, J. Wendler, and E. Pagello, editors, *Balancing Reactivity and Social Deliberation in Multi-Agent Systems*, volume 2103 of *Lecture Notes in Computer Science*, pages 9–34. Springer, 2000.

[9] M. Marchini and H. Purwins. Unsupervised generation of percussion sound sequences from a sound example. In *Proc. of the SMC Conference*, 2010.

[10] A. Martin, A. McEwan, C.T. Jin, and W. L. Martens. A similarity algorithm for interactive style imitation. In *Proc. of the ICMC*, pages 571–574, 2011.

[11] J. Nika and M. Chemillier. Improtek: integrating harmonic controls into improvisation in the filiation of OMax. In *Proc. of the ICMC*, pages 180–187, 2012.

[12] F. Pachet. The continuator: Musical interaction with style. *Journal of New Music Research*, 32(3):333–341, 2003.

[13] F. Pachet, P. Roy, J. Moreira, and M. d'Inverno. Reflexive loopers for solo musical improvisation. In *Proc. of the SIGCHI Conference*, CHI '13, pages 2205–2208. ACM, 2013. Best paper honorable mention award.

[14] C. Schörkhuber and A. Klapuri. Pitch shifting of audio signals using the constant-q transform. In *Proc. of the DAFx Conference*, 2012.

[15] D. Schwarz. Current research in concatenative sound synthesis. In *Proc. of the ICMC*, pages 9–12, 2005.

[16] G. Sioros, A. Holzapfel, and C. Guedes. On measuring syncopation to drive an interactive music system. In *Proc. of the ISMIR Conference*, pages 283–288, 2012.

[17] D. Stowell and M. D. Plumbley. Characteristics of the beatboxing vocal style. *Dept. of Electronic Engineering, Queen Mary, University of London, Technical Report, Centre for Digital Music C4DMTR-08-01*, 2008.

Oral Session 4:

Music Signal Analysis

SPARSE MODELING FOR ARTIST IDENTIFICATION: EXPLOITING PHASE INFORMATION AND VOCAL SEPARATION

Li Su and Yi-Hsuan Yang
Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan
lisu@citi.sinica.edu.tw, yang@citi.sinica.edu.tw

ABSTRACT

As artist identification deals with the vocal part of music, techniques such as vocal sound separation and speech feature extraction has been found relevant. In this paper, we argue that the phase information, which is usually overlooked in the literature, is also informative in modeling the voice timbre of a singer, given the necessary processing techniques. Specifically, instead of directly using the raw phase spectrum as features, we show that significantly better performance can be obtained by learning sparse features from the negative derivative of phase with respect to frequency (i.e., group delay function) using unsupervised feature learning algorithms. Moreover, better performance is achieved by using singing voice separation as a pre-processing step, and then learning features from both the magnitude spectrum and the group delay function. The proposed system achieves 66% accuracy in identifying 20 artists from the artist20 dataset, which is better than a prior art by 7%.

1. INTRODUCTION

Singing voice is one of the most prominent characteristics in music. To model the vocal signal accurately, much of the effort has been focus on two topics: 1) extracting speech-related features of the audio signal [25, 27], and 2) separating human voice from the accompaniment [9,18,24, 29].

For feature extraction in speech processing, the phase-based features has been noticed with the application of speaker recognition or the reconstruction of intelligible voice [13, 21, 23, 26]. Instead of using phase only, these works adopted the *group-delay function* (the negative derivative of phase by frequency), sometimes also merged together with amplitude-based features. On the other hand, the application of phase on music has been limited mostly in signal-level, such as onset detection [2, 5, 16, 17] and pitch tracking [10]. For high-level musical concepts, like genre or the timbre of the vocal artist, phase information has been rarely discussed. Since singing timbre is closely related to characteristics of the speech signal, it is worth investigating the use of phase derivatives for artist identification from popular songs.

As for vocal separation, one remarkable development emerged recently is the sparse and low-rank matrix decomposition, also known as robust principal component analysis (RPCA) technique. It notices that the main melody and the accompaniment can be suitably regarded as the sparse and low-rank counterparts respectively in an audio spectrogram [15, 30], since the former involves only a few notes at a time, and the latter are typically contributed by repetition of metre and harmonic structure. This technique satisfies the requirement of artist identification, as such application needs an efficient algorithm to remove the accompaniment and preserve the vocal (mostly melody) information.

Recent years have witnessed progression of sparse modeling techniques [4, 6, 28], which allow for constructing a succinct representation of raw features as a combination of only a few *atoms* learned from an external data collection [22]. The resulting signal reconstruction has been shown robust to noise or corruptions of data. In consequence, sparse coding techniques have been applied in many fields, including MIR. For example, Yeh *et al.* [31] demonstrated accuracy on par with state-of-the-systems for genre classification using sparse features learned by sparse modeling techniques.

This paper will investigate the sparse modeling techniques for singing voice separation and unsupervised feature learning of group-delay functions. In what follows, we will provide the details of the proposed system in Section 2, followed by experimental evaluations in Section 3. We will discuss the main findings and limitations in Section 4 and conclude the paper in Section 5.

2. SYSTEM OVERVIEW

Figure 1 shows the flow diagram of the system we implemented for artist identification. The system makes use of a collection of songs, named the "training corpus," to build the audio dictionaries, which are used to compute the sparse representation of a querying audio signal. Prior to dictionary learning or sparse coding, frame-level features are extracted from the audio files. We consider both the *magnitude-based* and *phase-based* features derived from the short-time Fourier transform (STFT), by taking the log-magnitude spectrogram as the amplitude-based feature,

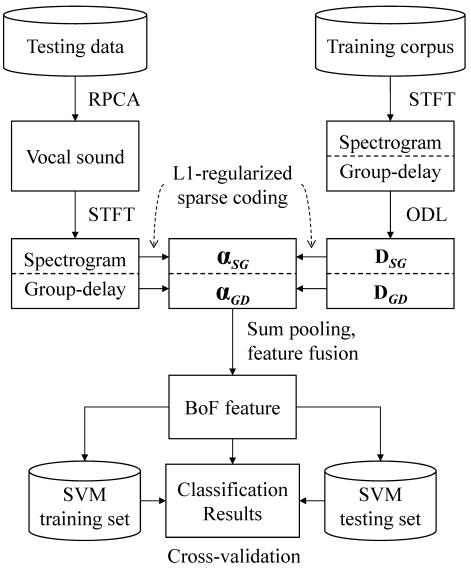


Figure 1. Proposed artist identification system.

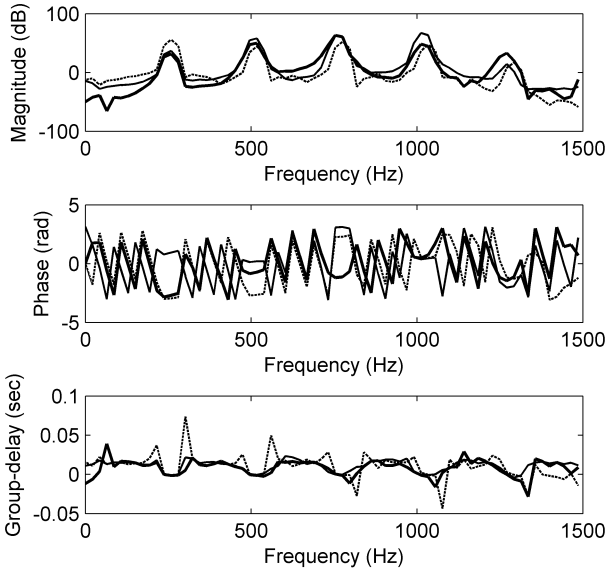


Figure 2. Examples of magnitude spectra, group-delay functions and phase profiles for tenor vocal /ah/ with fundamental frequencies at C4, selected from RWC Musical Instrument Sound Database [11]. Bold solid line: normal (less vibrato); thin solid line: vibrato; thin dashed line: falsetto.

and the group-delay as the phase-based feature.

Two types of dictionaries, $\mathbf{D}_{SG}$ and $\mathbf{D}_{GD}$, both of size $m \times k$, are respectively learned from the spectrograms and group-delay functions gathered from the training corpus using the online dictionary learning (ODL) algorithm (see Section 2.2). When there is a large overlap of artists between the training corpus and the querying songs, the generalizability of the learned dictionary might be limited (this is the so-called *transductive learning* setup). Therefore, it is preferable to use a training corpus that is representative enough, but is disjoint from the querying songs in the experiments.

As for testing, we first separate the query songs into singing voice and music accompaniment using RPCA technique, one of the state-of-the-art algorithms for singing voice separation [15]. Then, the log-magnitude spectrogram and group-delay functions are then encoded by $\mathbf{D}_{SG}$ and $\mathbf{D}_{GD}$ respectively by l_1-regularized sparse coding, engendering the codewords α_{SG} and α_{GD}. Each input feature is normalized to its Euclidean norm before sparse coding. After sparse coding, bag-of-frames (BOF) features are obtained by summing over all the frame-based features α_{SG} and α_{GD}, respectively, thereby creating a histogram of the cumulative term occurrence of the dictionary atoms [19]. Then we perform feature fusion by concatenating the two BOF features encoded from spectrograms and group-delay functions. Finally, the performance is evaluated by multi-class support vector classification in a cross-validation scheme.

2.1 Group Delay Function

Consider a general representation of short-time Fourier transform (STFT) of a time-domain signal $x(t)$:

$$\mathrm{S}_x^h(t,\omega) = \int_{-\infty}^{\infty} x(\tau)\, h^*(t-\tau)\, e^{-j\omega\tau} d\tau \quad (1)$$

$$= \mathrm{M}_x^h(t,\omega)\, e^{j\Phi_x^h(t,\omega)}, \quad (2)$$

where $\mathrm{S}_x^h(t,\omega) \in \mathbb{C}$ is the two-dimensional STFT representation on time-frequency plane, $h(t)$ is the window function, $\mathrm{M}_x^h(t,\omega)$ and $\Phi_x^h(t,\omega)$ of (1) are the amplitude and the phase of the STFT representation, respectively. By taking the natural logarithm of Eq. (2), we obtain the *log-magnitude spectrogram* in the real part and the *phase* the imaginary part:

$$\log \mathrm{M}_x^h(t,\omega) = \mathrm{Re}\left(\log \mathrm{S}_x^h(t,\omega)\right), \quad (3)$$

$$\Phi_x^h(t,\omega) = \mathrm{Im}\left(\log \mathrm{S}_x^h(t,\omega)\right). \quad (4)$$

Taking the negative derivative of phase (4) with respect to frequency, we have

$$-\frac{\partial \Phi_x^h(t,\omega)}{\partial \omega} = \mathrm{Re}\left(t - \frac{\mathrm{S}_x^{\mathcal{T}h}(t,\omega)}{\mathrm{S}_x^h(t,\omega)}\right), \quad (5)$$

where $\omega = 2\pi f$ is the angular frequency and $\mathcal{T}(\cdot)$ is the operator such that $\mathcal{T}h(t) = t \cdot h(t)$. The first term t denotes the current time, while the second term is defined as *group-delay function*. Detailed derivation procedures of group-delay function can be found in [1, 12]. In this work, the group-delay function is computed by the Time-Frequency Toolbox. [1]

To illustrate the effect of singing timbre on phase, in Figure 2 we show the spectral amplitudes, group-delay functions and phase profiles of three different examples (normal, vibrato and falsetto) of single vowel /ah/ sung by a tenor singer with the same fundamental frequencies C4. Since the fundamental frequencies of these three sounds are the same, the peaks in the amplitude spectra are mostly overlapped. At the spectral peak frequencies,

[1] http://tftb.nongnu.org/

the group-delay values of all three sounds are nearly zero. However, except for the spectral peaks, the group-delay function of falsetto sound is largely different from those of normal and vibrato sounds. The group-delay of falsetto is more peaky, possibly because of the occurrence of the transmission zeros (dips) evident from the magnitude spectrum, which mostly correspond to the spurious peaks seen in the group-delay function. We cannot observe this kind of spurious peaks from the magnitude spectra alone, due to the non-stationary nature of the signal under analysis (singing with vibrato), as well as the lack of sufficient frequency resolution. In contrast, group-delay functions better indicate the characteristics in frequency bands with small-energy.

2.2 Dictionary Learning and Sparse Coding

The atoms of a dictionary are typically learned from a large-scale training corpus. To overcome the difficulty of loading data into memory, the ODL algorithm is adopted [22]. ODL comes with a mini-batch mechanism that learns the dictionary incrementally by using a part of the training corpus in each update. Specifically, during the dictionary learning process, the atoms are updated for each input feature of the finite set of training signals $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_N]$ through the following joint optimization problem:

$$\begin{aligned} &\hat{\mathbf{D}} = \underset{\mathbf{D}\in\mathcal{C}}{\operatorname{argmin}} \tfrac{1}{N}\textstyle\sum_{i=1}^{N}\left(\tfrac{1}{2}\|\mathbf{y}_i - \mathbf{D}\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1\right), \\ &\forall j = 1, \ldots, k,\ \mathbf{d}_j^T\mathbf{d}_j \leq 1\,, \end{aligned} \tag{6}$$

where $\mathbf{y}_i \in \mathbb{R}^m$ is the i-th frame-level feature (column vector) of the input data (i.e., either $\log \mathrm{M}_x^h$ or Φ_x^h) from the training corpus, $\alpha_i \in \mathbb{R}^k$ is the codeword, $\mathbf{D} \in \mathbb{R}^{m\times k}$ is the dictionary, and the atom $\mathbf{d}_j$ is the jth column vector of $\mathbf{D}$. We can solve for $\mathbf{D}$ and α_i by minimizing one while keeping the other fixed [22]. The optimization of α involves a typical l_1-regularized sparse coding problem, which can be described as

$$\hat{\alpha_t} = \underset{\alpha_t}{\arg\min}\ \|\mathbf{x}_t - \mathbf{D}\alpha_t\|_2^2 + \lambda\|\alpha_t\|_1\,. \tag{7}$$

This problem has been well-studied in the machine learning and statistics fields, under different names such as the basis pursuit [4] or the lasso problem [28]. In this work, we use the open-source package SPAMS [2] and the LARS-lasso algorithm [6] for ODL and sparse coding, respectively. The parameter λ is set to $1/\sqrt{m}$ as in [6].

2.3 Source Separation

Given a monaural music signal, we first compute its $m \times n$ spectrogram $\mathbf{M}_x^h$ through STFT. Then, we separate the singing voice $\mathbf{E}$ (sparse components) from the music accompaniment $\mathbf{A}$ (low-rank components) by formulating the problem as the following RPCA problem,

$$\min_{\mathbf{A},\mathbf{E}:\ \mathbf{M}_x^h=\mathbf{A}+\mathbf{E}} \|\mathbf{A}\|_* + \mu\|\mathbf{E}\|_1\,, \tag{8}$$

[2] http://spams-devel.gforge.inria.fr/

Corpus	Original	Vocal	Accompaniment
USPOP	62.6%	65.9%	58.5%
USPOP2	61.7%	65.5%	56.6%
MIR-1K	55.9%	63.4%	53.4%
RWC	54.2%	59.9%	49.9%

Table 1. Average accuracy using log-magnitude spectrogram BOF features for various audio signals and training corpora. The vocal and accompaniment parts are separated by RPCA technique.

where $\|\cdot\|_*$ denotes the trace norm of a matrix (the sum of its singular values), $\|\cdot\|_1$ is the l_1 norm that denotes the sum of the absolute values of matrix entries, and μ is a positive weighting parameter that can be set to $1/\sqrt{\max(m,n)}$ as recommended in [3]. This algorithm is proven to be robust against gross errors and outliers, in comparison to its l_2-regularized counterpart, the well-known PCA algorithm. As Eq. (8) is convex, efficient algorithms such as accelerated proximal gradient (APG) and augmented Lagrange multipliers (ALM) [3, 20] can be employed to compute $\mathbf{A}$ and $\mathbf{E}$ in an iterative fashion. Open-source implementation of such solvers can be found from the Internet. [3]

3. EXPERIMENT

3.1 Dataset and Experimental Setup

We evaluate the performance of artist identification using the artist20 dataset [7], [4] which consists of six albums (1,413 songs in total) sung by 20 artists. For each song, a 30-second length audio signal with both vocal and music accompaniment is clipped for evaluation. The clips are sampled at 16 kHz. Most of the songs in the artist20 dataset can also be found in the uspop2002 dataset, [5] which contains over 7,000 Western Pop songs.

For computing the STFT, we use the Hanning window and try different values of the window size w (in terms of samples) and the hop factor h (the ratio of hop size and window size). For classifier training and testing, the l_2-regularized l_2-loss support vector classifier in LIBLINEAR [6] is employed for efficiency. To avoid album effect, a six-fold jack-knife cross-validation scheme is conducted. Each fold contains only one album from every artist. As there are many possible fold partitions, we perform the random partitions for ten times to get the average classification accuracy. Two-tailed t-test is also performed (over the ten six-fold partitions) to evaluate whether the performance difference between different methods or parameter settings is significant.

[3] http://perception.illinois.uiuc.edu/matrix-rank/

[4] http://labrosa.ee.columbia.edu/projects/artistid/

[5] http://labrosa.ee.columbia.edu/projects/musicsim/uspop2002.html

[6] http://www.csie.ntu.edu.tw/~cjlin/liblinear/

			w/ GD, w = 512			w/ GD, w = 1024			w/ GD, w = 2048		
SG		w/o GD	h = 0.1	h = 0.2	h = 0.5	h = 0.1	h = 0.2	h = 0.5	h = 0.1	h = 0.2	h = 0.5
w = 1024	h = 0.1	62.9%	63.3%	63.6%	64.0%	**64.5%**	64.3%	63.8%	**64.6%**	64.3%	63.8%
	h = 0.2	63.1%	62.1%	62.9%	64.2%	**64.4%**	**65.0%**	63.8%	**65.4%**	**64.7%**	64.3%
	h = 0.5	63.4%	58.1%	59.6%	62.1%	61.9%	62.5%	62.4%	**64.9%**	64.5%	63.9%

Table 2. Comparison of average accuracy among SG BOF features and fused SG + GD BOF features under various window sizes and hop factors. Vocal audio signal and MIR-1K training corpus are used. Data in bold style are those which achieve significant improvement in comparison to non-fused features under the same parameter settings.

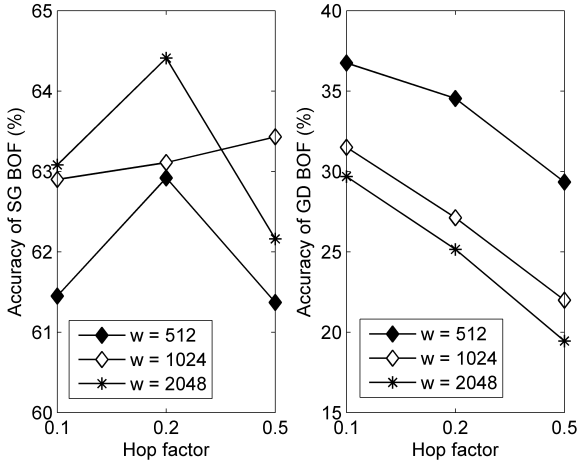


Figure 3. Average accuracy (in %) for SG BOF (left) and GD BOF (right) features constructed under various window sizes and hop factors. Vocal audio signal and MIR-1K training corpus are used.

3.2 Training Corpora

As discussed in Section 2, the size and the diversity of the training corpus are important to the generalizability of the learned dictionaries. We also have to pay attention to the possible overlap between the training corpus and the querying data (i.e., the dataset of the target classification problem; here artist20). To study the effect of training corpora, we compared the performance of artist identification using dictionaries learned from the following four training corpora: the whole uspop2002 dataset (USPOP), the uspop2002 dataset excluding overlap with artist20 (USPOP2), the MIR-1K dataset, and the whole vocal data in the RWC instrument dataset (RWC) [11]. MIR-1K contains 1,000 song clips extracted from 110 Chinese Pop songs released in karaoke format [14]; we use the vocal channel to train the dictionary. RWC vocal data contains five vowels (/a/, /i/, /u/, /e/, /o/) with various pitches and singing techniques sung by eight female and ten male singers, totaling more than 20,000 isolated notes. Among the four corpora, USPOP has the largest overlap with artist20. USPOP2 has no overlap but the music genre is the same (Western Pop). In contrast, the MIR-1K and RWC are considered more dissimilar from artist20.

As the first evaluation, we consider only BOF features computed from the log-magnitude spectrogram, using $w = 1024$ and $h = 0.5$. The size k of the dictionary is set to 1024. The evaluation result is shown in Table 1. The three columns of the table show the averaged accuracy using the original signal and the separated signals (vocal and accompaniment). By comparing the result of the four rows along the first column, we see that USPOP and USPOP2 lead to better accuracy comparing to MIR-1K and RWC. This is not surprising as the last two datasets are less similar with artist20. Although USPOP performs slightly better than USPOP2, the performance difference is not significant under the t-test.

3.3 Source Separation

By comparing the result of the three columns of Table 1, also it can be observed that using the separated vocal sound generally improves the classification accuracy, in comparison to using the original audio signals. In contrast, using the separated music accompaniment deteriorates the accuracy, which makes sense as the task emphasizes the vocal part of music. Two-tailed t-test shows that the improvement of Vocal over Original is significant ($p<0.001$, df=118). Moreover, when features are extracted from the vocal part, we see that the performance of using MIR-1K as the training corpus comes close to the case when USPOP is used (63.4% vs 65.9%).

To ensure the evaluation reported here is general enough, we use MIR-1K as the training corpus in the following experiments. Moreover, the features are computed from the separated vocal part of the song clips.

3.4 Incorporating Group Delay

Figure 3 shows the average accuracy of the BOF features computed from log-magnitude spectrogram (SG) and group-delay (GD) under various window sizes w and hop factors h. The sizes for the SG and GD dictionaries are both set to 1024. Note that we use different y-axes for the two features. As the figure shows, the performance of GD (20–35%) is generally much worse than the performance of SG (61–64%), possibly due to the highly noisy parts of the phase information. However, the result of GD is by no means random; the accuracies are significantly better than the random guess (whose accuracy is close to 5%). Moreover, we see a clear trend of the performance of GD with respect to w and h: better result is obtained by using smaller w and smaller h. In contrast, the performance of SG seems to be less sensitive to w and h.

We further experiment with the option of fusing the two types of features. The result is shown in Table 2, where we

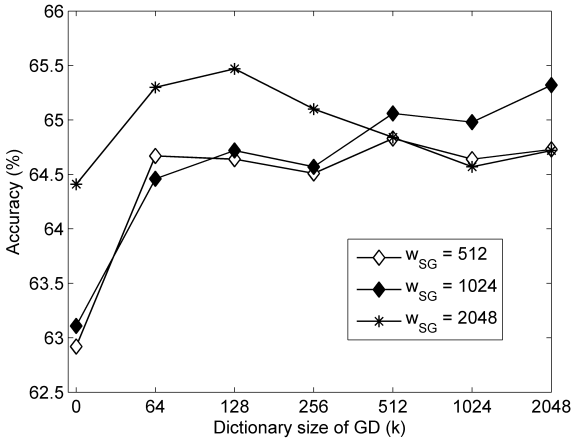


Figure 4. Average accuracies of different SG + GD BOF features by varying the dictionary size (k) and the window sizes of SG feature. Vocal audio signal and MIR-1K training corpus are used. As for GD feature, the window size is fixed to 1024. Hop factors for both SG and GD are 0.2.

compare the result without ('w/o') and with ('w/') using GD features with different values of w and h. We see the best result is obtained when SG (w=1024, h=0.2) and GD (w=2048, h=0.1) are fused, each using different values of w and h. Comparing to the case when GD is not used (i.e., the third column), the accuracy is improved from 63.1% to 65.4%, a significant improvement under the t-test ($p<0.05$, $df=118$). Actually, significant improvement is also observed for other cases, as indicated by the use of bold font weight in Table 2. This result shows that phase information is indeed useful for this task.

3.5 Influence of Dictionary Size

It has been shown that the performance of dictionary-based approach can be improved by using a larger dictionary [31]. As this is only verified on spectrogram-based features before, in this experiment we test the effects of the dictionary size k on the accuracy of artist identification using the group-delay features. Instead of using the GD features alone, we fuse it with SG features that is computed from a dictionary of size 1024, as this brings about better performance. Figure 4 shows the results as the dictionary size for GD ranges from 0 (no fusion) to 2048. When the window sizes w of SG are 512 or 1024, the performance gradually increases until reaching a plateau as we increase the dictionary size of GD. Generally speaking, using larger dictionary is also beneficial to phase-based features.

3.6 Comparison with the Existing Work

Finally, we compare our method with the GMM-based model proposed in [7], whose underlying frame-level feature representation is based on the early fusion of the classic MFCC and Chroma. Under a six-fold jack-knife cross-validation, the overall accuracy on artist20 was 56% for MFCC features and 59% for MFCC+Chroma features. Using the same fold partition and dictionaries learned from separated vocal parts of USPOP2, the proposed method reaches 66.0%, an improvement of 7%. Even if the less similar MIR-1K dataset is employed for dictionary learning, the classification accuracy reaches 65.5%.

4. DISCUSSION

In view of the source-filter model of voice, group-delay function contains the information on the phase-distortion behaviors of the vocal tract filter, which varies with the individual. Differently, the magnitude part contributes to the distribution of the resonance peaks on the time-frequency plane. Mathematically, the magnitude and the phase part of a STFT profile are strongly related, but they still have different characteristics [8]. Although the information provided by the magnitude part is prominent, the incorporation of group-delay usually better explains the characteristics of the target signal. There are many possibilities combining these two features, such as merging the two features at frame level [13], combining at BOF level, or in decision stage. Therefore, the way by which the features are combined and the relative weights of the two features are worthy for future study.

The use of singing voice separation generally improves the modeling of either the magnitude or phase information. However, the RPCA technique adopted in this work is not perfect. Under many cases the sparse counterpart of RPCA technique contains not only vocal sources but predominant instrument solo or the percussion sound. This issue might be partially solved by performing vocal detection first to exclude non-vocal segments.

5. CONCLUSION

In this paper, we have proposed a novel artist identification method based on sparse features learned from both the magnitude and phase parts of the spectrum. The features are computed from the separated vocal part of music signal using robust principal component analysis, in order to better model the characteristics of singing timbre. Our analysis shows that both singing voice separation and unsupervised feature learning are required steps for the features to be informative. Moreover, group-delay functions contain information complementing spectrogram. Evaluation on the artist20 dataset validates the effectiveness of the proposed phase feature. It is hoped that the present work can inspire more research towards the modeling of phase information of music signals, which might hold the promise of improving other MIR problems.

6. REFERENCES

[1] F. Auger and P. Flandrin. Improving the readability of time-frequency and time-scale representations by the method of reassigment. *IEEE Trans. Signal Processing*, 43(5):1068–1089, 1995.

[2] E. Benetos and Y. Stylianou. Auditory spectrum-based pitched instrument onset detection. *IEEE Trans. Audio, Speech, Language Process.*, 18(8):1968–1977, 2010.

[3] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *J. ACM*, 58(3):1–37, 2011.

[4] S. S. Chen, D. L. Donoho, Michael, and A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Scientific Computing*, 20:33–61, 1998.

[5] S. Dixon. Onset detection revisited. In *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx06)*, 2006.

[6] B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[7] D. Ellis. Classifying music audio with timbral and chroma features. In *ISMIR*, 2007.

[8] E. Chassande-Mottin F. Auger and P. Flandrin. On phase-magnitude relationships in the short-time fourier transform. *IEEE Signal Process. Lett*, 19(5):267–270, 2012.

[9] H. Fujihara, M. Goto, T. Kitahara, and H. 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. Audio, Speech, Language Process.*, 18(3):638–648, 2010.

[10] M. Goto. A robust predominant-f0 estimation method for real-time detection of melody and bass lines in cd recordings. In *IEEE ICASSP*, 2000.

[11] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Music genre database and musical instrument sound database. In *ISMIR*, 2003.

[12] S. Hainsworth, M. Macleod, S. W. Hainsworth, and M. D. Macleod. Time frequency reassignment: A review and analysis. Technical report, Cambridge University Engineering Department and Qinetiq, 2003.

[13] R. M. Hegde, H. A. Murthy, and V. R. R. Gadde. Significance of the modified group delay feature in speech recognition. *IEEE Trans. Audio, Speech, Language Process.*, 15(1):190–202, 2007.

[14] 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. Audio, Speech, Language Process.*, 18(2):310–319, 2010.

[15] P.-S. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *Proc. IEEE ICASSP*, 2012.

[16] S. Abdallah C. Duxbury M. Davies J. P. Bello, L. Daudet and M. B. Sandler. A tutorial on onset detection in music signals. *IEEE Trans. Speech, Audio Process.*, 13(5):1035–1047, 2005.

[17] A. Lacoste and D. Eck. A supervised classification algorithm for note onset detection. In *EURASIP Journal on Advances in Signal Processing*, 2007.

[18] M. Lagrange, A. Ozerov, and E. Vincent. Robust singer identification using melody enhancement and uncertainty learning. In *ISMIR*, 2012.

[19] M. Levy and M. Sandler. Music information retrieval using social tags and audio. *IEEE Trans. Multimedia*, 11(3):383–395, 2009.

[20] Z. Lin and et al. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. Technical Report UILU-ENG-09-2214, 2009.

[21] L. Liu, J. L. He, and G. Palm. Effects of phase on the perception of intervocalic stop consonants. *Speech Communication*, 22:403–417, 1997.

[22] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *ICML*, pages 689–696, 2009.

[23] I. McCowan, D. Dean, M. McLaren, R. Vogt, and S. Sridharan. The delta-phase spectrum with application to voice activity detection and speaker recognition. *IEEE Trans. Audio, Speech, Language Process.*, 19(7):2026–2038, 2011.

[24] A. Mesaros, T. Virtanen, and A. Klapuri. Singer identification in polyphonic music using vocal separation and pattern recognition methods. In *ISMIR*, 2007.

[25] T. L. Nwe and H. Li. Exploring vibrato-motivated acoustic features for singer identification. *IEEE Trans. Audio, Speech, Language Process.*, 15(2):519–530, 2007.

[26] K. K. Paliwal and L. D. Alsteris. Further intelligibility results from human listening tests using the short-time phase spectrum. *Speech Communication*, 48:727–736, 2006.

[27] L. Regnier and G. Peeters. Combining classification based on local and global features: application to singer identification. In *DAFx-11*, pages 127–134, Sep. 2011.

[28] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal Statistical Soc.*, 58:267–288, 1996.

[29] W.-H. Tsai and H.-P. Lin. Background music removal based on cepstrum transformation for popular singer identification. *IEEE Trans. Audio, Speech, Language Process.*, 19(5):1196–1205, 2011.

[30] Y.-H. Yang. On sparse and low-rank matrix decomposition for singing voice separation. In *ACM MM*, pages 757–760, 2012.

[31] C.-C. M. Yeh and Y.-H. Yang. Supervised dictionary learning for music genre classification. In *ACM ICMR*, 2012.

AUTOMATIC TRANSCRIPTION OF TURKISH MAKAM MUSIC

Emmanouil Benetos
City University London
emmanouil.benetos.1@city.ac.uk

Andre Holzapfel
Boğaziçi University
andre.holzapfel@boun.edu.tr

ABSTRACT

In this paper we propose an automatic system for transcribing makam music of Turkey. We document the specific traits of this music that deviate from properties that were targeted by transcription tools so far and we compile a dataset of makam recordings along with aligned microtonal ground-truth. An existing multi-pitch detection algorithm is adapted for transcribing music in 20 cent resolution, and the final transcription is centered around the tonic frequency of the recording. Evaluation metrics for transcribing microtonal music are utilized and results show that transcription of Turkish makam music in *e.g.* an interactive transcription software is feasible using the current state-of-the-art.

1. INTRODUCTION

The process of deriving a description for music in the form of a graphical representation is referred to as transcription in musicology. The practical use of this process lies usually in simplifying the analysis of mainly melodic, rhythmic and harmonic properties of a piece. The discussion of a proper graphical representation is almost as old as (comparative) musicology [2]. It can be stated as common sense that the goal of the analysis and traits of the music should guide the choices towards obtaining a transcription.

Automatic music transcription (AMT) is the process of automatically converting an acoustic music signal into some form of musical notation and is considered to be an open problem in the music information retrieval (MIR) literature, especially for transcribing multiple-instrument polyphonic music [5, 10]. The vast majority of AMT systems are targeted for transcribing well-tempered Eurogenetic [1] music and typically convert a recording into a piano-roll or a MIDI file (cf. [8] for a recent review of AMT systems). Evaluation of AMT systems is also typically made using a quarter tone tolerance, as in the MIREX Multiple-F0 Estimation and Note Tracking Tasks [1].

[1] Term used to avoid the misleading dichotomy of Western and non-Western music. It was proposed by Prof. Robert Reigle (MIAM, Istanbul) in personal communication.

In Turkish makam music, it is an open discussion which amount of steps should be used to divide an octave in order to be able to adequately describe pieces in all existent modes (*makams*). Academic theory adopted the division of the octave into 53 equal tempered intervals, which results in an interval of 22.642 cent, referred to a Holderian comma (see [6] for a concise description). A Western staff notation enhanced to represent such finer resolution (see Section 2) is used by musicians to practice and memorize pieces. Therefore, it appears to be reasonable to target a similar representation for a transcription system for Turkish makam music.

In this work, a system for transcribing Turkish makam music is proposed; we compile a dataset containing microtonal ground-truth and modify a state-of-the-art multi-pitch detection algorithm [4] to the specific challenges of Turkish makam music. This way we can obtain a first insight into the difficulty of the task, and gain insight into necessary future development steps. This includes the question if different music demands the design of a radically different approach, or if enhancing existing software can be an appropriate choice.

We will start by explaining the basic challenges that this music poses for AMT systems, and give our motivations to pursue this task for Turkish makam music in Section 2. We then describe the music collection that was compiled and used for evaluation, along with evaluation measures in Section 3. Section 4 describes the automatic transcription system, and experimental results are presented in Section 5. In the final Section we discuss our results and propose a strategy for an improved transcription system.

2. CHALLENGES AND MOTIVATION

The makam in Turkish music is a modal framework for melodic development. It includes the notion of a scale, proposes certain modulations to other makams, and describes characteristics of melodic progression. While a comprehensive explanation of these concepts is out of the focus of this paper, we are going to emphasize certain characteristics of Turkish makam music that make it a challenging repertoire for automatic transcription. Regarding the relation between music and notation, four traits deserve closer attention:

1. The notation of Turkish makam music uses a system that is based on Western staff notation, which introduces additional accidentals that signify a subset of the intervals obtained by dividing a whole tone into

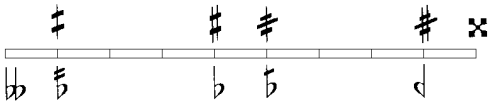


Figure 1. Visualization of the accidentals used in Turkish music. Only four of the possible eight intermediate steps that divide a whole tone are used.

nine equal steps (Holderian comma). This system comes with four additional flat and sharp accidentals respectively, see Fig. 1. However, as intensively discussed in music practice as well as in literature [6], this coarse grid does not properly reflect the intonation that is applied in practice. The knowledge about this correct intonation is part of the oral tradition.

2. Apart from the intonation of intervals, the fundamental frequency of a note depends on the transposition which a musician chooses. Only in one particular transposition the note a' is equal to $440Hz$, while there are a total of 12 transpositions to choose from.

3. A third particularity of Turkish music is that once a transposition is chosen, musicians in an ensemble choose an octave which suits the tonal range of their instrument. This can lead to up to three parallel melodies in octave intervals, while they are still considered to represent the (single) melody in the notation.

4. As fourth and final aspect, we have to emphasize the importance of augmenting the notes written in the score by various types of ornamentations and note additions. This is valid for pieces interpreted by a single musician, but also in the presence of several instruments. In the latter case, all musicians interpret a single melodic line, but are given a wide range of freedom to deviate from the simple representation in the score. This musical interaction is commonly referred to as *heterophony*.

These four traits of the relation between music and notation in Turkish makam music clarify that it poses some novel challenges to existing AMT systems. Current systems are usually focused on either polyphonic and monophonic sounds, and aim at a transcription into a representation with only 12 intervals per octave. The ability to extend systems beyond these specifications, and to tackle the above mentioned traits with an algorithmic approach can provide us with the means to transcribe performances of Turkish makam music. Such transcription is of great value for studying *e.g.* improvised performances of this music, or to study performance differences between different musicians.

It is worth to note that Turkish makam music is related to modal practice in many other cultures, in the sense that it shares the traits of microtonality and heterophonic performance. Its advantage for our studies is the availability of a large corpus of notation from more than 400 years. It therefore represents an ideal point of entry into the development of automatic transcription approaches for modal music throughout the world.

3. MUSIC COLLECTION AND EVALUATION METRICS

3.1 Music Collection

The main instrumental forms of Turkish makam music, besides the improvised *taksim*, are the *Peşrev* and *Saz Semaisi*. Many notations of such pieces are publicly available in collections such as the one presented in [9], which contains notation with microtonal information in a machine-readable format (referred to as SymbTr format).

As our goal is to transcribe performances that use the most popular Turkish instruments, we decided to organize our collection according to that. As can be seen in Table 1, we chose five performances that use *tanbur*, a plucked string instrument, and five performances that contain *ney*, a rim-blown reed flute. Apart from solo performances, we also picked six ensemble performances that include various instruments. The horizontal lines in Table 1 divide between groups that represent different recordings of the same composition. We restricted ourselves to pieces that are available in the SymbTr collection [9], in order to use those notations as a starting point for the note-to-note alignment between notation and performance, which is a necessary element of the evaluation procedure for a transcription system (see Section 3.2).

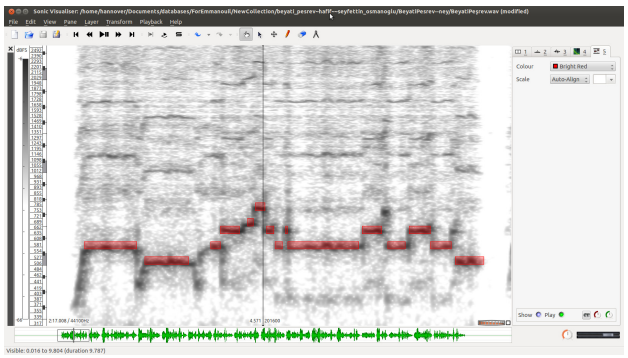


Figure 2. Screenshot of the manual alignment: the spectrogram for the first melodic phrase of the Beyati peşrev, with the aligned MIDI note events overlayed as red rectangles.

In order to obtain a note-to note alignment, we followed a semi-automatic approach. First, we determined if a performance deviates from the sequence of sections as found in the notation, and edited the SymbTr notation accordingly. This was necessary, as it is common to omit some units of a composition, or not to follow the repetitions as given in the notation. Then we converted the SymbTr representation to MIDI, and applied the approach presented in [11] in order to get a first estimate of the temporal alignment between the recording and the notes in the MIDI representation. The obtained aligned MIDI file was then loaded to the Sonic Visualiser [2] software as a notation layer

[2] http://www.sonicvisualiser.org/

on top of the spectrogram of the recording, and the alignment was corrected manually; see Fig. 2 for an illustration. It is important to point out that the notations used in the Turkish tradition only outline the basic structure of the melody. The pieces are meant to be livened up by the performers using embellishments and adding notes in a way that still respects the notes in the score. That means that in most cases, the notes found in the score are present in the performance, but possibly with changed durations due to the additions of the performer. Therefore, we restricted the manual correction mainly to aligning the notes in the MIDI layer with the performance, and did not annotate the added performance features. This means that we did not attempt to obtain a descriptive transcription of the performance [13], but rather target at the correct recognition of the outline given in the score in our evaluations.

The alignment results in a list of MIDI notes with the correct temporal alignment to the performance. As a final step, we obtain the microtonal information for each MIDI note from our edited SymbTr notation, which includes a resolution equal to the Holderian comma (53 equal tempered steps per octave). All note values are given in $cent$, normalized such that the tonic of the piece obtains a value of $0cents$. Which note represents the tonic in the notation is a given information when we assume the melodic mode (*makam*) to be known. This is a realistic assumption, as for most recordings the makam is an available metadata.

	Form	Makam	Instr.	Notes	Tonic/Hz
1	Peşrev	Beyati	Ensemble	906	125
2	Peşrev	Beyati	Ney	233	438
3	Saz S.	Hicazkar	Tanbur	706	147
4	Peşrev	Hüseyni	Ensemble	302	445
5	Peşrev	Hüseyni	Ensemble	614	124
6	Saz S.	Muhayyer	Ney	560	495
7	Saz S.	Muhayyer	Ensemble	837	294
8	Peşrev.	Rast	Tanbur	658	148
9	Peşrev	Rast	Ney	673	392
10	Peşrev.	Segah	Ney	379	541
11	Peşrev.	Segah	Ensemble	743	246
12	Saz S.	Segah	Ensemble	339	311
13	Saz S.	Segah	Tanbur	364	186
14	Saz S.	Uşşak	Tanbur	943	165
15	Saz S.	Uşşak	Tanbur	784	162
16	Saz S.	Uşşak	Ney	566	499

Table 1. Collection of recordings used for transcription.

Summing up, our ground truth annotation consists of a list of time instances and a note value in cent assigned to each time instance. In order to compare the output of our transcription system with such a ground truth, we need to estimate the tonic frequency of the recording and then normalize the estimated note values accordingly. We manually determined the tonic frequencies for all recordings, and list them along with other information of our collection in Table 1. We also include the automatic tonic detection approach proposed by Bozkurt [6] into our system, and we monitor how its errors affect the transcription. The listed tonic frequencies illustrate the wide range between tanbur and ney registers, which causes the playing of the melody in different octaves by these instruments (see the trait list in Section 2). Our ground truth annotation consists of a total of 9607 notes, distributed into 2411 for ney, 3455 for tanbur and 3741 for ensemble pieces.

3.2 Evaluation Metrics

For assessing the performance of Turkish music transcription systems, we propose a set of transcription metrics which are based on the metrics used for the MIREX Note Tracking evaluations [3]. In onset-based transcription evaluation of Eurogenetic music, a note event is assumed to be correct if its F0 is within 50 cent of the ground-truth pitch and its onset within a +/-50 or +/-100ms tolerance. Likewise for onset-offset evaluation, a note is considered correct if its offset time is also within 20% of the ground-truth note's duration around its offset value, or within a tolerance of the ground-truth notes offset.

For the proposed evaluations, we consider a note to be correct if its F0 is within a +/-20 cent tolerance around the ground-truth pitch and its onset is within a 100ms tolerance. The +/-20 cent and +/-100ms tolerance levels are considered to be "fair margins for an accurate transcription" according to Charles Seeger [13]. Thus, we define the following onset-based Precision, Recall, and F-measure:

$$\mathcal{P}_{ons} = \frac{N_{tp}}{N_{sys}}, \quad \mathcal{R}_{ons} = \frac{N_{tp}}{N_{ref}}, \quad \mathcal{F}_{ons} = \frac{2\mathcal{R}\mathcal{P}}{\mathcal{R}+\mathcal{P}} \qquad (1)$$

where N_{tp} is the number of correctly detected notes, N_{sys} the number of notes detected by the transcription system, and N_{ref} the number of reference notes. It should be noted that duplicate notes are considered as false alarms.

Even though onset-based evaluation is sufficient when transcribing recordings from pitched percussive instruments like the tanbur (where defining note offsets is generally an ill-defined problem), it can be useful performing onset-offset evaluation for pitched non-percussive instruments like ney. We thus consider a note event to be correct if, apart from the onset condition, its offset time is also within 20% of the ground-truth note's duration around its offset value, or within 100ms offset tolerance. We thus propose onset-offset Precision, Recall, and F-measure, denoted $\mathcal{P}_{off}$, $\mathcal{R}_{off}$, and $\mathcal{F}_{off}$, respectively, in the same way as in (1).

4. SYSTEM

The proposed transcription system takes as input a recording and information about the makam. Multi-pitch detection is performed using a shift-invariant model which was proposed in [4], modified for using ney and tanbur templates. Note tracking is performed as a post-processing step, followed by tonic detection. The final transcription output is a list of note events in cent scale centered around the tonic. A diagram of the proposed transcription system can be seen in Fig. 3.

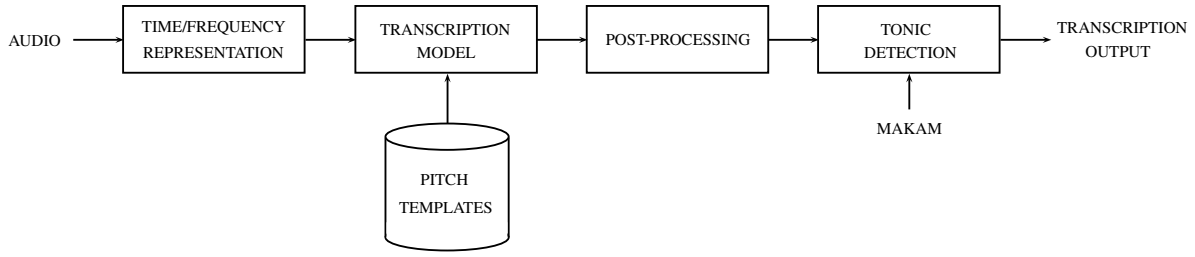


Figure 3. Proposed transcription system diagram.

4.1 Pitch Template Extraction

In systems for transcribing Eurogenetic music, pitch templates are typically extracted from isolated note samples (e.g. [7]). Since to the authors' knowledge such a database of isolated note samples for Turkish instruments does not exist, we extracted pitch templates using a dataset of 3 solo ney and 4 solo tanbur recordings.

Each note segment is identified and manually labeled, and the probabilistic latent component analysis (PLCA) method [14] with one component is employed per segment in order to extract a single spectral template per pitch. The time/frequency representation used is the constant-Q transform (CQT) with a spectral resolution of 60 bins/octave, with 27.5Hz as the lowest bin [12]. Since in log-frequency representations like the CQT the inter-harmonic spacings are constant for all pitches, templates for missing notes in the set were created by shifting the CQT spectra of neighboring notes. In the final collection of pitch templates, the range (in MIDI scale) for ney is 60-88 and the range for tanbur is 39-72.

4.2 Transcription Model

For performing multi-pitch detection, we employ the model of [4], which was originally developed for transcribing Eurogenetic music. The model expands PLCA techniques by supporting multiple templates per pitch and instrument source, as well as shift invariance over log-frequency; the latter is useful for performing multi-pitch detection in frequency resolution higher than MIDI scale.

The model takes as input a log-frequency spectrogram $V_{\omega,t}$ and approximates it as a joint distribution over time and log-frequency $P(\omega, t)$ (ω is the log-frequency index and t the time index). $P(\omega, t)$ is factored into $P(t)$ (spectrogram energy, which is a known quantity) and $P(\omega|t)$, which is modeled as:

$$P(\omega|t) = \sum_{p,s,f} P(\omega - f|s,p)P(f|p,t)P(s|p,t)P(p|t) \quad (2)$$

In (2), p is the pitch index in MIDI scale, s is the instrument source index, and f the pitch shifting factor (which accounts for frequency modulations or tuning deviations). Thus, $P(\omega|s,p)$ denotes the spectral template for instrument source s and pitch p, $P(f|p,t)$ is the time-varying log-frequency shift per pitch, $P(s|p,t)$ is the time-varying source contribution per pitch, and $P(p|t)$ the pitch activation over time. The shifting factor f is constrained to one semitone range, thus with a CQT resolution of 60 bins/octave f has a length of 5.

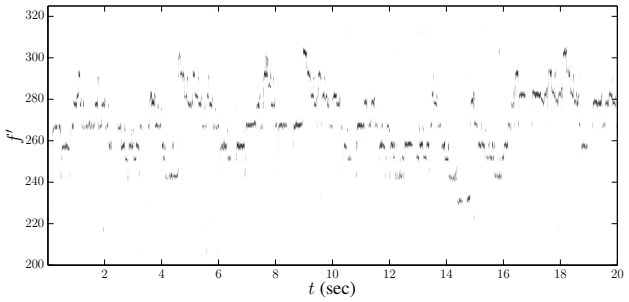


Figure 4. The time-pitch representation $P(f', t)$ for the 'Beyati Peşrev' piece performed by ney.

The unknown model parameters $P(f|p,t)$, $P(s|p,t)$, and $P(p|t)$ are estimated using iterative update rules based on the Expectation-Maximization algorithm, and can be found in [4]. The spectral templates $P(\omega|s,p)$ are kept fixed using the pre-extracted pitch templates from Section 4.1 and are not updated. The number of iterations is set to 15. The output of the transcription model is a pitch activation matrix and a shifting tensor which are respectively given by:

$$P(p,t) = P(t)P(p|t) \quad (3)$$
$$P(f,p,t) = P(t)P(p|t)P(f|p,t) \quad (4)$$

By stacking slices of $P(f,p,t)$ for all pitch values, a time-pitch representation with 20 cent resolution, useful for pitch content visualization, can be created:

$$P(f',t) = [P(f,p_{low},t)\cdots P(f,p_{high},t)] \quad (5)$$

where f' denotes pitch in 20 cent resolution, $p_{low} = 39$ is the lowest pitch value, and $p_{high} = 88$ the highest pitch value considered. In Fig. 4 the time-pitch representation for a ney piece can be seen.

4.3 Post-processing

The output of the transcription model of Section 4.2 is a non-binary pitch activation matrix which needs to be converted into a list of note events, listing onset, offset, and pitch. In order to achieve that, we first perform median filtering on $P(p,t)$ and then perform thresholding on the activation, followed by minimum duration pruning (with a minimum note event duration of 130ms), in the same way as in [7].

Since a significant portion of the dataset consists of ensemble pieces where the tanbur and ney are performing in octave unison, we need to convert that heterophonic output of the multi-pitch detection algorithm into a monophonic output which will be usable as a final transcription. Thus, a simple 'ensemble detector' is created by measuring the percentage of octave intervals in the detected transcription. If the percentage is above 20%, the piece is considered to be an ensemble one. Then, for each ensemble piece each octave interval is processed by merging the note event of the higher note with that of the lowest one.

In order to convert a detected note event into the cent scale, information from the pitch shifting factor f is used. For each detected event with pitch p and for each time frame, we find the value of f that maximizes $P(f, p, t)$:

$$\hat{f}_{p,t} = \arg\max_{f} P(f, p, t) \quad (6)$$

Then, the median of $\hat{f}_{p,t}$ for all time frames belonging to that note event is selected as the pitch shift that best represents that note event. Given the CQT resolution (60 bins/octave), the value in cent scale wrt the lowest frequency bin of the detected pitch is simply $20(\hat{f}-1)$, where $\hat{f}$ is the pitch index (in 20 cent scale) of the detected note.

4.4 Tonic detection

In order to detect the tonic frequency of the recording we apply the procedure described in [6]. It computes a histogram of the detected pitch values, and aligns it with a template histogram for a given makam using the cross-correlation function. Then the peak in the pitch histogram is assigned to the tonic which is closest to the peak of the tonic in the template, and all detected pitches are centered around this value. Finally, after centering the detected note events by the tonic, we eliminate note events that occur more than 1700 cents or less than -500 cents apart from the tonic, since such note ranges are rarely encountered in Turkish makam music.

5. RESULTS

Using the evaluation metrics defined in Section 3.2, transcription results using the proposed system with manually annotated tonic can be seen in Table 2, where the reported F-measure reaches 51.24% using 20 cent tolerance. It can be seen that the worst performance of $\mathcal{F}_{ons} = 47.35\%$ is reported for the subset of ensemble recordings, which is to be expected compared to the performance of the monophonic recordings. Also, results using automatically estimated tonic can be seen in Table 3, where there is a performance drop of about 10.5% in terms of F-measure. It should be noted that since the output of the transcription system is centered around the tonic and the F0 tolerance for the evaluation is 20 cent, even a slight tonic miscalculation might lead to a performance decrease. Using the dataset of Section 3, major tonic mis-estimations were observed for recordings 1, 5, and 10 (described in Table 1), leading to an F-measure for these recordings which is close to zero.

	$\mathcal{P}_{ons}$	$\mathcal{R}_{ons}$	$\mathcal{F}_{ons}$
Ney recordings	51.58%	52.67%	51.51%
Tanbur recordings	61.69%	49.66%	54.82%
Ensemble recordings	41.48%	56.28%	47.35%
All recordings	51.58%	52.85%	51.24%

Table 2. Transcription onset-based results using manually annotated tonic.

	$\mathcal{P}_{ons}$	$\mathcal{R}_{ons}$	$\mathcal{F}_{ons}$
Ney recordings	44.67%	43.98%	43.90%
Tanbur recordings	36.27%	45.13%	40.04%
Ensemble recordings	45.56%	33.21%	38.13%
All recordings	41.23%	42.07%	40.89%

Table 3. Transcription onset-based results using automatically detected tonic.

Regarding the impact of the F0 tolerance in the evaluation, Fig. 5 shows transcription results in F-measure using different F0 tolerance values, ranging from 10 cent to 50 cent. It can be seen that using 50 cent tolerance (*i.e.* the standard for evaluating Eurogenetic music transcription systems), the performance of the proposed system reaches $\mathcal{F}_{ons} = 61.58\%$. As far as the impact of the onset tolerance is concerned, Fig. 6 shows transcription results in F-measure using different onset tolerance values, ranging from 50ms to 150ms.

As far as onset-offset evaluation results using the ney recordings, the reported F-measure is 25.04%, while $\mathcal{P}_{off} = 25.12\%$ and $\mathcal{R}_{off} = 25.54\%$. Even though the onset-offset detection results might seem low, it should be stressed that estimating offsets (both manually and automatically) is an ill-defined problem and that similar results are reported in the MIREX Note Tracking evaluations [1].

The impact of several sub-components of the system can also be seen by disabling the 'ensemble detection' procedure, which leads to an F-measure of 45.08% for the ensemble pieces, which is more than a 2% decrease in performance. By removing the minimum duration pruning process (which was applied for detected note events less than 130ms), the reported F-measure with manually annotated tonic is 49.35%, which is a performance decrease of about 2%. Finally, by disabling the system sub-component which deletes note events that appear more than 1700 cents or less than -500 cents away from the tonic, the system performance drops to 47.42%; system decrease is more apparent in the ensemble pieces (which are performed in an octave unison, spanning a wider note range), leading to an F-measure of 40.60%.

6. DISCUSSION

Overall, it was shown that although the performance of automatic music transcription systems is still below that of a human expert, it is possible to produce fairly accurate transcriptions from Turkish music, which can be used as a basis for manually corrected scores. This can be verified

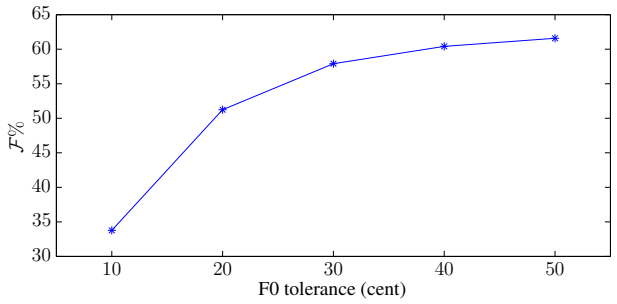


Figure 5. Onset-based transcription results in $\mathcal{F}$ with manually annotated tonic using different F0 tolerance.

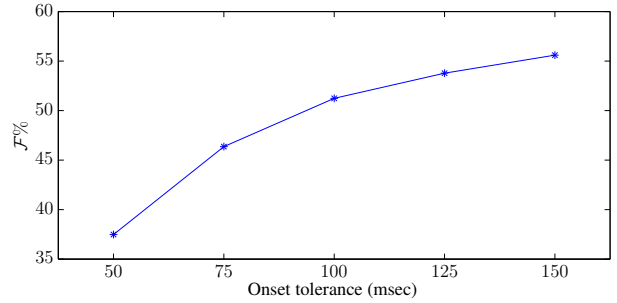


Figure 6. Onset-based transcription results in $\mathcal{F}$ with manually annotated tonic using different onset tolerance.

by comparing original recordings and MIDI transcriptions (without pitch bend), which can be found online [3] .

It was also shown that transcribing heterophonic music is an additional challenge for creating a system for transcribing Turkish makam music. On the one hand, one major drawback from automatic transcription systems is the fact that they produce octave errors, while on the other hand the multi-pitch output needs to be converted into a single melodic line for a makam score, which poses a different set of challenges.

One major issue with creating a spectrogram factorization-based transcription system for Turkish music is the lack of isolated note recordings for creating a training set; a set was created by segmenting some relatively clean solo recordings but we believe that an isolated sounds database would considerably improve the performance of the proposed system, especially in the tanbur case, where there are strong transient elements in tanbur tones. Another issue is the presence of percussion in makam pieces, which in certain cases resulted in false alarms in the transcriptions; again, a set of isolated percussive sounds could improve the performance of the proposed system. Apart from these template-related improvements, we intend to evaluate the system on vocal recordings and aim at including an improved tonic detection into the algorithm.

7. ACKNOWLEDGEMENTS

The authors would like to thank Barış Bozkurt for his advice and for providing us with software-tools. E. Benetos is supported by a City University London Research Fellowship. 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).

[3] http://www.soi.city.ac.uk/~sbbj660/MakamTranscriptionSamples.zip

8. REFERENCES

[1] Music Information Retrieval Evaluation eXchange (MIREX). `http://music-ir.org/mirexwiki/`.

[2] O. Abraham and E. M. von Hornbostel. Propositions for the transcription of exotic melodies, (in german language). In *Sonderdruck aus "Sammelbände der Internationalen Musikgesellschaft" XI, 1*. Leipzig, Internationale Musikgesellschaft, n.d., 1909.

[3] M. Bay, A. F. Ehmann, and J. S. Downie. Evaluation of multiple-F0 estimation and tracking systems. In *ISMIR*, pages 315–320, 2009.

[4] E. Benetos and S. Dixon. A shift-invariant latent variable model for automatic music transcription. *Computer Music Journal*, 36(4):81–94, 2012.

[5] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: breaking the glass ceiling. In *ISMIR*, pages 379–384, 2012.

[6] B. Bozkurt. An automatic pitch analysis method for turkish maqam music. *Journal of New Music Research*, 37(1):1–13, 2008.

[7] A. Dessein, A. Cont, and G. Lemaitre. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. In *ISMIR*, pages 489–494, 2010.

[8] P. Grosche, B. Schuller, M. Müller, and G. Rigoll. Automatic transcription of recorded music. *Acta Acustica united with Acustica*, 98(2):199–215, March 2012.

[9] K. Karaosmanoğlu. A turkish makam music symbolic database for music information retrieval: Symbtr. In *ISMIR*, 2012.

[10] A. Klapuri and M. Davy, editors. *Signal Processing Methods for Music Transcription*. New York, 2006.

[11] R. Macrae and S. Dixon. Accurate real-time windowed time warping. In *ISMIR*, pages 423–428, 2010.

[12] C. Schörkhuber and A. Klapuri. Constant-Q transform toolbox for music processing. In *7th Sound and Music Computing Conference*, Barcelona, Spain, July 2010.

[13] C. Seeger. Prescriptive and descriptive music-writing. *Music Quarterly*, 64(2):184–195, 1958.

[14] P. Smaragdis, B. Raj, and M. Shashanka. A probabilistic latent variable model for acoustic modeling. In *Neural Information Processing Systems Workshop*, Whistler, Canada, December 2006.

LOCAL GROUP DELAY BASED VIBRATO AND TREMOLO SUPPRESSION FOR ONSET DETECTION

Sebastian Böck and Gerhard Widmer
Department of Computational Perception
Johannes Kepler University, Linz, Austria
sebastian.boeck@jku.at

ABSTRACT

In this paper we present a new vibrato and tremolo suppression technique for onset detection. It weights the differences of the magnitude spectrogram used for the calculation of the spectral flux onset detection function on the basis of the local group delay information. With this weighting technique applied, the onset detection function is able to reliably distinguish between genuine onsets and spectral energy peaks originating from vibrato or tremolo present in the signal and lowers the number of false positive detections considerably. Especially in cases of music with numerous vibratos and tremolos (e.g. opera singing or string performances) the number of false positive detections can be reduced by up to 50% without missing any additional events. Performance is evaluated and compared to current state-of-the-art algorithms using three different datasets comprising mixed audio material (25,927 onsets), violin recordings (7,677 onsets) and solo voice recordings of operas (1,448 onsets).

1. INTRODUCTION AND RELATED WORK

Onset detection is the process of finding the starting points of all musically relevant events in an audio performance. While the detection of percussive onsets can be considered a solved problem, [1] softer onsets, vibrato and tremolo are still a major challenge for existing algorithms.

Soft onsets (e.g. bowed string or woodwind instruments) have a long attack phase with a slow rise in energy, thus energy or magnitude-based approaches are not the best fit to detect these sort of onsets. In the past, special algorithms have been proposed to solve the problem of soft onsets by incorporating (additionally) phase [3, 4, 10] or pitch information [9, 14, 15] or a combination thereof [12] to overcome the shortcomings of energy or magnitude-based onset detection algorithms. However, advances in magnitude-based methods [6] show that these methods are now on par with the before-mentioned methods but outperform them on all sorts of percussive audio material.

The current state-of-the-art methods for online [5] and offline [11] onset detection are based on a probabilistic model and incorporate a recurrent neural network with the spectral magnitude and its first time derivative as input features. Especially the offline variant *OnsetDetector* shows superior performance on all sorts of signals [1]. Because of its bidirectional architecture, it is able to model the context of an onset to both detect barely dicernible onsets in complex mixes and suppress events which are erroneously considered as onsets by other algorithms.

Vibrato is an artistic effect commonly used in classical music and can be sung or played by (mostly) string instruments. It reflects a periodic change of the played or sung frequency of the note. Vibrato is technically characterized by the amount of pitch variation (e.g. $\pm$ a semitone for string instruments and up to a complete tone in operas) and the frequency with which the pitch changes over time (e.g. 6 Hz). It is sometimes used synonymously as a combination with another effect: the tremolo, which describes the changes in volume of the note. Because it is technically hard for a human musician to play pure vibratos or tremolos, usually both effects are performed simultaneously. The resulting fluctuations in loudness and frequency make it very difficult for onset detection algorithms to distinguish correctly between new note onsets and an intended variation of the note.

So far only a few publications have addressed the problem of spuriously detected onsets music containing vibrato and tremolo. Collins [9] uses a vibrato suppression stage in his pitch-based onset detection method, which first identifies vibrato regions that fluctuate at most one semitone around the center frequency and collects the extrema in a list. The region is expanded gradually in time to cover the whole duration of the vibrato. After having identified the complete extent of the vibrato, all values within this window are replaced by the mean of the extrema list. The onset detection function is based on the concept of stable pitches and uses the change in pitches as cues for new onsets.

Schleusing et al. [14] deploy a system based on the inverse correlation of N consecutive spectral frames centered around the current location. Regions of stable pitch lead to low inverse correlation values, and pitch changes result in peaks in the detection function. To suppress vibrato they deploy a warp compensation which cancels out

[1] F-measure values > 0.95 as obtained with state-of-the-art onset detection algorithms [1] can be considered to have solved the problem.

small pitch changes within the considered window, leaving genuine onsets mostly untouched.

Recent research [7] applies a maximum filter to suppress vibrato in audio signals. This method operates in the spectral domain; specifically it only considers the magnitude spectrogram without incorporating any phase information. Like the common spectral flux algorithm [13] it relies on the detection of positive changes in the energy over time, but instead of calculating the difference between the same frequency bin for the current and previous frames, it includes a special magnitude trajectory tracking stage which is able to suppress spurious positive energy fragments.

Still, all algorithms (apart from those relying solely on phase information) suffer from loudness variations, which mostly originate from the tremolo effect. This paper addresses this problem by incorporating the phase – more specifically the local group delay (LGD) information – to determine steady tones and suppress the spurious loudness variations accordingly.

2. PROPOSED METHOD

Incorporating phase information is only feasible if each frequency bin of the spectrogram is considered separately as in the methods described in [3, 4, 10]. However, these methods have proven to perform poorly compared to current state-of-the-art algorithms [6]. Thus, our method is based on the recently proposed *SuperFlux* [7] algorithm, which is an enhanced version on the common spectral flux algorithm [13]. It is already significantly less sensitive to frequency variations caused by vibrato, but adding a special local group delay based weighting technique to the difference calculation step, makes this method even more robust against loudness variations of steady tones, e.g., those caused by tremolo.

2.1 SuperFlux

The system performs a frame-wise processing of the audio signal (sample rate 44,1 kHz). The signal is divided into overlapping chunks of length $N = 2048$ samples and each frame is weighted with a Hann window of the same length before being transformed to the spectral domain via the discrete Fourier transform (DFT). Two adjacent frames are located 220.5 samples apart, resulting in a resolution of 200 frames per second, which allows reportin of onsets to within 5 ms.

It has been found advantageous [6] to first filter the resulting magnitude spectrogram $|X(n,k)|$ (n denotes the frame number and k the frequency bin index) with a filterbank $F(k,m)$ (with m being the filter band number) before being processed further. The filterbank has $M = 138$ filters aligned equally on the logarithmic frequency scale with quarter-tone spacing. To better match the human perception of loudness, the resulting filtered spectrogram $X_F(n,m)$ is then transferred to a logarithmic magnitude scale, denoted $X_{L,F}(n,m)$ hereafter. Instead of calculating the bin-wise difference to the previous frame of the same logarithmic filtered spectrogram, a maximum filter along the frequency axis is applied (i.e. the value of a bin is set to the maximum of the same bin and its direct neighbors on the frequency axis) and the difference is calculated with respect to the μ-th previous frame of this maximum filtered spectrogram $X^{max}_{L,F}(n,m)$ resulting in the following equation for the difference calculation stage:

$$D(n,m) = X_{L,F}(n,m) - X^{max}_{L,F}(n-\mu,m) \quad (1)$$

The parameter μ depends on the frame rate f_r, which is set to 200 fps, resulting in $\mu = 2$ frames. The *SuperFlux* onset detection function is then defined as the sum of all positive differences:

$$SF(n) = \sum_{m=1}^{m=M} H\left(D(n,m)\right) \quad (2)$$

with $H(x) = \frac{x+|x|}{2}$ being the half-wave rectifier function.

The positive effect of these measures can be seen clearly in Figures 1a to 1c, which depict a 4 second recording of a violin played with vibrato and tremolo. However, there are still some spurious positive energy fragments left, which can be eliminated with the approach described in the next section. For a more detailed description of the *SuperFlux* algorithm, please refer to [7].

2.2 Local Group Delay based difference weighting

Using solely the magnitude information of the spectrogram enables onset detection algorithms to detect most onsets reliably, but also makes them susceptible to all kinds of loudness variations of steady tones. Using the phase as an additional source of information helps to lower the impact of these loudness variations. However, the main problem of incorporating the phase information is that it can only be combined easily with the magnitude spectrogram if all frequency bins of the STFT are considered individually. But since filtering the magnitude spectrogram with a filterbank (i.e. merging several frequency bins into a single one) previous to the difference calculation yields much better performance for almost all kinds of audio signals [6], the phase information of constituent frequency bins of a filter band have to be combined such that phase can be used in conjunction with the filtered spectrogram.

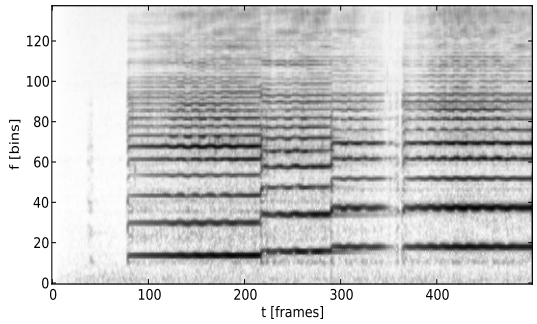
We investigated different approaches for combining the phase information of several frequency bands into one, and propose the following simple but effective solution. Given the phase ϕ of the complex spectrogram X by:

$$\phi(n,k) = \text{angle}\left(X(n,k)\right), \quad (3)$$

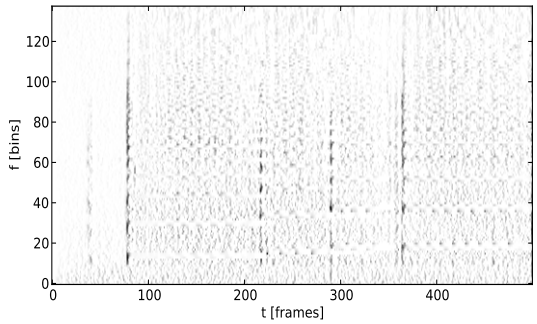
we can estimate the local group delay (LGD) of the spectrogram as:

$$LGD(n,k) = \phi^*(n,k) - \phi^*(n,k-1), \quad (4)$$

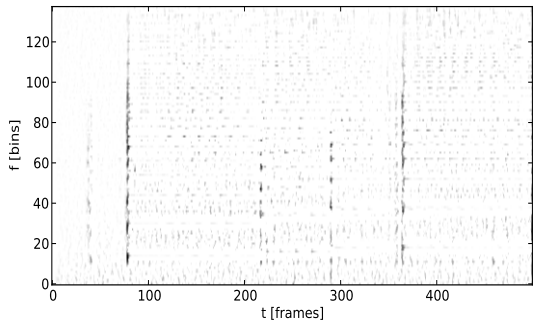
with ϕ^* defined as the 2π-unwrapped (over the frequency axis) phase. The local group delay gives information as where the gravitational centre of the magnitude is located. The spectrogram reassignment method [2] uses this information to gather a sharpened (reassigned) representation



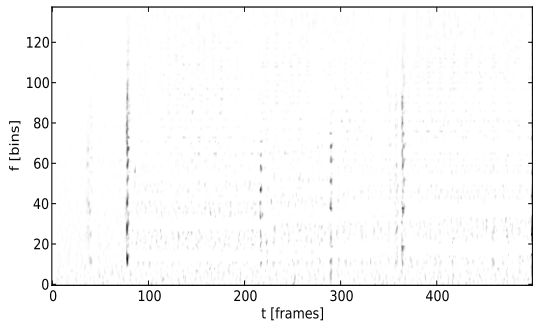
(a) magnitude spectrogram



(b) classical bin-wise positive difference



(c) maximum filtering trajectory tracking based difference



(d) local group delay based difference weighting

Figure 1: *(a) logarithmic magnitude spectrogram of a 5s violin played with vibrato and tremolo, (b) the positive differences calculated as in the spectral flux algorithm, (c) with applied maximum filtering as in [7] and (d) the proposed local group delay based difference weighting approach .*

of the magnitude spectrogram. Although this representation is more exact, the process leads to areas with lower magnitudes. The reassigned spectrogram looks a bit like a "scattered" version of the well known magnitude spectrogram. Thus, using this representation directly to calculate the spectral flux showed worse performance, mostly because of lots of smaller energy peaks, which we are trying to avoid.

Instead of using the local group delay information to relocate the magnitudes of the spectrogram, the information can be interpreted in a different way: regions with values close to zero indicate stable tones (or percussive sounds if they are aligned along the frequency axis) and regions with absolute values greater than zero indicate a possible onset. Holzapfel et al. [12] use the average of all local group delay values along the frequency axis as a feature for their onset detection function. Instead of averaging the individual values, we determine the local minimum within each band of the filterbank $F(k, m)$ for the *SuperFlux* calculation, and use these values as a weighting function.

Care has to be taken that the individual filters of the filterbank do not cover too many frequency bins, as the likelihood that there is a local group delay minimum that does not belong to any steady tone increases accordingly. Filterbanks with 24 filters per octave yielded good results for all kinds of music material. The higher the expected fluctuations in frequency, the lower should be the chosen number of filter bands. However, the fewer filter bands used, the wider the individual filter bands become, and in turn, this impacts the performance on percussive onsets. Percussive onsets have low local group delay values over a broad range of the frequency axis, thus applying the local minimum as a weighting would "erase" almost all percussive onsets.

To lower the impact of local group delay weighting on percussive sounds, we first apply a maximum filter over time which covers the range of 15 ms. For a frame rate of $f_r = 200$ fps, this equals to three frames and results in a temporal maximum filtered version of the LGD spectrogram:

$$LGD^*(n, k) = \max\left(|LGD(n-1:n+1, k)|\right) \quad (5)$$

After this first filtering step, we get the final local group delay based weighting by applying the previously described minimum filter, which sets the value of a bin to the local minimum of the region defined by the filter band:

$$W(n, m) = \min\left(LGD^*(n, k_{L(m)} : k_{U(m)})\right) \quad (6)$$

with $k_{L(m)}$ representing the lower frequency bin index of the filter band m of the filterbank $F(k, m)$, and $k_{U(m)}$ the upper bound respectively. This function is then used to weight the difference of the *SuperFlux* (cf. Equation 1), resulting in the modified detection function:

$$SF^*(n) = \sum_{m=1}^{m=M} H\left(D(n, m)\right) \cdot W(n, m) \quad (7)$$

with $H(x) = \frac{x+|x|}{2}$ being the half-wave rectifier function, n the frame number and m the frequency bin index. The '$\cdot$' operator denotes the element wise multiplication of the two matrices.

The effect of all proposed measures can be seen in Figure 1. Compared to the standard spectral flux implementation (1b), the difference with applied maximum filtering trajectory tracking (1c) already shows fewer positive energy components, which are further reduced by the proposed method, as can be seen in (1d). Figure 2 shows the sums of the positive differences. It is evident that the new approach lowers the overall noise in regions with vibrato and tremolo but keeps very sharp peaks at the onset positions.

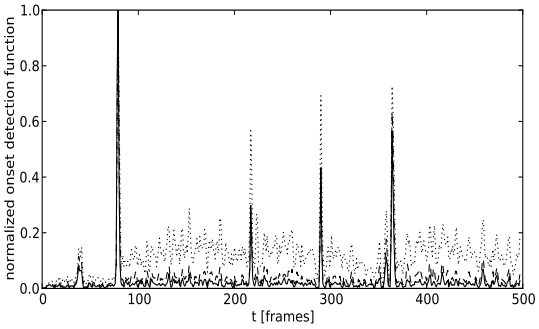


Figure 2: *Spectral flux sum of the differences shown in Figure 1. The simple filtered spectral flux is shown as dotted line, the SuperFlux as dashed line, and the proposed local group delay based difference weighting approach as solid line.*

It should be mentioned that the same weighting technique could be used for unfiltered magnitude spectrograms (i.e. the original spectral flux implementation). Instead of using the local maximum of all frequencies of a filter band, only the same frequency bin and its direct neighbors should be considered. Although the same positive impact on signals containing vibrato and tremolo can be observed, the overall performance compared to the filtered variants of the spectral flux (e.g. the *LogFiltSpecFlux* [6] or the *SuperFlux* [7]) is much lower, especially for polyphonic music.

2.3 Peak-picking

For selecting the final onsets of the weighted *SuperFlux* detection function we use the same peak-picking method as in [7]. Since the new onset detection function $SF^*(n)$ has a lower noise floor and shows sharper peaks than the original implementation (Equation 2), we had to alter the parameters for the peak-picking method used in [7]. A frame n of the onset detection function $SF^*(n)$ is selected as an onset if it fulfills the following three conditions:

1. $SF^*(n) = \max\left(SF^*(n-\omega_1 : n+\omega_2)\right)$
2. $SF^*(n) \geq \mathrm{m}ean(SF^*(n-\omega_3 : n+\omega_4)) + \delta$
3. $n - n_{previous\,onset} > \omega_5$

where δ is the tunable threshold. The other parameters were chosen to yield the best performance on the complete dataset. $\omega_1 = 30\,\mathrm{ms}$, $\omega_2 = 30\,\mathrm{ms}$, $\omega_3 = 100\,\mathrm{ms}$, $\omega_4 = 70\,\mathrm{ms}$ and the combination width parameter $\omega_5 = 30\,\mathrm{ms}$ showed good overall results. Parameter values must be converted to frames depending on the frame-rate f_r used.

3. EVALUATION

For the evaluation of the algorithm, different datasets and settings have been used to allow highest comparability with previous publications.

3.1 Performance measures and evaluation settings

For evaluating the performance of onset detection methods, commonly Precision, Recall, and F-measure are used. If a detected onset is within the evaluation window around an annotated ground truth onset location, it is considered a correctly identified onset. But every detected onset can only match once, thus any detected onset within the evaluation window of two different annotated onsets counts as one true positive and one false negative (a missed onset). The same applies to annotations, i.e. all additionally reported onsets within the evaluation window of an annotation are counted as false positive detections. In order to keep the comparability with other results, we match the evaluation parameters as follows:

Our standard setting is the one used in [6], which combines all annotated onsets within 30 ms to a single onset and uses an evaluation window of $\pm$ 25 ms to identify correctly detected onsets. Thus the combination width parameter ω_5 of our peak-picking method is set to 30 ms as well.

The second set of parameters (denoted with an asterisk in Table 1) uses the same settings as in [14], where all onsets within 50 ms are combined (i.e. $\omega_5 = 50\,\mathrm{ms}$) and an evaluation window of $\pm$ 70 ms is used.

Unless otherwise noted, all given results are obtained by swiping the threshold parameter δ of the peak-picking stage and choosing the value that maximizes the F-measure on the respective dataset.

3.2 Datasets

For comparison with the former state-of-the-art algorithm for pitched non-percussive music, the dataset from [14] is used. Unfortunately not all sound files and annotations could be used for evaluation, since the authors were only able to provide part of this set. Still, we believe that the achieved results are comparable, because the dataset has over three quarters of the size of the original dataset (7,677 instead of 9,717 onsets) and an identical distribution of the different playing styles (50% contain vibrato, some staccato etc.). This will be called the *Wang* dataset.

To show the ability to suppress tremolo and vibrato present in sung opera vocals, a second dataset introduced in [7] and consisting of solo singing rehearsal recordings of a Haydn opera is used. The set covers both male and female singers and has a total length of 10 minutes containing 1,448 onsets. It is called the *Opera* dataset.

The biggest dataset used for evaluation is that described in [6], which consists mostly of mixed audio material covering different types of musical genres performed on various instruments. It includes the sets used in [3], [12], and [11]. The 321 files have a total length of approximately 102 minutes and have 27,774 annotated onsets (25,927 if all onsets within 30 ms are combined). The main purpose of this set is to show how the new local group delay weighting for the *SuperFlux* algorithm impacts the performance on a general purpose dataset. This dataset is named *Böck*. Based on this set, we build a subset that contain violin and cello recordings played with vibrato and tremolo, but also feature accompaniment instruments. These 16 files have 849 onsets.

3.3 Results & Discussion

Because the local group delay weighting technique is designed especially for audio signals containing mostly vibrato and tremolo, the main focus should be put on the results obtained on the *Wang* and *Opera* datasets. But since we expect that it does not harm the overall performance of the underlying *SuperFlux* algorithm too much when used on other musical signals, the results given on the general purpose *Böck* dataset should not be neglected.

3.3.1 Competitors

Besides the former state-of-the-art algorithm for pitched non-percussive music presented in [14] (for comparison on the *Wang* dataset), we chose the winning submissions of last year's MIREX evaluation [1] for comparison. We consider these submissions to be state-of-the-art, since they achieved the highest F-measure ever measured during the MIREX evaluation.

The *OnsetDetector.2012* is an improved version of the method originally proposed in [11], which shows superior performance in offline scenarios, and represents the group of probabilistic onset detection approaches. Since the *OnsetDetector.2012* was trained on the *Böck* dataset, the results given in Table 3 and 4 for this algorithm were obtained with 8-fold cross-validation and parameters selected solely on the training set. Instead of the *LogFiltSpecFlux* [6] algorithm, we chose the recently proposed *SuperFlux* algorithm [7], which shows better performance on all datasets. The *SuperFlux* algorithm does not use any probabilistic information and thus has much lower computational demands, marking the current upper bound of performance of so-called "simple" algorithms.

Because the onset detection functions of the compared methods show very different shapes and characteristics, and the choice of peak-picking methods and parameters highly influence the final results, we use offline peak-picking only. Since all algorithms yield their best performance in offline mode and are less sensitive to variations of parameters, we consider this a valid choice. Nonetheless, all algorithms can be used in online mode with slightly lower performance.

3.3.2 Wang set

Table 1 shows the performance on violin music for the *Wang* dataset. The new local group delay weighted *SuperFlux* method outperforms all other algorithms with respect to false positive detections by at least 25%. Compared side-by-side with the current state-of-the-art onset detection algorithm, the *OnsetDetctor*, the weighted *SuperFlux* is able to achieve the same level of true positive detections, but improves regarding false positive detections by an impressive 56%.

	TP	FP
OnsetDetector.2012 [11] *	96.5%	15.5%
Schleusing et.al. [14] *	91.2%	9.2%
SuperFlux [7] *	94.7%	9.1%
SuperFlux w/ LGD weighting *	97.0%	6.8%

Table 1: *True and false positive rates of different onset detection algorithms on the Wang dataset. Results for Schleusing's algorithms were taken from [14]. Asterisks mark the evaluation method used in [14].*

Since the recordings in the *Wang* dataset are exclusively solo recordings made in a sound absorbing room and contain only very few polyphonic parts, this result can be seen as the maximum possible performance boost that can be obtained with the local group delay weighting method for this type of music.

3.3.3 Opera set

On the *Opera* dataset with male and female opera rehearsal recordings, the new method also shows its strength and is able to dramatically lower the number of false positive detections. Compared with the original *SuperFlux* implementation, the number of false detections go down from 450 to 221 (which is a reduction by 51%), if the new local group delay based weighting technique is applied. The new method even outperforms the current best-performing probabilistic approach (with respect to F-measure), but it should be noted that the neural network based method was not trained on any opera material.

	P	R	F
OnsetDetector.2012 [11]	0.576	0.777	0.662
SuperFlux [7]	0.672	0.635	0.653
SuperFlux w/ LGD weighting	0.806	0.635	0.711

Table 2: *Precision, Recall and F-measure of different onset detection algorithms on the Opera dataset.*

3.3.4 Böck set

In Table 3 results for the full *Böck* dataset are given. With the new difference weighting scheme, slightly lower performance can be observed. This was expected, since the new approach is tuned specifically towards music with vibrato and tremolo but which otherwise contains only very

few percussive sounds (as present in complex audio mixes like pop songs). It could be argued, that the impressive performance gains achievable for this special type of music justify the small performance penalty on this dataset.

	P	R	F
OnsetDetector.2012 [11]	0.892	0.855	0.873
SuperFlux [7]	0.883	0.793	0.836
SuperFlux w/ LGD weighting	0.873	0.778	0.823

Table 3: *Precision, Recall and F-measure of different onset detection algorithms on the Böck dataset.*

More interesting are the results given in Table 4 for the *strings* subset, which includes pieces with string instrumentation that also feature accompaniment instruments – which make vibrato and tremolo suppression harder. As can be seen, the local group delay weighted *SuperFlux* method also performs slightly worse than the original *SuperFlux* implementation. Thus, it must be concluded that the new weighting scheme is mainly suited for signals which feature numerous vibratos and tremolos but do not contain many other instruments.

	P	R	F
OnsetDetector.2012 [11]	0.834	0.820	0.827
SuperFlux [7]	0.836	0.701	0.762
SuperFlux w/ LGD weighting	0.777	0.710	0.742

Table 4: *Precision, Recall and F-measure of different onset detection algorithms on the strings subset of the Böck dataset using the same parameters as used for the results in Table 3.*

4. CONCLUSIONS

In this paper a new method for vibrato and tremolo suppression with local group delay based spectral weighting was presented. The new weighting scheme can be applied to any spectral flux like onset detection method and is able to reduce the number of false positive detections originating from vibrato and tremolo by up to 50% compared to current state-of-the-art implementations.

For future versions of this weighting technique, the Constant-Q transform could be investigated. Using this transform instead of the Short-Time Fourier Transform would make both the use of a filterbank for the magnitude spectrogram and the rather simple combination technique for the phase information of several frequency bins into one obsolete, but retain the beneficial behavior of this approach.

5. ACKNOWLEDGMENTS

This work is supported by the European Union Seventh Framework Programme FP7 / 2007-2013 through the PHENICX project (grant agreement no. 601166).

6. REFERENCES

[1] MIREX 2012 onset detection results. `http://nema.lis.illinois.edu/nema_out/mirex2012/results/aod/`, 2012, accessed 2013-03-27.

[2] F. Auger and P. Flandrin. Improving the readability of time-frequency and time-scale representations by the reassignment method. *IEEE Transactions on Signal Processing*, 43(5):1068–1089, May 1995.

[3] J.P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047, September 2005.

[4] J.P. Bello, C. Duxbury, M. Davies, and M. Sandler. On the use of phase and energy for musical onset detection in the complex domain. *IEEE Signal Processing Letters*, 11(6):553–556, June 2004.

[5] S. Böck, A. Arzt, F. Krebs, and M. Schedl. Online real-time onset detection with recurrent neural networks. In *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx-12)*, York, UK, September 2012.

[6] S. Böck, F. Krebs, and M. Schedl. Evaluating the online capabilities of onset detection methods. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR 2012)*, pages 49–54, Porto, Portugal, October 2012.

[7] S. Böck and G. Widmer. Maximum filter vibrato suppression for onset detection. In *Proceedings of the 16th International Conference on Digital Audio Effects (DAFx-13)*, Maynooth, Ireland, September 2013.

[8] N. Collins. A comparison of sound onset detection algorithms with emphasis on psychoacoustically motivated detection functions. In *Proceedings of the AES Convention 118*, pages 28–31, Barcelona, Spain, May 2005.

[9] N. Collins. Using a pitch detector for onset detection. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR 2005)*, London, UK, September 2005.

[10] S. Dixon. Onset detection revisited. In *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-06)*, pages 133–137, Montreal, Quebec, Canada, September 2006.

[11] F. Eyben, S. Böck, B. Schuller, and A. Graves. Universal onset detection with bidirectional long short-term memory neural networks. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 589–594, Utrecht, Netherlands, August 2010.

[12] A. Holzapfel, Y. Stylianou, A.C. Gedik, and B. Bozkurt. Three dimensions of pitched instrument onset detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1517–1527, August 2010.

[13] P. Masri. *Computer Modeling of Sound for Transformation and Synthesis of Musical Signals*. PhD thesis, University of Bristol, UK, December 1996.

[14] O. Schleusing, B. Zhang, and Y. Wang. Onset detection in pitched non-percussive music using warping-compensated correlation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2008)*, pages 117–120, April 2008.

[15] R. Zhou, M. Mattavelli, and G. Zoia. Music onset detection based on resonator time frequency image. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(8):1685–1695, November 2008.

Oral Session 5:

Source Identification and Separation

BEYOND NMF: TIME-DOMAIN AUDIO SOURCE SEPARATION WITHOUT PHASE RECONSTRUCTION

Kazuyoshi Yoshii[1] **Ryota Tomioka**[2] **Daichi Mochihashi**[3] **Masataka Goto**[1]
[1]National Institute of Advanced Industrial Science and Technology (AIST)
[2]The University of Tokyo [3]The Institute of Statistical Mathematics (ISM)
{k.yoshii, m.goto}@aist.go.jp tomioka@mist.i.u-tokyo.ac.jp daichi@ism.ac.jp

ABSTRACT

This paper presents a new fundamental technique for source separation of single-channel audio signals. Although nonnegative matrix factorization (NMF) has recently become very popular for music source separation, it deals only with the amplitude or power of the spectrogram of a given mixture signal and completely discards the phase. The component spectrograms are typically estimated using a Wiener filter that reuses the phase of the mixture spectrogram, but such rough phase reconstruction makes it hard to recover high-quality source signals because the estimated spectrograms are inconsistent, *i.e.*, they do not correspond to any real time-domain signals. To avoid the frequency-domain phase reconstruction, we use *positive semidefinite tensor factorization* (PSDTF) for directly estimating source signals from the mixture signal in the time domain. Since PSDTF is a natural extension of NMF, an efficient multiplicative update algorithm for PSDTF can be derived. Experimental results show that PSDTF outperforms conventional NMF variants in terms of source separation quality.

1. INTRODUCTION

Source separation of music audio signals is a fundamental task for music information retrieval (MIR). High-quality source separation could help users find their favorite songs according to the content (such as vocals or instruments) [1]. It would also let them enjoy *active music listening* [2] based on the remixing of existing instrumental parts [1–4].

Nonnegative matrix factorization (NMF) [5] has recently played a key role in the source separation of single-channel audio signals. It can approximate a nonnegative matrix (the amplitude or power spectrogram of a given mixture signal) as the product of two nonnegative matrices— a set of basis spectra and a set of the corresponding activations. Then the complex spectrogram of the mixture signal is decomposed into a sum of source spectrograms by using a Wiener filter that simply reuses the original phase. However, we cannot recover high-quality source signals from the decomposed spectrograms having the unreal phase.

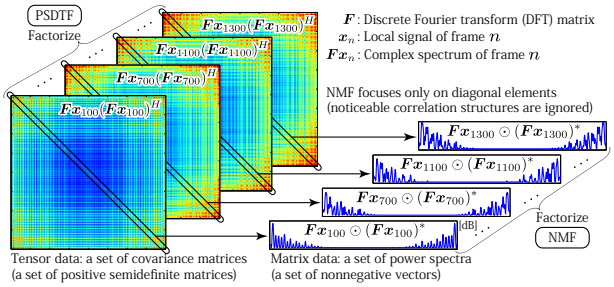


Figure 1. PSDTF is a natural extension of NMF.

Considerable effort has been devoted to estimating *consistent* complex spectrograms that correspond to real time-domain signals. To reconstruct the phase of a given amplitude spectrogram, Griffin and Lim [6] proposed an iterative short-time Fourier transform (STFT) method that estimates a time-domain signal such that its amplitude spectrogram is closest to the given spectrogram. Le Roux *et al.* [7] proposed a cost function that evaluates the inconsistency of a complex spectrogram and derived an efficient algorithm for minimizing the cost function [8]. Kameoka *et al.* [9], on the other hand, formulated complex NMF for directly factorizing a complex spectrogram. The cost function evaluating the inconsistency could be integrated into complex NMF as suggested in [10]. Note that improved consistency does not always result in improved sound quality.

To circumvent the phase reconstruction, we use *positive semidefinite tensor factorization* (PSDTF) [11] for time-domain separation of single-channel audio signals. Instead of explicitly considering the wave shapes and phases of basis signals, we focus on the statistical characteristics (*e.g.*, periodicity and whiteness) of those signals. More specifically, we assume that each basis signal follows a Gaussian process (GP) having a stationary kernel. A given mixture signal consisting of multiple basis signals is thus locally modeled by a GP with a convex combination of the corresponding kernels. These kernels can be estimated from a set of local covariances of the mixture signal by using a multiplicative update algorithm.

We can show that the time-domain PSDTF has an equivalent frequency-domain representation used for factorizing a mixture spectrogram like NMF. As shown in Figure 1, PSDTF deals with a set of Hermitian positive semidefinite matrices (outer products of complex spectra) for considering the correlations between frequency components. This is reasonable because the discrete Fourier transform (DFT)

cannot perfectly decorrelate the frequency components of the mixture signal. On the other hand, NMF focus only on a set of the nonnegative diagonal elements of those matrices (power spectra) by discarding the correlations between frequency components. This indicates that PSDTF is a natural and elegant extension of NMF.

2. FREQUENCY-DOMAIN SOURCE SEPARATION

This section aims to reveal the theoretical basis underlying nonnegative matrix factorization (NMF) in the context of source separation. We review two popular variants of NMF called KL-NMF [12] and IS-NMF [13] and how these variants can be used for source separation.

2.1 Nonnegative Matrix Factorization

The goal of NMF is to approximate a nonnegative matrix $\boldsymbol{X} \in \mathbb{R}^{M \times N}$ as the product of two nonnegative matrices $\boldsymbol{W} \in \mathbb{R}^{M \times K}$ and $\boldsymbol{H} \in \mathbb{R}^{K \times N}$ as follows:

$$\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H} \stackrel{\text{def}}{=} \boldsymbol{Y}, \tag{1}$$

where $\boldsymbol{W}$ and $\boldsymbol{H}$ respectively represent a set of basis vectors and a set of the corresponding weight vectors, $K \ll \min(M, N)$ is the number of basis vectors, and $\boldsymbol{Y} \in \mathbb{R}^{M \times N}$ represents a reconstruction matrix. Eq. (1) can be rewritten in an element-wise manner as follows:

$$X_{mn} \approx \sum_{k=1}^{K} W_{mk} H_{kn} \stackrel{\text{def}}{=} Y_{mn}. \tag{2}$$

We here let $Y_{mn}^k = W_{mk} H_{kn}$ be a component reconstruction such that $Y_{mn} = \sum_k Y_{mn}^k$. A popular way to evaluate the reconstruction error $\mathcal{C}(X_{mn}|Y_{mn})$ between X_{mn} and Y_{mn} is to use the Bregman divergence [14] defined as

$$\begin{aligned}&\mathcal{C}_\phi(X_{mn}|Y_{mn})\\ &= \phi(X_{mn}) - \phi(Y_{mn}) - \phi'(Y_{mn})(X_{mn} - Y_{mn}),\end{aligned} \tag{3}$$

where ϕ is a strictly convex function. This divergence is no less than zero and is zero only when $X_{mn} = Y_{mn}$. The Kullback-Leibler (KL) divergence ($\phi(x) = x \log x - x$) and the Itakura-Saito (IS) divergence ($\phi(x) = -\log x$) are well-known special cases of the Bregman divergence. To estimate $\boldsymbol{W}$ and $\boldsymbol{H}$ such that the cost function $\mathcal{C}_\phi(\boldsymbol{X}|\boldsymbol{Y}) = \sum_{mn} \mathcal{C}_\phi(X_{mn}|Y_{mn})$ is minimized, we can use an efficient multiplicative update (MU) algorithm [15].

2.2 Application to Source Separation

The goal of source separation is to decompose a given mixture signal into the sum of K source signals. NMF enables us to perform source separation in the frequency domain. We regard the *nonnegative* spectrogram of the mixture signal as an $\boldsymbol{X}$ for which M is the number of frequency bins and N is the number of frames. We then factorize the given spectrogram $\boldsymbol{X}$ as $\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H}$, where $\boldsymbol{W}$ and $\boldsymbol{H}$ respectively represent a set of basis *nonnegative* spectra and a set of the corresponding temporal activations.

A probabilistic formulation of NMF enables us to estimate latent source signals. Let $\boldsymbol{S} \in \mathbb{C}^{M \times N}$ be the complex spectrogram of the mixture signal and $\boldsymbol{S}^k \in \mathbb{C}^{M \times N}$ be that of the k-th source signal. If the mixture signal is an instantaneous mixture of K source signals, we can say

$$\boldsymbol{S} = \sum_{k=1}^{K} \boldsymbol{S}^k. \tag{4}$$

Given the mixture spectrogram $\boldsymbol{S}$ (observed variable), each component spectrogram $\boldsymbol{S}^k$ (latent variable) can be estimated in a probabilistic manner as follows:

$$\mathbb{E}[S_{mn}^k|S_{mn}] = \frac{Y_{mn}^k}{Y_{mn}} S_{mn} = \frac{W_{mk}H_{kn}}{\sum_k W_{mk}H_{kn}} S_{mn}. \tag{5}$$

Eq. (5) is known as Wiener filtering in which the original phase of $\boldsymbol{S}$ is directly attached to each $\boldsymbol{S}^k$. The real-valued source signal can then be recovered from $\mathbb{E}[\boldsymbol{S}^k|\boldsymbol{S}]$ by using the overlap-add synthesis method [16]. Note that the complex spectrogram of the resulting source signal is unlikely to be equal to $\mathbb{E}[\boldsymbol{S}^k|\boldsymbol{S}]$ because in general $\mathbb{E}[\boldsymbol{S}^k|\boldsymbol{S}]$ is an *inconsistent* spectrogram that does not correspond to any actual time-domain signals.

2.3 Source Separation based on KL-NMF

KL-NMF is used for factorizing the *amplitude* spectrogram [12], *i.e.*, $X_{mn} = |S_{mn}|$. The cost function is given by $\mathcal{C}_{\text{KL}}(X_{mn}|Y_{mn}) = X_{mn} \log \frac{X_{mn}}{Y_{mn}} - X_{mn} + Y_{mn}$. Note that KL-NMF is not theoretically justified for source separation because the cost function is scale-variant, *i.e.*, $\mathcal{C}_{\text{KL}}(\boldsymbol{X}|\boldsymbol{Y}) \neq \mathcal{C}_{\text{KL}}(\alpha\boldsymbol{X}|\alpha\boldsymbol{Y})$ for a positive number α.

The probabilistic model of KL-NMF can be formulated by assuming that each latent component $|S_{mn}^k|$ is Poisson distributed with a mean parameter Y_{mn}^k as follows:

$$|S_{mn}^k| \big| Y_{mn}^k \sim \text{Poisson}(Y_{mn}^k). \tag{6}$$

We here assume that the condition for amplitude additivity is satisfied, *i.e.*, that the phase of each $\boldsymbol{S}^k$ is equal to that of $\boldsymbol{S}$. Eq. (4) can then be written as $|S_{mn}| = \sum_k |S_{mn}^k|$. Using $X_{mn} = |S_{mn}|$ and $Y_{mn} = \sum_k Y_{mn}^k$, the reproducing property of the Poisson distribution gives

$$X_{mn} \big| Y_{mn} \sim \text{Poisson}(Y_{mn}). \tag{7}$$

This probabilistic model based on superimposed Poisson variables $\{|S_{mn}^k|\}_{k=1}^K$ satisfying $|S_{mn}| = \sum_k |S_{mn}^k|$ enables us to calculate the expectation of each latent variable $|S_{mn}^k|$ in a principled manner as follows:

$$\mathbb{E}[|S_{mn}^k| \big| |S_{mn}|] = Y_{mn}^k Y_{mn}^{-1} |S_{mn}|. \tag{8}$$

Since the phase is assumed to be preserved, we get Eq. (5).

2.4 Source Separation based on IS-NMF

IS-NMF is used for factorizing the *power* spectrogram [13], *i.e.*, $X_{mn} = |S_{mn}|^2$. The cost function is $\mathcal{C}_{\text{IS}}(X_{mn}|Y_{mn}) = \frac{X_{mn}}{Y_{mn}} - \log \frac{X_{mn}}{Y_{mn}} - 1$. IS-NMF is suitable for source separation because the cost function is scale-invariant.

The probabilistic model of IS-NMF can be formulated by assuming that each latent component S_{mn}^k is complex Gaussian distributed with a variance Y_{mn}^k as follows:

$$S_{mn}^k | Y_{mn}^k \sim \mathcal{N}_c(0, Y_{mn}^k). \tag{9}$$

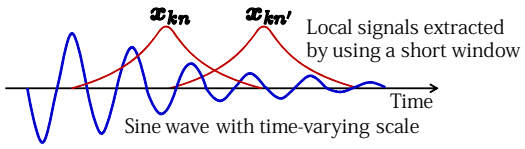


Figure 2. Local signals $\boldsymbol{x}_{kn}$ and $\boldsymbol{x}_{kn'}$ have different wave shapes and phases, but share the same periodicity.

Using $S_{mn} = \sum_k S_{mn}^k$ and $Y_{mn} = \sum_k Y_{mn}^k$, the reproducing property of the Gaussian distribution gives

$$S_{mn}|Y_{mn} \sim \mathcal{N}_c(0, Y_{mn}). \tag{10}$$

Using $X_{mn} = |S_{mn}|^2$, we get an exponential distribution:

$$X_{mn}|Y_{mn} \sim \text{Exponential}(Y_{mn}). \tag{11}$$

The probabilistic model based on superimposed Gaussian variables $\{S_{mn}^k\}_{k=1}^K$ satisfying $S_{mn} = \sum_k S_{mn}^k$ enables us to represent a posterior distribution of latent variable S_{mn}^k as a Gaussian distribution whose mean and variance are given by Eq. (5) and

$$\mathbb{V}[S_{mn}^k|S_{mn}] = Y_{mn}^k - Y_{mn}^k Y_{mn}^{-1} Y_{mn}^k. \tag{12}$$

3. TIME-DOMAIN SOURCE SEPARATION

This section recasts the problem of source separation in the time domain. We propose a probabilistic model of source-signal superimposition and show how latent source signals can be estimated in a probabilistic manner.

3.1 Problem Specification

The goal of source separation is to decompose a given mixture signal into the sum of K source signals. This decomposition is performed on a frame-by-frame basis. Suppose we have a set of N samples $\boldsymbol{O} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \in \mathbb{R}^{M \times N}$, where $\boldsymbol{x}_n \in \mathbb{R}^M$ is a local signal in the n-th frame extracted from the mixture signal by using a window of size M. Each $\boldsymbol{x}_n$ can be decomposed as follows:

$$\boldsymbol{x}_n = \sum_{k=1}^K \boldsymbol{x}_{kn}, \tag{13}$$

where $\boldsymbol{x}_{kn}$ is a local signal extracted from the k-th source signal. This time-domain formulation is equivalent to the frequency-domain formulation given by Eq. (4). Although $\{\boldsymbol{x}_{kn}\}_{n=1}^N$ are different, we assume that $\{\boldsymbol{x}_{kn}\}_{n=1}^N$ share some characteristics. For example, suppose the k-th source signal is a sine wave whose scale varies over time as shown in Figure 2. Note that $\boldsymbol{x}_{kn}$ and $\boldsymbol{x}_{kn'}$ $(n \neq n')$ have different *scales* but have the same *period*. We factorize $\boldsymbol{x}_{kn}$ into *nonstationary* and *stationary* factors as $\boldsymbol{x}_{kn} = \pi_{kn}\boldsymbol{\phi}_{kn}$, where π_{kn} is a coefficient (scale) of a local signal $\boldsymbol{\phi}_{kn}$ extracted from the k-th stationary *basis* signal. Note that the basis signal is assumed to vary over time according to stationary characteristics (*e.g.*, periodicity and whiteness).

Given $\boldsymbol{O}$ as observed data, we aim to estimate a set of latent signals $\{\boldsymbol{x}_{kn}\}_{n=1}^N$ for each k. The k-th source signal can be obtained by the overlap-add synthesis method [16]. We do not need any frequency analysis such as short-term Fourier transform (STFT) or inverse STFT.

3.2 Probabilistic Formulation

We formulate a probabilistic model of Eq. (13). A key feature is to focus on the *stationary* characteristics of the basis signal. Since the stationarity means that $\{\boldsymbol{\phi}_{kn}\}_{n=1}^N$ are expected to have the same covariance, we put a multivariate Gaussian prior shared over all frames as follows:

$$\boldsymbol{\phi}_{kn} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{V}_k), \tag{14}$$

where $\boldsymbol{V}_k \in \mathbb{R}^{M \times M}$ is a *full* covariance matrix. The mean is set to a zero vector because an audio signal is recorded as real numbers distributed on both sides of zero.

We can say that the k-th basis signal is Gaussian process (GP) distributed with a stationary (shift-invariant) kernel $\boldsymbol{V}_k$. Since in reality the signal exists over *continuous* time, it is essential to consider a probability distribution of the continuous signal. Such a distribution is a GP by definition because its marginal over any M discrete time points is a Gaussian distribution, as indicated by Eq. (14). If $\boldsymbol{V}_k$ is a periodic kernel, $\{\boldsymbol{\phi}_{kn}\}_{n=1}^N$ are expected to have a certain period but their phases and wave shapes can be different.

We will derive a likelihood of the observed signal $\boldsymbol{x}_n$. The linear relationship $\boldsymbol{x}_{kn} = \pi_{kn}\boldsymbol{\phi}_{kn}$ and Eq. (14) lead to a likelihood of $\boldsymbol{x}_{kn}$ as follows:

$$\boldsymbol{x}_{kn}|\boldsymbol{\pi}, \boldsymbol{V} \sim \mathcal{N}(\boldsymbol{0}, \pi_{kn}^2 \boldsymbol{V}_k). \tag{15}$$

Then, using the reproducing property of the Gaussian distribution and the linear relationship given by Eq. (13), we get the likelihood of $\boldsymbol{x}_n$ as follows:

$$\boldsymbol{x}_n|\boldsymbol{\pi}, \boldsymbol{V} \sim \mathcal{N}\left(\boldsymbol{0}, \sum_{k=1}^K \pi_{kn}^2 \boldsymbol{V}_k\right). \tag{16}$$

Note that Eq. (16) does not include $\boldsymbol{\phi}_{kn}$, *i.e.*, all possibilities of $\boldsymbol{\phi}_{kn}$ are taken into account. This formulation frees us from explicitly considering the phase of $\boldsymbol{\phi}_{kn}$. We here define some symbols as $H_{kn} = \pi_{kn}^2 \geq 0$, $\boldsymbol{X}_n = \boldsymbol{x}_n \boldsymbol{x}_n^T \succeq \boldsymbol{0}$, and $\boldsymbol{Y}_n = \sum_k H_{kn} \boldsymbol{V}_k \succeq \boldsymbol{0}$, where $\boldsymbol{\Psi} \succeq \boldsymbol{0}$ means that $\boldsymbol{\Psi}$ is a positive semidefinite (PSD) matrix. Then Eq. (16) gives the log-likelihood of $\boldsymbol{X}_n$ as follows:

$$\log p(\boldsymbol{X}_n|\boldsymbol{Y}_n) \stackrel{c}{=} -\frac{1}{2}\log|\boldsymbol{Y}_n| - \frac{1}{2}\text{tr}(\boldsymbol{X}_n \boldsymbol{Y}_n^{-1}), \tag{17}$$

where $\stackrel{c}{=}$ represents equality except for the constant terms.

Given a tensor $\boldsymbol{X} = [\boldsymbol{X}_1, \cdots, \boldsymbol{X}_N] \in \mathbb{R}^{M \times M \times N}$, we aim to estimate $\boldsymbol{H} \in \mathbb{R}^{K \times N}$ and $\boldsymbol{V} = [\boldsymbol{V}_1, \cdots, \boldsymbol{V}_K] \in \mathbb{R}^{M \times M \times K}$ such that the log-likelihood $\sum_n \log p(\boldsymbol{X}_n|\boldsymbol{Y}_n)$ is maximized. As shown in Section 4, this is a special case of PSDTF in which each $\boldsymbol{X}_n$ is restricted to a rank-1 PSD matrix ($\boldsymbol{X}_n = \boldsymbol{x}_n \boldsymbol{x}_n^T$). We can therefore use a multiplicative update algorithm described in Section 4.3.

3.3 Probabilistic Decomposition

After $\boldsymbol{H}$ and $\boldsymbol{V}$ are obtained, we can estimate a local signal $\boldsymbol{x}_{kn} = \pi_{kn}\boldsymbol{\phi}_{kn}$ in a probabilistic manner. Instead of estimating $\boldsymbol{\phi}_{kn}$, we can directly calculate a Gaussian posterior of $\boldsymbol{x}_{kn}$ whose mean and covariance are given by

$$\mathbb{E}[\boldsymbol{x}_{kn}|\boldsymbol{x}_n, \boldsymbol{H}, \boldsymbol{V}] = \boldsymbol{Y}_{nk} \boldsymbol{Y}_n^{-1} \boldsymbol{x}_n, \tag{18}$$

$$\mathbb{V}[\boldsymbol{x}_{kn}|\boldsymbol{x}_n, \boldsymbol{H}, \boldsymbol{V}] = \boldsymbol{Y}_{nk} - \boldsymbol{Y}_{nk} \boldsymbol{Y}_n^{-1} \boldsymbol{Y}_{nk}, \tag{19}$$

where $\boldsymbol{Y}_{nk} = H_{kn}\boldsymbol{V}_k \succeq \boldsymbol{0}$ such that $\boldsymbol{Y}_n = \sum_k \boldsymbol{Y}_{nk} \succeq \boldsymbol{0}$. Eq. (18) works as a Wiener filter that passes only a component signal of $\boldsymbol{x}_n$ matching the characteristics of kernel $\boldsymbol{V}_k$ without explicitly considering the phase and wave shape. Eqs. (18) and (19) formulated in the time domain look similar in form to Eqs. (5) and (12) formulated in the frequency domain. The key difference is that we consider the covariance structure over frequency components (*e.g.*, harmonic partials) when decomposing $\boldsymbol{x}_n$.

3.4 Frequency-Domain Representation

We discuss a frequency-domain representation (Figure 1). Let $\boldsymbol{F} \in \mathbb{C}^{M\times M}$ be the DFT matrix. Eq. (16) means that the complex spectrum $\boldsymbol{F}\boldsymbol{x}_n$ (linear transformation of $\boldsymbol{x}_n$) is complex-Gaussian distributed as follows:

$$\boldsymbol{F}\boldsymbol{x}_n|\boldsymbol{H},\boldsymbol{V} \sim \mathcal{N}_c\left(\boldsymbol{0}, \sum_{k=1}^{K} H_{kn}\boldsymbol{F}\boldsymbol{V}_k\boldsymbol{F}^H\right), \quad (20)$$

where the full covariance structure between frequency bins is considered. Note that $\boldsymbol{F}\boldsymbol{V}_k\boldsymbol{F}^H$ becomes a diagonal matrix if $\boldsymbol{V}_k$ is a circulant matrix. A trivial example is a case that $\boldsymbol{V}_k$ is an identity matrix, *i.e.*, $\boldsymbol{\phi}_{kn}$ is stationary white Gaussian noise. If $\boldsymbol{V}_k$ is a periodic kernel and its size M is much larger than its period, $\boldsymbol{V}_k$ can be *roughly* viewed as a circulant matrix. If $\boldsymbol{F}\boldsymbol{V}_k\boldsymbol{F}^H$ is diagonal, Eq. (20) reduces to a probabilistic model of IS-NMF discarding the covariance structure between frequency bins [13]. In reality, however, $\boldsymbol{V}_k$ is considerably different from a circulant matrix as shown in Section 5 (Figure 4 and Figure 5).

4. POSITIVE SEMIDEFINITE TENSOR FACTORIZATION

This section explains a new tensor factorization technique called *positive semidefinite tensor factorization* (PSDTF), in a general-purpose way. As NMF approximates N nonnegative vectors (a matrix) as the convex combinations of K nonnegative vectors, PSDTF approximates N PSD matrices (a tensor) as the convex combinations of K PSD matrices. PSDTF is therefore a natural extension of NMF.

4.1 Problem Specification

We formalize the problem of PSDTF. Suppose we have as observed data a three-mode tensor $\boldsymbol{X} = [\boldsymbol{X}_1, \cdots, \boldsymbol{X}_N] \in \mathbb{R}^{M\times M\times N}$, where each slice $\boldsymbol{X}_n \in \mathbb{R}^{M\times M}$ is a real symmetric positive semidefinite (PSD) matrix. Note that a similar discussion can be applied even if $\boldsymbol{X}_n \in \mathbb{C}^{M\times M}$ is a complex Hermitian PSD matrix such that $\boldsymbol{X}_n = \boldsymbol{X}_n^H$.

The goal of PSDTF is to approximate each PSD matrix $\boldsymbol{X}_n$ by a convex combination of PSD matrices $\{\boldsymbol{V}_k\}_{k=1}^K$ (K basis matrices) as follows:

$$\boldsymbol{X}_n \approx \sum_{k=1}^{K} H_{kn}\boldsymbol{V}_k \stackrel{\text{def}}{=} \boldsymbol{Y}_n, \quad (21)$$

where $H_{kn} \geq 0$ is a weight at the n-th slice. Eq. (21) can also be represented as $\boldsymbol{X} \approx \sum_k \boldsymbol{h}_k \otimes \boldsymbol{V}_k \stackrel{\text{def}}{=} \boldsymbol{Y}$, where $\otimes$ indicates the Kronecker product.

To evaluate the reconstruction error between PSD matrices $\boldsymbol{X}_n$ and $\boldsymbol{Y}_n$, we propose to use a Bregman matrix divergence [14] defined as follows:

$$\mathcal{C}_\phi(\boldsymbol{X}_n|\boldsymbol{Y}_n) = \phi(\boldsymbol{X}_n) - \phi(\boldsymbol{Y}_n) - \mathrm{tr}\big(\nabla\phi(\boldsymbol{Y}_n)^T(\boldsymbol{X}_n - \boldsymbol{Y}_n)\big), \quad (22)$$

where ϕ is a strictly convex matrix function. In this paper we focus on the log-determinant (LD) divergence ($\phi(\boldsymbol{Z}) = -\log|\boldsymbol{Z}|$) [17] given by

$$\mathcal{C}_{\text{LD}}(\boldsymbol{X}_n|\boldsymbol{Y}_n) = -\log\left|\boldsymbol{X}_n\boldsymbol{Y}_n^{-1}\right| + \mathrm{tr}\left(\boldsymbol{X}_n\boldsymbol{Y}_n^{-1}\right) - M. \quad (23)$$

This divergence is always nonnegative and is zero if and only if $\boldsymbol{X}_n = \boldsymbol{Y}_n$. The Itakura-Saito (IS) divergence over nonnegative numbers given by $\mathcal{C}_{\text{IS}}(x|y) = -\log(x/y) + x/y - 1$ is a well-known special case when $M = 1$, and it is often used for audio source separation.

Our goal is to estimate $\boldsymbol{H} = [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_K] \in \mathbb{R}^{N\times K}$ and $\boldsymbol{V} = [\boldsymbol{V}_1, \cdots, \boldsymbol{V}_K] \in \mathbb{R}^{M\times M\times K}$ such that the cost function $\mathcal{C}_{\text{LD}}(\boldsymbol{X}|\boldsymbol{Y}) = \sum_n \mathcal{C}_{\text{LD}}(\boldsymbol{X}_n|\boldsymbol{Y}_n)$ is minimized. Note that our model imposes the nonnegativity constraint on $\boldsymbol{H}$ and the positive semidefiniteness constraint on $\boldsymbol{V}$. This specific model is called LD-PSDTF.

4.2 Auxiliary Function Approach

We use the auxiliary function approach [15] for *indirectly* maximizing the cost function $\mathcal{C}_{\text{LD}}(\boldsymbol{X}|\boldsymbol{Y})$ with respect to $\boldsymbol{Y}$ (*i.e.*, $\boldsymbol{H}$ and $\boldsymbol{V}$) because of its analytical tractability. Let $\mathcal{F}(\boldsymbol{\theta})$ is an objective function to be minimized with respect to a parameter $\boldsymbol{\theta}$. A function $\mathcal{F}^+(\boldsymbol{\theta}, \boldsymbol{\phi})$ satisfying

$$\mathcal{F}(\boldsymbol{\theta}) \leq \mathcal{F}^+(\boldsymbol{\theta}, \boldsymbol{\phi}) \quad (24)$$

is an auxiliary function for $\mathcal{F}(\boldsymbol{\theta})$, where $\boldsymbol{\phi}$ is an auxiliary parameter. It can be proved that $\mathcal{F}(\boldsymbol{\theta})$ is non-increasing through the following iterative update rules:

$$\boldsymbol{\phi}^{\text{new}} \leftarrow \mathrm{argmin}_{\boldsymbol{\phi}}\mathcal{F}^+(\boldsymbol{\theta}^{\text{old}}, \boldsymbol{\phi}), \quad (25)$$

$$\boldsymbol{\theta}^{\text{new}} \leftarrow \mathrm{argmin}_{\boldsymbol{\theta}}\mathcal{F}^+(\boldsymbol{\theta}, \boldsymbol{\phi}^{\text{new}}). \quad (26)$$

This algorithm is theoretically guaranteed to converge and in fact a similar idea was used for IS-NMF [18].

To derive an auxiliary function $\mathcal{U}(\boldsymbol{X}|\boldsymbol{Y})$ for $\mathcal{C}_{\text{LD}}(\boldsymbol{X}|\boldsymbol{Y})$, we need to use matrix-variate inequalities based on function concavity and convexity. First, since $f(\boldsymbol{Z}) = \log|\boldsymbol{Z}|$ is concave, we calculate a tangent plane of $f(\boldsymbol{Z})$ by using a first-order Taylor expansion as follows:

$$\log|\boldsymbol{Z}| \leq \log|\boldsymbol{\Omega}| + \mathrm{tr}(\boldsymbol{\Omega}^{-1}\boldsymbol{Z}) - M, \quad (27)$$

where $\boldsymbol{\Omega}$ is an auxiliary PSD matrix (tangent point), M is the size of $\boldsymbol{Z}$, and the equality holds when $\boldsymbol{\Omega} = \boldsymbol{Z}$. For a convex function $g(\boldsymbol{Z}) = \mathrm{tr}(\boldsymbol{Z}^{-1}\boldsymbol{A})$ for any PSD matrix $\boldsymbol{A}$, we can use a sophisticated inequality [19] as follows:

$$\mathrm{tr}\left(\left(\sum_{k=1}^K \boldsymbol{Z}_k\right)^{-1}\boldsymbol{A}\right) \leq \sum_{k=1}^{K}\mathrm{tr}\left(\boldsymbol{Z}_k^{-1}\boldsymbol{\Phi}_k\boldsymbol{A}\boldsymbol{\Phi}_k^T\right), \quad (28)$$

where $\{\boldsymbol{Z}_k\}_{k=1}^K$ is a set of arbitrary PSD matrices, $\{\boldsymbol{\Phi}_k\}_{k=1}^K$ is a set of auxiliary matrices that sum to the identity matrix (*i.e.*, $\sum_k \boldsymbol{\Phi}_k = \boldsymbol{I}$), and the equality holds when $\boldsymbol{\Phi}_k = \boldsymbol{Z}_k(\sum_{k'}\boldsymbol{Z}_{k'})^{-1}$.

Using Inequalities (27) and (28), we can derive an auxiliary function $\mathcal{U}(\boldsymbol{X}_n|\boldsymbol{Y}_n)$ for Eq. (23) as follows:

$$\begin{aligned}
&\mathcal{C}_{\mathrm{LD}}(\boldsymbol{X}_n|\boldsymbol{Y}_n) \stackrel{c}{=} \log|\boldsymbol{Y}_n| + \mathrm{tr}\left(\boldsymbol{X}_n\boldsymbol{Y}_n^{-1}\right) \\
&\leq \log|\boldsymbol{\Omega}_n| + \mathrm{tr}\left(\boldsymbol{Y}_n\boldsymbol{\Omega}_n^{-1}\right) - M \\
&\quad + \textstyle\sum_k \mathrm{tr}\left(\boldsymbol{Y}_{nk}^{-1}\boldsymbol{\Phi}_{nk}\boldsymbol{X}_n\boldsymbol{\Phi}_{nk}^T\right) \qquad (29) \\
&\leq \log|\boldsymbol{\Omega}_n| + \textstyle\sum_k \mathrm{tr}\left(H_{kn}\boldsymbol{V}_k\boldsymbol{\Omega}_n^{-1}\right) - M \\
&\quad + \textstyle\sum_k \mathrm{tr}\left(H_{kn}^{-1}\boldsymbol{V}_k^{-1}\boldsymbol{\Phi}_{nk}\boldsymbol{X}_n\boldsymbol{\Phi}_{nk}^T\right) \stackrel{\mathrm{def}}{=} \mathcal{U}(\boldsymbol{X}_n|\boldsymbol{Y}_n),
\end{aligned}$$

where $\boldsymbol{\Omega}_n$ is a PSD matrix and $\{\boldsymbol{\Phi}_{nk}\}_{k=1}^K$ is a set of auxiliary matrices that sum to the identity matrix. The equality holds, *i.e.*, $\mathcal{U}(\boldsymbol{X}_n|\boldsymbol{Y}_n)$ is minimized, when

$$\boldsymbol{\Omega}_n = \boldsymbol{Y}_n, \quad \boldsymbol{\Phi}_{nk} = \boldsymbol{Y}_{nk}\boldsymbol{Y}_n^{-1}. \qquad (30)$$

4.3 Multiplicative Update

We can derive multiplicative update (MU) rules that monotonically decrease the total auxiliary function $\mathcal{U}(\boldsymbol{X}|\boldsymbol{Y}) = \sum_n \mathcal{U}(\boldsymbol{X}_n|\boldsymbol{Y}_n)$. We here assume $\mathrm{tr}(\boldsymbol{V}_k) = 1$ (unit trace) to remove the scale arbitrariness. If $\mathrm{tr}(\boldsymbol{V}_k) = s$, the scale adjustments $\boldsymbol{V}_k \leftarrow \frac{1}{s}\boldsymbol{V}_k$ and $H_{kn} \leftarrow sH_{kn}$ do not change $\mathcal{C}_{\mathrm{LD}}(\boldsymbol{X}_n|\boldsymbol{Y}_n)$ and $\mathcal{U}(\boldsymbol{X}_n|\boldsymbol{Y}_n)$. Letting the partial derivative of Eq. (29) with respect to H_{kn} be equal to be zero and using Eq. (30), we get the following update rule:

$$H_{kn} \leftarrow H_{kn}\sqrt{\frac{\mathrm{tr}\left(\boldsymbol{Y}_n^{-1}\boldsymbol{V}_k\boldsymbol{Y}_n^{-1}\boldsymbol{X}_n\right)}{\mathrm{tr}\left(\boldsymbol{Y}_n^{-1}\boldsymbol{V}_k\right)}}. \qquad (31)$$

Then, letting the partial derivative with respect to $\boldsymbol{V}_k$ be equal to be zero and using Eq. (30), we get

$$\boldsymbol{V}_k\boldsymbol{P}_k\boldsymbol{V}_k = \boldsymbol{V}_k^{\mathrm{old}}\boldsymbol{Q}_k\boldsymbol{V}_k^{\mathrm{old}}, \qquad (32)$$

where $\boldsymbol{P}_k$ and $\boldsymbol{Q}_k$ are PSD matrices given by

$$\boldsymbol{P}_k = \sum_{n=1}^N H_{kn}\boldsymbol{Y}_n^{-1}, \ \boldsymbol{Q}_k = \sum_{n=1}^N H_{kn}\boldsymbol{Y}_n^{-1}\boldsymbol{X}_n\boldsymbol{Y}_n^{-1}. \qquad (33)$$

Eq. (32) can be solved analytically by using the Cholesky decomposition $\boldsymbol{Q}_k = \boldsymbol{L}_k\boldsymbol{L}_k^T$, where $\boldsymbol{L}_k$ is a lower triangular matrix. Finally, we get the following update rule:

$$\boldsymbol{V}_k \leftarrow \boldsymbol{V}_k\boldsymbol{L}_k(\boldsymbol{L}_k^T\boldsymbol{V}_k\boldsymbol{P}_k\boldsymbol{V}_k\boldsymbol{L}_k)^{-\frac{1}{2}}\boldsymbol{L}_k^T\boldsymbol{V}_k, \qquad (34)$$

where the positive semidefiniteness of $\boldsymbol{V}_k$ is ensured. Note that a real matrix $\boldsymbol{A}$ is said to be positive semidefinite if and only if $\boldsymbol{A} = \boldsymbol{Z}\boldsymbol{Z}^T$ is satisfied for some real matrix $\boldsymbol{Z}$.

4.4 Connection to IS-NMF and Source Separation

LD-PSDTF reduces to IS-NMF if PSD matrices $\boldsymbol{X}_n$ and $\boldsymbol{V}_k$ are restricted to diagonal matrices. Since the diagonal elements of an arbitrary PSD matrix are always nonnegative, the cost function given by Eq. (23) is decomposed as $\mathcal{C}_{\mathrm{LD}}(\boldsymbol{X}_n|\boldsymbol{Y}_n) = \sum_m \mathcal{C}_{\mathrm{IS}}(X_{nmm}|Y_{nmm})$ and the MU rules given by Eq. (31) and Eq. (34) thus reduce to the MU rules of IS-NMF [15]. As described in [15], several algorithms such as an expectation-maximization (EM) algorithm and a convergence-guaranteed MU algorithm can be used for IS-NMF. The same is true for LD-PSDTF, and for faster convergence we derived the MU algorithm.

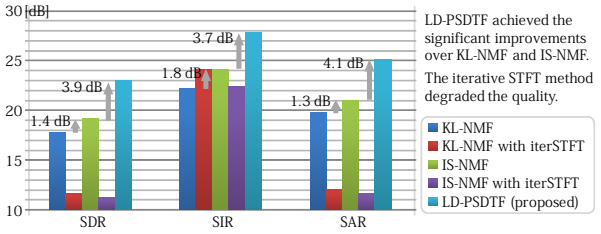


Figure 3. Source separation performance.

To use LD-PSDTF for source separation, we set $\boldsymbol{X}_n = \boldsymbol{x}_n\boldsymbol{x}_n^T$ (rank-1 matrix) as shown in Section 3.2. Since the negative of the log-likelihood given by Eq. (17) is equal to the cost function given by Eq. (23) except for constant terms, the maximum-likelihood estimates of $\boldsymbol{H}$ and $\boldsymbol{V}$ can be obtained by the MU algorithm for LD-PSDTF. Note that general LD-PSDTF is formulated for *any-rank* matrix $\boldsymbol{X}_n$.

5. EVALUATION

This section reports a comparative experiment evaluating the source separation performance of LD-PSDTF.

5.1 Experimental Conditions

We used three mixture audio signals each of which was synthesized using piano sounds (011PFNOM), electric guitar sounds (131EGLPM), or clarinet sounds (311CLNOM) recorded in the RWC Music Database: Musical Instrument Sound [20]. Each mixture signal was made by concatenating seven 2.0-s isolated or mixture sounds (C4, E4, G4, C4+E4, C4+G4, E4+G4, and C4+E4+G4). The resulting 14.0-s signals were sampled at 16kHz.

The task was to separate each mixture signal into three source signals corresponding to C4, E4, and G4. The local signals $\{\boldsymbol{x}_n\}_{n=1}^N$ were extracted by using a Gaussian window with a width of 512 samples ($M = 512$) and a shifting interval of 160 samples ($N = 1400$). The PSD matrices $\boldsymbol{V}$ and their activations $\boldsymbol{H}$ were estimated by using the MU algorithm with $K = 3$. For comparison, we used KL-NMF for amplitude-spectrogram decomposition and IS-NMF for power-spectrogram decomposition (Section 2.3 and Section 2.4). The number of iterations was 100 in each method. We also tested the iterative STFT method [6] as a phase reconstructor for NMF. We evaluated the quality of separated signals in terms of source-to-distortion ratio (SDR), source-to-interferences ratio (SIR), and sources-to-artifacts ratio (SAR) using the BSS Eval toolbox [21].

5.2 Experimental Results

The experimental results showed the clear superiority of LD-PSDTF for source separation (Figure 3). The average SDR, SIR, and SAR were 17.7 dB, 22.2 dB, and 19.7 dB in KL-NMF, 19.1 dB, 24.0 dB, and 21.0 dB in IS-NMF, and 23.0 dB, 27.7 dB, and 25.1 dB in LD-PSDTF.[1] We found that the iterative STFT method degraded the quality of separated signals. This implies that the spectrogram consistency does not always lead to the perceived quality of au-

[1] Audio files and a sample code are at the first author's website.

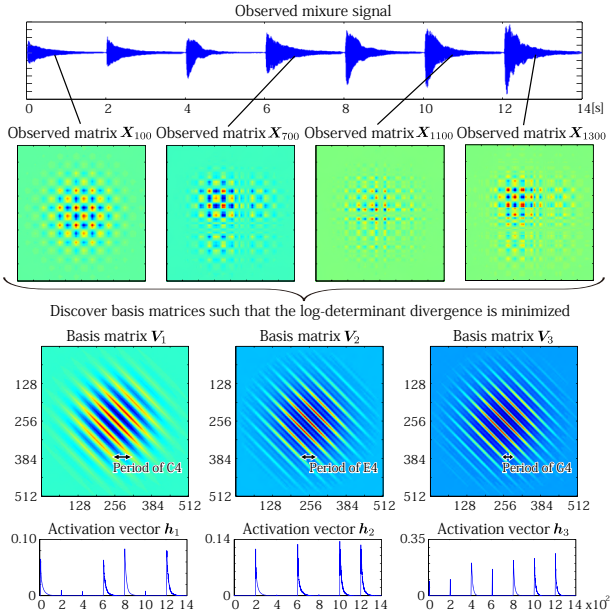


Figure 4. Factorization of a piano mixture signal.

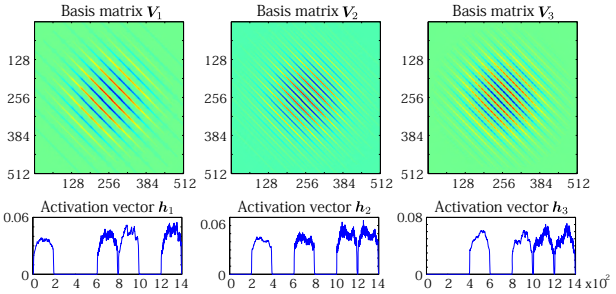


Figure 5. Factorization of a clarinet mixture signal.

dio signals, as suggested in [10]. We confirmed that LD-PSDTF can appropriately estimate $\boldsymbol{V}$ and $\boldsymbol{H}$ from both decaying and sustained sounds (Figure 4 and Figure 5). The reason that each $\boldsymbol{X}_n$ does not appear to be well approximated by $\boldsymbol{Y}_n$ (a convex combination of $\{\boldsymbol{V}_k\}_{k=1}^K$) is that the cost function based on the LD divergence allows $\boldsymbol{Y}_n$ to overestimate $\boldsymbol{X}_n$ with a smaller penalty. A main limitation of LD-PSDTF is that its computational cost is $O(KNM^3)$ while the computational cost of NMF is $O(KNM)$. In this experiment, LD-PSDTF spent several hours for analyzing each mixture signal on Xeon X5492 (3.4 GHz). Therefore, we think that initializing LD-PSDTF by using basis vectors and their activations obtained by IS-NMF can reduce the computational cost and help avoid local minima.

6. CONCLUSION

This paper presented log-determinant positive semidefinite tensor factorization (LD-PSDTF) as a natural extension of Itakura-Saito NMF (IS-NMF). We derived a convergence-guaranteed multiplicative update algorithm and showed the clear superiority of LD-PSDTF over NMF variants in terms of source separation quality.

There are several interesting directions. To separate music signals into instrument parts, we plan to fuse the source-filter model into the framework of LD-PSDTF as in the composite autoregressive system [22]. We also plan to investigate another variant of PSDTF based on the von Neumann divergence ($\phi(\boldsymbol{Z}) = \mathrm{tr}(\boldsymbol{Z}\log\boldsymbol{Z} - \boldsymbol{Z})$ in Eq. (22)) that can be viewed as an extension of KL-NMF.

Acknowledgment: This study was supported in part by JSPS KAKENHI 23700184, MEXT KAKENHI 25870192, and JST CREST OngaCREST.

7. REFERENCES

[1] K. Itoyama *et al.* Query-by-example music information retrieval by score-informed source separation and remixing technologies. *EURASIP Journal*, 2010. Article ID 172961.

[2] M. Goto. Active music listening interfaces based on signal processing. *ICASSP*, volume 4, pp. 1441–1444, 2007.

[3] K. Yoshii *et al.* Drumix: An audio player with real-time drum-part rearrangement functions for active music listening. *IPSJ Digital Courier*, 3:134–144, 2007.

[4] N. Sturmel *et al.* Linear mixing models for active listening of music productions in realistic studio conditions. *AES Convention*, 2012.

[5] D. Lee and H. Seung. Algorithms for non-negative matrix factorization. *NIPS*, pp. 556–562, 2000.

[6] D. W. Griffin and J. S. Lim. Signal estimation from modified short-time Fourier transform. *IEEE Trans. on ASLP*, 32(2):236–243, 1984.

[7] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama. Explicit consistency constraints for STFT spectrograms and their application to phase reconstruction. *SAPA*, pp. 23–28, 2008.

[8] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama. Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency. *DAFx*, pp. 397–403, 2010.

[9] H. Kameoka *et al.* Complex NMF: A new sparse representation for acoustic signals. *ICASSP*, pp. 45–48, 2009.

[10] J. Le Roux *et al.* Consistent Wiener filtering: Generalized time-frequency masking respecting spectrogram consistency. *LVA/ICA*, pp. 89–96, 2010.

[11] K. Yoshii, R. Romioka, D. Mochihashi, and M. Goto. Infinite positive semidefinite tensor factorization for source separation of mixture signals. *ICML*, pp. 576–584, 2013.

[12] P. Smaragdis and J. Brown. Nonnegative matrix factorization for polyphonic music transcription. *WASPAA*, 2003.

[13] C. Févotte, N. Bertin, and J.-L. Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis. *Neu. Comp.*, 21(3):793–830, 2009.

[14] L. M. Bregman. The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex programming. *USSR CMMP*, 1967.

[15] M. Nakano *et al.* Convergence-guaranteed multiplicative algorithms for non-negative matrix factorization with beta divergence. *MLSP*, pp. 283–288, 2010.

[16] J. Allen and L. Rabiner. A unified approach to short-time Fourier analysis and synthesis. *IEEE*, 1977.

[17] B. Kulis, M. Sustik, and I. Dhillon. Low-rank kernel learning with Bregman matrix divergences. *JMLR*, 10:341–376, 2009.

[18] M. Hoffman *et al.* Bayesian nonparametric matrix factorization for recorded music. *ICML*, pp. 439–446, 2010.

[19] H. Sawada, H. Kameoka, S. Araki, and N. Ueda. Efficient algorithms for multichannel extensions of Itakura-Saito nonnegative matrix factorization. *ICASSP*, pp. 261–264, 2012.

[20] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Music genre database and musical instrument sound database. *ISMIR*, pp. 229–230, 2003.

[21] E. Vincent, R. Gribonval, and C. Févotte. Performance measurement in blind audio source separation. *IEEE Trans. on ASSP*, 14(4):1462–1469, 2006.

[22] H. Kameoka and K. Kashino. Composite autoregressive system for sparse source-filter representation of speech. *ISCAS*, pp. 2477–2480, 2009.

BETA PROCESS SPARSE NONNEGATIVE MATRIX FACTORIZATION FOR MUSIC

Dawen Liang
LabROSA, EE Dept.
Columbia University
dl2771@columbia.edu

Matthew D. Hoffman
Adobe Research
Adobe Systems Incorporated
mathoffm@adobe.com

Daniel P. W. Ellis
LabROSA, EE Dept.
Columbia University
dpwe@ee.columbia.edu

ABSTRACT

Nonnegative matrix factorization (NMF) has been widely used for discovering physically meaningful latent components in audio signals to facilitate source separation. Most of the existing NMF algorithms require that the number of latent components is provided *a priori*, which is not always possible. In this paper, we leverage developments from the Bayesian nonparametrics and compressive sensing literature to propose a probabilistic *Beta Process Sparse NMF* (BP-NMF) model, which can automatically infer the proper number of latent components based on the data. Unlike previous models, BP-NMF explicitly assumes that these latent components are often completely silent. We derive a novel mean-field variational inference algorithm for this nonconjugate model and evaluate it on both synthetic data and real recordings on various tasks.

1. INTRODUCTION

Nonnegative matrix factorization (NMF) [9] has been extensively applied to analyze audio signals, since the approximate decomposition of the audio spectrogram into the product of 2 nonnegative matrices $\mathbf{X} \approx \mathbf{WH}$ provides a physically meaningful interpretation. We can view each column of $\mathbf{X}$, which represents the power density across frequencies at a particular time, as a nonnegative linear combination of the columns of $\mathbf{W}$, determined by the column of activation $\mathbf{H}$. Thus $\mathbf{W}$ can be considered as a dictionary, where each column acts as a component. This can be particularly useful for audio source separation, where the goal is to find out the individual sources from mixed signal.

Audio source separation poses a meaningful and challenging problem, which has been actively studied for the last few decades. One of the obstacles which makes source separation difficult is that the number of sources is generally not known. For example, when we listen to a piece of polyphonic music, it is difficult and tedious to figure out how many notes or instruments are being played. However, most existing NMF algorithms require the number of components to be provided as input, based on the assumption that there exists a certain mapping between the learned components and real sources. To address this issue, we propose BP-NMF, a nonparametric Bayesian NMF model that uses a beta process prior. The model automatically determines how many sources it needs to explain the data during posterior inference.

1.1 Related Work

NMF has been applied to many music analysis problems such as music transcription [1, 12], music analysis [5], and music source separation [10, 15].

On the other hand, most of the literature on nonparametric Bayesian latent factor models focuses on conjugate linear Gaussian models, for example, beta process factor analysis [11] which is the main inspiration for BP-NMF. However, such models are not appropriate for audio spectrograms as they do not impose nonnegativity constraints. To address this limitation, [7] proposed a nonparametric Bayesian NMF model based on the gamma process.

BP-NMF extends the standard NMF model in two ways:

- BP-NMF can explicitly and completely silence latent components when they should not be active. This captures the intuition that a note which appears frequently during one phrase may not contribute anything in another phrase, and most notes are silent most of the time.
- The number of latent components, which is difficult to set *a priori*, is inferred by the model.

Both of these issues have been addressed in previous work, but to the authors' knowledge, BP-NMF is the first model to combine them.

2. BP-NMF

We adopt the notational conventions that upper case bold letters (e.g. $\mathbf{X}, \mathbf{D}$, $\mathbf{S}$ and $\mathbf{Z}$) denote matrices and lower case bold letters (e.g. $\boldsymbol{x}$, $\boldsymbol{d}$ $\boldsymbol{s}$, and $\boldsymbol{z}$) denote vectors. $f \in \{1, 2, \cdots, F\}$ is used to index frequency. $t \in \{1, 2, \cdots, T\}$ is used to index time. $k \in \{1, 2, \cdots, K\}$ is used to index dictionary components.

BP-NMF is formulated as:

$$\mathbf{X} = \mathbf{D}(\mathbf{S} \odot \mathbf{Z}) + \mathbf{E} \tag{1}$$

where $\mathbf{X}$ is a $F \times T$ spectrogram and $\mathbf{D}$ is a $F \times K$ dictionary with K components. The activation is the Hadamard product between a nonnegative matrix $\mathbf{S}$ and a binary mask $\mathbf{Z}$, both of which have the shape of $K \times T$. $\mathbf{E}$ is an i.i.d. Gaussian noise matrix which has the same shape as $\mathbf{X}$. We use Gaussian noise model instead of Poisson or exponential mainly for mathematical convenience and extending BP-NMF to more audio-oriented model is part of the future work. Unlike previous models, BP-NMF can explicitly silence some components by turning off the corresponding elements in $\mathbf{Z}$. For example, when a clarinet is playing A3 the model should silence all clarinet notes that are not A3.

We place a beta process [6] prior on the binary mask $\mathbf{Z}$ so that the number of components K can potentially go to infinity and the inference algorithm will choose the proper number to describe the data. We adopt the finite approximation to a beta process in [11]:

$$\begin{aligned} Z_{kt} &\sim \text{Bernoulli}(\pi_k) \\ \pi_k &\sim \text{Beta}(a_0/K, b_0(K-1)/K) \end{aligned} \tag{2}$$

where K is set to a large number. As shown in [4], a finite approximation for Indian buffet process [1] performs comparably well as the infinite model. In this formulation π_k explicitly controls the prevalence of each individual component; the closer π_k is to zero, the less frequently it will contribute to $\mathbf{X}$. The rest of the model is specified as:

$$\begin{aligned} \log(\boldsymbol{d}_k) &\sim \mathcal{N}(0, \mathbf{I}_F) & \boldsymbol{\epsilon}_t &\sim \mathcal{N}(0, \gamma_\epsilon^{-1}\mathbf{I}_F) \\ \boldsymbol{s}_t &\sim \text{Gamma}(\alpha, \beta) & \gamma_\epsilon &\sim \text{Gamma}(c_0, d_0) \end{aligned} \tag{3}$$

The choice of component $\boldsymbol{d}_k$ being lognormal distributed will become natural as we describe our inference algorithm in Section 3.2. For activation $\boldsymbol{s}_t$, being gamma distributed instead of lognormal distributed is easier to extend to a time-dependent gamma chain prior [5]. The full model is summarized in Figure 1.

Exact inference for this model is infeasible, so we instead derive a mean-field [8] variational inference algorithm. Note that this model is nonconjugate between the observation $\mathbf{X}$ and priors $\mathbf{D}$ and $\mathbf{S}$, which makes deriving an inference algorithm more difficult.

3. VARIATIONAL INFERENCE

3.1 Laplace Approximation Variational Inference

Since BP-NMF is nonconjugate, we use Laplace approximation variational inference [16].

Given a probabilistic model $P(X, \Theta)$ where X denotes the observation and Θ denotes hidden variables, mean-field inference approximates the posterior $P(\Theta|X)$ with a fully factorized variational distribution $q(\Theta) = \prod_i q(\theta_i)$ by minimizing the KL-divergence between the variational distribution and the true posterior. Inference can be carried out via coordinate descent for each hidden variable θ_i which is guaranteed to find a local optimum.

[1] It has been shown [13] that beta process is the de Finetti mixing measure for the Indian buffet process.

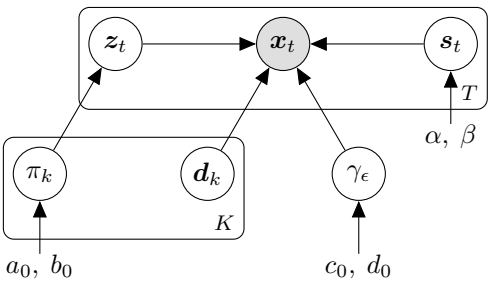


Figure 1: Graphical model representation of BP-NMF. Shaded node represents observed variable (spectrogram). Unshaded nodes represent hidden variables. A directed edge from node a to node b denotes that the variable b depends on the value of variable a. Plates denote replication by the value in the lower right of the plate.

The general mean-field update for a variable θ_i can be shown [3] to be:

$$q(\theta_i) \propto \exp\{\langle \log P(X, \Theta) \rangle_{-\theta_i}\} \tag{4}$$

where $\langle \cdot \rangle_{-\theta_i}$ indicates the expectation with respect to $q(\Theta \setminus \{\theta_i\})$. For simplicity, we will omit the subscript $-\theta_i$ when there is no ambiguity.

For nonconjugate model, we cannot write Eq. 4 in closed form. The Laplace method is used to approximate $q(\theta_i)$:

$$\begin{aligned} f(\theta_i) &= \langle \log P(X, \Theta) \rangle_{-\theta_i} \\ &\approx f(\hat{\theta}_i) + \frac{1}{2}(\theta_i - \hat{\theta}_i)^T H(\hat{\theta}_i)(\theta_i - \hat{\theta}_i) \end{aligned} \tag{5}$$

A second-order Taylor expansion is taken in (5) where $\hat{\theta}_i$ is a local maximum of $f(\theta_i)$ and $H(\hat{\theta}_i)$ is the Hessian matrix. This suggests that we use a Gaussian variational distribution for $q(\theta_i)$:

$$q(\theta_i) = \mathcal{N}(\hat{\theta}_i, -H(\hat{\theta}_i)^{-1}) \tag{6}$$

$H(\hat{\theta}_i)$ is guaranteed to be negative definite at any local maximum of $f(\theta)$, provided $f(\theta)$ is smooth. $-H(\hat{\theta}_i)^{-1}$ is therefore a valid covariance matrix. Conjugate gradient or L-BFGS can be used to search for $\hat{\theta}_i$.

3.2 Inference for BP-NMF

Laplace approximation variational inference assumes that the nonconjugate continuous variables are unconstrained, thus we reparametrize $\{\mathbf{D}, \mathbf{S}\}$ as $\{\mathbf{\Phi}, \mathbf{\Psi}\}$, where $\mathbf{\Phi} \in \mathbb{R}^{F \times K}$ with $\Phi_{fk} = \log(D_{fk})$ and $\mathbf{\Psi} \in \mathbb{R}^{K \times T}$ with $\Psi_{kt} = \log(S_{kt})$. The fully-factorized variational distribution is:

$$q(\Theta) = q(\gamma_\epsilon) \prod_{k=1}^{K} q(\pi_k) \Big(\prod_{f=1}^{F} q(\Phi_{fk}) \Big) \prod_{t=1}^{T} q(\Psi_{kt}) q(Z_{kt})$$

where the variational distributions are specified as:

$$\begin{aligned} q(\Phi_{fk}) &= \mathcal{N}(\mu_{fk}^{(\Phi)}, 1/\gamma_{fk}^{(\Phi)}) \\ q(\Psi_{kt}) &= \mathcal{N}(\mu_{kt}^{(\Psi)}, 1/\gamma_{kt}^{(\Psi)}) \\ q(Z_{kt}) &= \text{Bernoulli}(p_{kt}^{(z)}) \\ q(\pi_k) &= \text{Beta}(\alpha_k^{(\pi)}, \beta_k^{(\pi)}) \\ q(\gamma_\epsilon) &= \text{Gamma}(\alpha^{(\epsilon)}, \beta^{(\epsilon)}) \end{aligned} \tag{7}$$

We will briefly describe the mean-field update and a Python implementation is available online [2].

3.2.1 Update $\mathbf{\Phi}$ and $\mathbf{\Psi}$

Following Eq. 4, we can write $q(\Phi_{fk})$ as:

$$\begin{aligned} q(\Phi_{fk}) &\propto \exp\{\langle \log P(\mathbf{X}, \Theta)\rangle_{-\Phi_{fk}}\} \\ &\propto \exp\{\langle \log P(\boldsymbol{x}_f|\boldsymbol{\phi}_f, \mathbf{\Psi}, \mathbf{Z}, \gamma_\epsilon)\rangle + \log P(\Phi_{fk})\} \\ &= \exp\{f(\Phi_{fk})\} \end{aligned} \tag{8}$$

and express $P(\boldsymbol{x}_f|\boldsymbol{\phi}_f, \mathbf{\Psi}, \mathbf{Z}, \gamma_\epsilon)$ in exponential family form:

$$\begin{aligned} &\langle \log P(\boldsymbol{x}_f|\boldsymbol{\phi}_f, \mathbf{\Psi}, \mathbf{Z}, \gamma_\epsilon)\rangle \\ =&\langle \eta(\boldsymbol{\phi}_f, \mathbf{\Psi}, \mathbf{Z}, \gamma_\epsilon)^T\rangle T(\boldsymbol{x}_f) - \langle A(\eta)\rangle. \end{aligned} \tag{9}$$

For BP-NMF, both $\langle \eta(\boldsymbol{\phi}_f, \boldsymbol{\psi}_t, \boldsymbol{z}_t, \gamma_\epsilon)\rangle$ and $\langle A(\eta)\rangle$ can be computed in closed form. Thus, we can search for a local maximum $\hat{\Phi}_{fk}$. The mean-field update following Eq. 6 is:

$$\begin{aligned} \mu^{(\Phi)}_{fk} &\leftarrow \hat{\Phi}_{fk}, \\ \gamma^{(\Phi)}_{fk} &\leftarrow -\frac{\partial^2 f}{\partial \Phi^2_{fk}}(\hat{\Phi}_{fk}). \end{aligned} \tag{10}$$

The update for Ψ_{kt} is basically the same as Φ_{fk}, except that $P(\Psi_{kt})$ is a log-gamma distribution:

$$P(\Psi_{kt}) \propto \exp\{\alpha\Psi_{kt} - \beta\exp\{\Psi_{kt}\}\}. \tag{11}$$

3.2.2 Update $\mathbf{Z}$

Similarly, we can follow Eq. 4:

$$\begin{aligned} q(Z_{kt}) &\propto \exp\{\langle \log P(\mathbf{X}, \Theta)\rangle_{-Z_{kt}}\} \\ &\propto \exp\{\langle \log P(\boldsymbol{x}_t|\mathbf{\Phi}, \boldsymbol{\psi}_t, \mathbf{z}_t, \gamma_\epsilon)\rangle + \langle \log P(Z_{kt}|\pi_k)\rangle\} \end{aligned} \tag{12}$$

Since Z_{kt} is Bernoulli distributed, we can explicitly compute $P_0 \propto q(Z_{kt} = 0)$ and $P_1 \propto q(Z_{kt} = 1)$, respectively. Then the update for Z_{kt} can be carried out:

$$p^{(z)}_{kt} \leftarrow \frac{P_1}{P_0 + P_1} \tag{13}$$

3.2.3 Update $\boldsymbol{\pi}$ and γ_ϵ

In BP-NMF, $\boldsymbol{\pi}$ and $\mathbf{Z}$ are conjugate, therefore we can directly derive the mean-field update for $\boldsymbol{\pi}$ in closed form:

$$\begin{aligned} \alpha^{(\pi)}_k &\leftarrow \frac{a_0}{K} + \sum_{t=1}^{T}\langle Z_{kt}\rangle \\ \beta^{(\pi)}_k &\leftarrow \frac{b_0(K-1)}{K} + T - \sum_{t=1}^{T}\langle Z_{kt}\rangle \end{aligned} \tag{14}$$

Similarly, γ_ϵ can also be updated in closed form:

$$\begin{aligned} \alpha^{(\epsilon)} &\leftarrow c_0 + \frac{1}{2}FT \\ \beta^{(\epsilon)} &\leftarrow d_0 + \frac{1}{2}\sum_{t=1}^{T} \parallel \boldsymbol{x}_t - \langle \mathbf{D}(\boldsymbol{s}_t \odot \boldsymbol{z}_t)\rangle \parallel_2^2 \end{aligned} \tag{15}$$

[2] https://github.com/dawenl/bp_nmf

3.3 Accelerating inference

Both [11] and [7] proposed heuristics to speed up the inference. In general, we want to set number of dictionary components K to be large so that it can better approximate the infinite functional prior. On the other hand, a large value of K will dramatically increase the time for inference. This can be compensated by setting K initially to a large value and truncating the rarely-used dictionary components as the inference proceeds. The heuristic applied for BP-NMF is that, for dictionary component $\boldsymbol{d}_k$, if the corresponding π_k drops below 10^{-3} of the maximal $\boldsymbol{\pi}$, we skip it during the inference. The first few iterations may be slow, but inference accelerates as more elements of $\boldsymbol{\pi}$ are driven towards 0.

4. EXPERIMENTS

We conducted a set of experiments to evaluate if BP-NMF can effectively capture the latent components from music recordings. First, we performed a sanity check on a synthetic example. Then we tested BP-NMF on 2 different tasks: bandwidth expansion and blind source separation. We also designed a transcription-based mechanism to evaluate the quality of the learned dictionary.

All experiments were done on magnitude spectrum with hyperparameters $\alpha = \beta = 2$, $a_0 = b_0 = 1$, and $c_0 = d_0 = 10^{-6}$. All the variational parameters were randomly initialized. The initial K was set to 512. All recordings were sampled at 22.05 kHz.

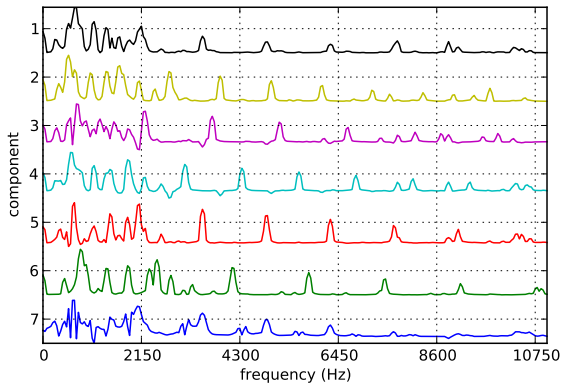
We compared with three other NMF algorithms: GaP-NMF [7] which is another nonparametric Bayesian NMF based on the gamma process, IS-NMF [5] which uses the audio-oriented Itakura-Saito divergence as loss function, and EUC-NMF [9] which minimizes the sum of the squared Euclidean distance and can be considered as a finite version of BP-NMF.

4.1 Synthetic Data

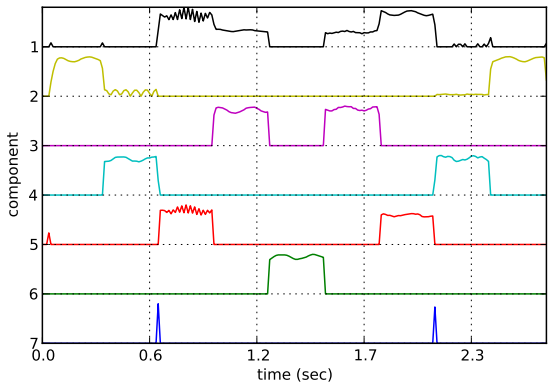
We synthesized a short clip of audio with 5 distinct piano notes and 5 distinct clarinet notes using *ChucK* [3] which is based on physical models for the instruments. At any given time, one piano note and one clarinet note are played simultaneously at different pitches.

DFTs of 512 samples (23.2 ms) were computed with 50% overlap. K quickly converged to 7 after a few iterations of variational inference. Ideally, there should be 10 components, but since some of the notes only appear with some others, they were grouped together by BP-NMF. The learned dictionary components (in log scale) and activations are shown in the Figure 2, from top to bottom in descending order of π_k. As we can see, there are clear harmonic structures in the learned dictionary and the activation does reasonably reflect the location where note combinations are played. Most importantly, the binary mask $\boldsymbol{Z}$ succeeds in explicitly controlling the appearance and disappearance of the components, which is not reflected when we test on GaP-NMF, IS-NMF, and EUC-NMF.

[3] http://chuck.stanford.edu/



(a) Dictionary components $\mathbf{D}$ in log scale.



(b) Activations $\mathbf{S} \odot \mathbf{Z}$.

Figure 2: The learned dictionary components and activations from BP-NMF on synthetic data. Both of them are listed in descending order of π_k.

4.2 Bandwidth Expansion

The basic idea of bandwidth expansion [2] is to infer the high-frequency content of a signal given only the low frequency part of the spectrum.

We use 2 pieces of music: *Pink Moon* by Nick Drake and *Funky Kingston* by Toots and the Maytals, both of which are also used in [7] for bandwidth expansion evaluation. DFTs of 512 samples are computed with no overlap. We take the middle 4000 frames of each piece and do a 5-fold cross-validation: 4/5 of the data is used to learn the dictionary. For the remaining 1/5, the top 2 octaves (192 frequency bins) are removed as a held-out set. We encode the low-frequency content with the corresponding part of the learned dictionary and predict the high-frequency content by reconstructing the full frequency band with the whole dictionary on the encoded activation.

Here we use predictive likelihood as a metric. We compare BP-NMF with GaP-NMF and EUC-NMF. The reason for not including IS-NMF is that it has been compared on the exactly same task with GaP-NMF in [7] and GaP-NMF has comparably better performance.

Unlike BP-NMF and GaP-NMF, EUC-NMF needs to specify the number of components K. Given the relationship between BP-NMF and EUC-NMF, we set K to the average number of dictionary components inferred by BP-NMF. Since both BP-NMF and EUC-NMF assume the noise is Gaussian distributed while GaP-NMF assumes the noise is exponential distributed, the predictive likelihood should be computed differently. However, this would give the exponential model an advantage, as the Gaussian distribution assigns very low probability to outcomes far from its mean, while the exponential distribution can give moderately high likelihood to values close to 0 even if they are far from the mean. To adjust for this, we evaluate the predictive likelihood under an exponential distribution for all three models. This may arguably still favor GaP-NMF as it is trained using the exponential model that it is tested on.

Geometric mean of predictive likelihood with standard error under exponential model is reported in Figure 3. Contrary to the results in [7], GaP-NMF does not dominate. This is partially due to the adjustment for Gaussian models. But the lower sampling rate of 22.05 kHz also helps the Gaussian model, since a fixed-variance Gaussian distribution has trouble modeling low-energy signals such as those that tend to appear at very high frequencies.

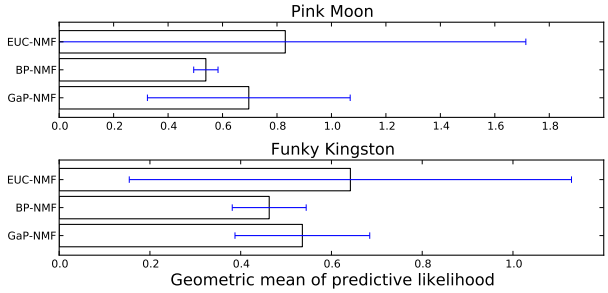


Figure 3: The geometric mean of the predictive likelihood under an exponential model for *Pink Moon* and *Funky Kingston* on a 5-fold cross-validation with standard error. In general, there is no significant difference among EUC-NMF, BP-NMF, and GaP-NMF. However, BP-NMF gives more stable results with smaller standard error.

The results in Figure 3 show that there is no significant difference among EUC-NMF, BP-NMF, and GaP-NMF. However, BP-NMF gives more stable results with smaller standard error, while its parametric counterpart EUC-NMF has much larger standard error compared with both BP-NMF and GaP-NMF.

4.3 Blind Source Separation

As with GaP-NMF, BP-NMF can also be applied to blind source separation. The model formulation of BP-NMF can be directly adopted for blind source separation, where each dictionary component can be considered as all or part of the source.

We evaluate BP-NMF for blind source separation and compare with GaP-NMF using MIREX F_0 estimation data,

Table 1: Instrument-level `bss_eval` results. The last column lists the number of components K inferred by the models.

	SDR	SIR	SAR	K
BP-NMF	0.65	7.46	4.81	46
GaP-NMF	-1.86	3.89	6.12	31

which is a woodwind quintet recording, consisting of bassoon, clarinet, flute, horn, and oboe. This piece has rich content across frequencies and various sound textures. The goal is to separate the signal on the instrument level. We compute DFTs of 1024 samples with 50% overlap.

To separate out different instruments, we need to filter out the audio signals belonging to different dictionary components. As in [5], given the complex spectrogram $\mathbf{X}^c$, to reconstruct the estimated complex spectrogram for the kth component $\hat{\mathbf{X}}^{(k)}$, we can apply Wiener filtering:

$$\hat{X}_{ft}^{(k)} = X_{ft}^{c} \frac{D_{fk}S_{kt}Z_{kt}}{\sum_{l=1}^{K} D_{fl}S_{lt}Z_{lt}} \tag{16}$$

There is no direct information to determine how the sources and instruments correspond. The heuristic in [7] is adopted: for each instrument, we pick the single component whose corresponding activation $\boldsymbol{s}_k \odot \boldsymbol{z}_k$ has the largest correlation with the power envelope of the single-track instrument signal. Note that the number of components inferred is larger than the number of instruments, thus the selected components only represent part of sources.

`Bss_eval` [14] is used to quantitatively evaluate the blind source separation. Table 1 lists the average SDR (Source to Distortion Ratio), SIR (Source to Interferences Ratio), and SAR (Sources to Artifacts Ratio) across instruments for BP-NMF and GaP-NMF (higher ratios are better). BP-NMF performs comparably well. BP-NMF decomposes the piece into 46 components, and GaP-NMF decomposes the piece into 31 components. We attribute this to the sparsity induced by BP-NMF's binary mask $\mathbf{Z}$; one needs a richer dictionary to explain the data with a sparse activation matrix.

4.4 Dictionary Quality Evaluation

The evaluation of latent component discovery and source separation is always difficult. We propose an evaluation mechanism similar to music transcription and provide statistically significant results.

To evaluate the model's ability to discover latent components from mixed signals, we can instead work on monophonic signals, which is a substantially simpler problem. We can then compare the results with those from mixed signal. If there is significant similarity, it indicates that the model can do equally well as it would have even if the problem were made artificially easier.

Since we have the single-track recordings for each instrument in the woodwind quintet recording from Section 4.3, we can apply BP-NMF to each of them separately and we will expect the learned dictionaries to be of high quality. We compute DFTs of 512 samples with no overlap. The number of learned components from each instrument is larger than the number of distinct notes V, thus only the top V components are selected according to the corresponding importance π_v. The selected components are shown in Figure 4a. The blocks are grouped according to instruments and sorted by approximated fundamental frequency. In the original piece, the bassoon is mostly playing low-pitch notes, while flute is playing high-pitch notes, both of which are reflected in the learned dictionaries.

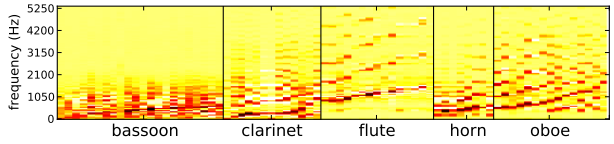
Now we would like to see if the results from BP-NMF on the mixed signal are similar to those from single-track recordings. Again there is no explicit information about the correspondence between the components learned from the mixed signal and the single-track signals. Thus, we adopt a greedy search which tries to match the dictionary components based on their correlation. BP-NMF discovers 29 components to describe the data [4], which is less than the number of distinct notes. Thus we only match the top 29 from components in Figure 4a.

After obtaining a one-to-one matching between dictionaries, we compute the correlations between the corresponding activations $\boldsymbol{s}_k \odot \boldsymbol{z}_k$. A box-and-whisker plot of correlations is shown in Figure 4b. As comparison, we also show the correlations from random matchings. Random matching has correlation close to 0, indicating there is no linear dependence. The minimum of the correlation from BP-NMF matching is close to 0 due to the fact that a few of the activations on the mixed signal are fairly sparse. But the overall quantiles do not overlap.

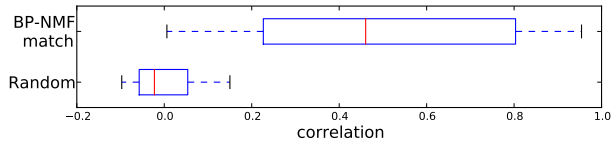
To formally test if the results from BP-NMF matching and random matching are significantly different, we apply hypothesis testing. Since we do not assume the correlations are normally distributed, a paired Wilcoxon signed-rank test [17], instead of a Student's t-test, is performed between the correlations from BP-NMF matching and random matching. The null hypothesis is that the correlations from BP-NMF matching and random matching come from the same population and we get p-value less than 0.01, which indicates that their difference is statistically significant. This gives a solid evidence that BP-NMF is able to learn dictionary components equally well in mixed signal, when compared with dictionaries learned from single-track instrument recordings.

We also apply the same procedures to IS-NMF and GaP-NMF. For IS-NMF, as we need to specify the number of components K, each single-track recording is decomposed with K equals the number of distinct notes. For the mixed signal, we set $K \in \{5, 10, 20, \cdots, 70\}$. When K is between 10 and 30, the Wilcoxon signed-rank test shows that the difference is significant at $p = 0.05$ level. For the rest of the K's, we get larger p-values and cannot reject the null hypothesis. GaP-NMF decomposes the data into 20 components and the test results show significant difference between GaP-NMF matching and random matching.

[4] This number is smaller than that in Section 4.3 because a smaller DFT size with no overlap is used, which leads to less data.



(a) The selected components learned from single-track instrument. For each instrument, the components are sorted by approximated fundamental frequency. The dictionary is cut off above 5512.5 Hz for visualization purposes.



(b) The box-and-whisker plot for the correlations from both BP-NMF matching and random matching. A paired Wilcoxon signed-rank test shows that they are significantly different.

Figure 4: The results from the proposed evaluation.

Therefore, this evaluation mechanism can also be applied to determine a range for the "proper" number of components to describe the data.

5. CONCLUSION

In this paper, we propose BP-NMF, a Bayesian nonparametric extension of nonnegative matrix factorization, which can automatically infer the number of latent components. BP-NMF explicitly assumes that some of the components are often completely silent. BP-NMF performs well under existing metrics and under a novel evaluation mechanism.

6. ACKNOWLEDGMENTS

The authors thank the reviewers for comments and the helpful discussion with Brian McFee and Colin Raffel. This work was supported by the NSF under grant IIS-1117015.

7. REFERENCES

[1] Samer A. Abdallah and Mark D. Plumbley. Polyphonic music transcription by non-negative sparse coding of power spectra. In *Proceedings of the 5th International Society for Music Information Retrieval Conference*, pages 10–14, 2004.

[2] Dhananjay Bansal, Bhiksha Raj, and Paris Smaragdis. Bandwidth expansion of narrowband speech using non negative matrix factorization. In *9th European Conference on Speech Communication (Eurospeech)*, 2005.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[4] Finale Doshi-Velez, Kurt T. Miller, Jurgen Van Gael, and Yee Whye Teh. Variational inference for the Indian buffet process. In *International Conference on Artificial Intelligence and Statistics*, 2009.

[5] Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence: with application to music analysis. *Neural Computation*, 21(3):793–830, 2009.

[6] Nils Lid Hjort. Nonparametric bayes estimators based on beta processes in models for life history data. *The Annals of Statistics*, pages 1259–1294, 1990.

[7] Matthew D. Hoffman, David M. Blei, and Perry R. Cook. Bayesian nonparametric matrix factorization for recorded music. In *Proceedings of the 27th Annual International Conference on Machine Learning*, pages 439–446, 2010.

[8] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[9] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*, 13:556–562, 2001.

[10] Alexey Ozerov and Cédric Févotte. Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(3):550–563, 2010.

[11] John Paisley and Lawrence Carin. Nonparametric factor analysis with beta process priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 777–784, 2009.

[12] Paris Smaragdis and Judith 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. IEEE, 2003.

[13] Romain Thibaux and Michael I. Jordan. Hierarchical beta processes and the Indian buffet process. In *International Conference on Artificial Intelligence and Statistics*, 2007.

[14] 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.

[15] Tuomas Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(3):1066–1074, 2007.

[16] Chong Wang and David M. Blei. Variational inference in nonconjugate models. *Journal of Machine Learning Research*, 14:899–925, 2013.

[17] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83, 1945.

SEMI-SUPERVISED POLYPHONIC SOURCE IDENTIFICATION USING PLCA BASED GRAPH CLUSTERING

Vipul Arora, Laxmidhar Behera
Department of Electrical Engineering, Indian Institute of Technology, Kanpur
vipular@iitk.ac.in, lbehera@iitk.ac.in

ABSTRACT

For identifying instruments or singers in the polyphonic audio, supervised probabilistic latent component analysis (PLCA) is a popular tool. But in many cases individual source audio is not available for training. To address this problem, this paper proposes a novel scheme using semi-supervised PLCA with probabilistic graph clustering, which does not require individual sources for training. The PLCA is based on source-filter approach which models the spectral envelope as a weighted sum of elementary band-pass filters. The novel graph based approach, embedded in the PLCA framework, takes into account various perceptual cues for characterizing a source. These cues include temporal cues like the evolution of F0 contours as well as the acoustic cues like mel-frequency cepstral coefficients. The proposed scheme shows better results in identifying vocal sources than a state of the art unsupervised scheme. In addition, the proposed framework can be used to incorporate perceptual cues so as to enhance the performance of supervised schemes too.

1. INTRODUCTION

We humans can selectively focus our attention on listening to a particular sound source even in the midst of many interfering sounds. This ability makes it possible for us to listen to the polyphonic music where we can intently hear a particular instrument (or singer) as if it were playing alone. In order to understand the nature of human cognition it is important to find out what features human ears identify in order to be familiar with and cognize a sound. In music, several perceptual attributes have been found to correspond to particular mathematical properties of the audio signal. For example, pitch approximately corresponds to the fundamental frequency (F0), loudness roughly corresponds to the amplitude/intensity etc. However, even if the pitch and loudness may vary during the song, we tend to recognize and cluster apart a particular musical source from the mixture of sounds, even if we have not listened to that source before. [9] discusses various acoustic attributes which help in perceptual grouping of various acoustic stimuli. This grouping is conceptualized as taking place at two levels. At the first level, the acoustic stimuli are grouped into group objects or the perceptual units, and at the second level, these group objects are linked to different source streams. Our present work mostly focusses on the second level of grouping, which relies upon the features which characterize a source. The first level of grouping in one time frame is characterised by the given F0 value and its harmonics.

There have been several works on finding the multiple pitches or fundamental frequencies (F0's) in a polyphonic audio, so much so that it is a popular task in the MIREX challenge [1] . This paper focuses on identifying the instruments associated with producing those F0's. One popular paradigm to quantify the quality of sound is the spectral envelope. There are several works in the literature that learn spectral envelopes for instrument identification in a supervised fashion. [7] uses sinusoidal modeling to represent the audio signal and groups the peaks over a few consecutive frames using heuristic auditory cues and spectral clustering. These clusters are then assigned to a source with the help of timbre models trained beforehand. [6] models the harmonic partials and temporal evolution using Gaussian mixture model (GMM). Instrument identification is performed using pre-trained support vector machine classifier over statistical features derived from the spectral and temporal parameters. A popular tool to analyze polyphonic sound is the non-negative matrix factorizarion (NMF). [12] uses a source-filter model based NMF for sound separation and subsequently, the instrument identification is carried out using pre-trained GMM's over Mel-Frequency cepstral coefficients.

These supervised methods are dependent on the availability of isolated sounds from each contributing source. But many times, the isolated source audio is not available, for example for commercial CD's. Hence, we have to resort to unsupervised techniques for source identification. This possibility is asserted by the human ability to cognize and cluster any singer from the polyphonic song even if the song or the singer is heard for the first time.

Several works aiming in this direction cluster the various sources in an unsupervised way by matching their timbral properties. [8] applies NMF to extract dictionary components, which are then grouped into sources by unsupervised clustering over the Mel-frequency cepstral co-

[1] http://www.music-ir.org/mirex/wiki/2012

efficients extracted from the dictionary components. [11] improves upon this method by using shifted NMF for the unsupervised clustering of the dictionary components. [5] uses agglomerative approach to cluster the matrices, obtained by the NMF decomposition of the input signal, into different sources. These works have their foundations rooted in the hypothesis that the sounds having similar spectral envelops originate from the same source.

The present work is an exploration of this hypothesis for identifying the sources present in a polyphonic audio in a semi-supervised way. A polyphonic audio, with several musical sources playing together, is given. Since this work focuses on identifying the underlying instrument, we assume the active fundamental frequencies to be given apriori, as they can be extracted using a multi-F0 extractor [3]. Our goal is to identify the instrument associated with each F0. The system is semi-supervised in the sense that a human user annotates a few segments of pitch contours with the corresponding source labels and feeds this information for initializing the system. The proposed algorithm then labels the entire audio with the corresponding instruments. A source-filter based Probabilistic Latent Component Analysis (PLCA) framework is used to decompose the given polyphonic audio spectrum into an appropriate representation in terms of various elementary spectra embedded in a Bayesian network of various lateral variables. The clustering is performed using probabilistic graph clustering framework. All the samples in the given F0 contours act as nodes of the graph and the edges connecting them are modelled by a similarity relation. The similarity between nodes is derived from various acoustic cues like pitch continuity, spectral characteristics and simultaneity constraints.

The major advantage of our approach is that it does not need individual source audios for training but requires only a few annotations per instrument. Hence this approach can be used even for unseen instruments and commercial recordings. Conceptually, the novelty of this work is that it applies graph clustering to the NMF framework. Graph based method helps to model the metrics (or distances) between the sound objects in a non-Euclidean way and hence gives a lot of flexibility to model the auditory space. The probabilistic graph based clustering can incorporate various perceptual cues which may help in grouping the sources, thereby enhancing the performance. In this work, we propose novel ways to model these constraints.

The polyphonic signal representation using PLCA framework is explained in Section 2. The graph based modeling of instrument similarity is proposed in Section 3 followed by the probabilistic graph clustering method in Section 4. Section 5 gives the complete overview of the proposed algorithm and the experimental evaluation of the same.

2. SIGNAL REPRESENTATION

The single channel polyphonic audio is considered to be a linear additive mixture of various source waveforms. The audio waveform is transformed to frequency domain by taking its 2048-point short-time Fourier transform with hanning window of length 56ms and hop size of 10ms. The magnitude spectrum is boosted by 6dB per octave. PLCA [10] assumes the magnitude spectrogram $V(f,t)$ of the polyphonic signal to be a histogram of *energy quanta* piled up in the time-frequency bins, indexed by t and f respectively, and generated by an underlying probability distribution function (pdf) $P_t(f)$. It further assumes that the magnitude spectrum of polyphonic audio is the linear sum of the magnitude spectra of individual source signals. Using the source-filter model [12], $P_t(f)$ can be factorized into the following latent variable representation:

$$P_t(f) = \sum_{p,s,z,a} P_t(f|p,a)P_t(p)P_t(s|p)P_t(z|p,s)P(a|s,z) \tag{1}$$

Here, p, s, z, a are latent variables underlying the generation of energy quanta at each time-frequency bin, which can take values from 1 to N_p, N_s, N_z, N_a, respectively. p indexes the pitch, s indexes the source instrument, z indexes the dictionary component and a indexes the frequency band associated with each t-f bin. $P_t(f|p,a)$ is the fixed spectrum consisting of the Gaussian harmonic peaks placed at the integer multiples of the given F0 associated with the pitch index p at time t and filtered through the ath triangular mel-frequency band pass filter. The other parameters can be learnt using the Expectation Maximization (EM) algorithm which minimizes the Kullback-Leibler divergence between the observed spectrum $V(f,t)$ and the underlying distribution model $P_t(f)$, as explained below.

E-step:

$$P_t(p,s,z,a|f) = \frac{P_t(f|p,a)P_t(p)P_t(s|p)P_t(z|p,s)P(a|s,z)}{P_t(f)} \tag{2}$$

M-step:

$$P_t(p) = \frac{\sum_{f,s,z,a} V(f,t)P_t(p,s,z,a|f)}{\sum_p \sum_{f,s,z,a} V(f,t)P_t(p,s,z,a|f)} \tag{3}$$

$$P_t(s|p) = \frac{\sum_{f,z,a} V(f,t)P_t(p,s,z,a|f)}{\sum_s \sum_{f,z,a} V(f,t)P_t(p,s,z,a|f)} \tag{4}$$

$$P_t(z|p,s) = \frac{\sum_{f,a} V(f,t)P_t(p,s,z,a|f)}{\sum_z \sum_{f,a} V(f,t)P_t(p,s,z,a|f)} \tag{5}$$

$$P(a|s,z) = \frac{\sum_{f,t,p} V(f,t)P_t(p,s,z,a|f)}{\sum_a \sum_{f,t,p} V(f,t)P_t(p,s,z,a|f)}. \tag{6}$$

Also, we assume the total number of instruments N_s to be known apriori. Our goal is to cluster all the (t,p) units based on the underlying sources s. This problem can be seen as that of estimating $P_t(s|p)$.

3. INSTRUMENT SIMILARITY MODELING

After suitably modeling the spectra, the next task is to cluster them based upon their originating sources. In order to cluster the sources, we need to know what features quantify the distinction among the spectra of different sources and the similarity between those of the same source.

All the objects to be clustered, i.e. the (t, p) units, are modeled as the nodes of a graph. The edges of the graph are modeled by the link-weights matrix A, such that $A_{ki;lj}$ represents the closeness between the pair of nodes (t_k, p_i) and (t_l, p_j). Here, t_k implies that the time index is k and p_i implies that the pitch index is i. The link-weights are constructed to lie within the range $[0, 1]$. A larger value of the link-weight denotes more closeness between the pair of nodes, i.e. it is more likely that both the nodes correspond to the same cluster.

To model the closeness between the nodes, we consider three factors which quantify the similarity between the spectra from same sources, namely, the spectral envelope, the temporal continuity of the pitch contours and the simultaneity constraint. Each one of these is explained as follows.

3.1 Spectral Envelope

This step uses Mel-frequency cepstral coefficients (MFCC's) in order to measure spectral-timbral closeness of sounds. Although the source-filter PLCA is also based on spectral envelope, but MFCC features provide different advantages, like de-correlating the filter bank weights. MFCC features have been very successful in the areas of speech research like speaker characterization [4]. The use of MFCC features for source characterization is based on the hypothesis that the spectra from the same source have similar spectral envelopes. The MFCC's quantify the position of the node (t, p) in the space of spectral envelopes. They are computed from the PLCA representation as follows.

The ath band-filtered spectrum associated with pitch index p at time t is estimated using the following Weiner filter,

$$V_{p,a}(f,t) = \frac{P_t(f,p,a)}{P_t(f)} V(f,t) \tag{7}$$

where, $P_t(f)$ is given by Equation (1) and

$$P_t(f,p,a) = \sum_{s,z} P_t(f|p,a) P_t(p) P_t(s|p) P_t(z|p,s) P(a|s,z)$$

The energy values in the N_a sub-band spectra associated with a (t, p) unit are used to compute the 13 dimensional MFCC vectors $\mathbf{m}_{t,p}$. The closeness between two nodes, based on these features, is modeled using the cosine similarity measure which is defined as

$$\mathcal{D}_{\cos}(\mathbf{m}_1, \mathbf{m}_2) = \frac{\mathbf{m}_1 \cdot \mathbf{m}_2}{|\mathbf{m}_1||\mathbf{m}_2|} \tag{8}$$

where, $\cdot$ represents the vector dot product and $|\mathbf{m}|$ stands for the Euclidean norm of vector $\mathbf{m}$. The link-weight matrix is estimated as

$$A^{\mathcal{S}}_{ki;lj} = \frac{\mathcal{D}_{\cos}(\mathbf{m}_{t_k,p_i}, \mathbf{m}_{t_l,p_j}) + 1}{2} \tag{9}$$

with the superscript $\mathcal{S}$ denoting that it has been derived from the spectral features. The aforementioned equation ensures that although $\mathcal{D}_{\cos} \in [-1, 1]$, $A^{\mathcal{S}}$ is constrained to lie in the interval $[0, 1]$.

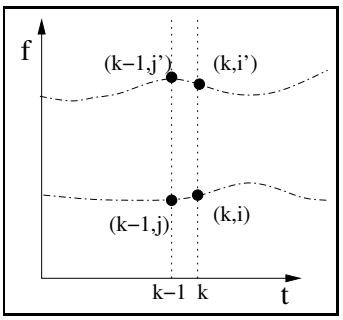


Figure 1. Temporal Continuity: dashed-dotted line shows two pitch contours.

3.2 Temporal Continuity

The pitch contour produced by a musical source changes slowly due to physical as well as musicological constraints. We can use this information as a cue [13] for clustering up the (t, p) units by forming the link-weight matrix $A^{\mathcal{T}}$, where the superscript $\mathcal{T}$ denotes the temporal connectedness measure. The closeness between F0s is modeled as

$$d^{lj}_{ki} = \exp\{-(F0_{ki} - F0_{lj})^2/\sigma^2\}. \tag{10}$$

For $A^{\mathcal{T}}_{ki;lj}$ to be large, we need to satisfy the following constraints:

1. $|t_k - t_l| = 1$
2. If d^{lj}_{ki} is large, $d^{lj'}_{ki}$ should be small, for $j' \neq j$, in order to avoid the ambiguity when the F0s at both the nodes are close to each other. For the same reason, $d^{ki'}_{ki}$ should also be small, for $i' \neq i$ (see Figure 1 with $l = k - 1$).
3. If (t, p) is well connected to (t', p') which is further well connected to (t'', p''), then (t, p) should also be well connected to (t'', p''). We call this as agglomeration.

Constraint 2 ensures that the temporal connectedness is strong only when the F0 trajectory under consideration is far apart in frequency from the other F0 trajectories.

In accordance with the above factors we devise the following novel scheme to construct $A^{\mathcal{T}}$.

for $k = 2$ to N_t **do**

$$A^{\mathcal{T}}_{ki;(k-1)j} = d^{(k-1)j}_{ki} \left[\frac{d^{(k-1)j}_{ki}}{\sum_j d^{(k-1)j}_{ki} + \sum_{j \neq i} d^{kj}_{ki}} \right] \tag{11}$$

{Agglomeration:}
if $A^{\mathcal{T}}_{ki;(k-1)j} > \epsilon$ **then**
$A^{\mathcal{T}}_{ki;k'i'} \leftarrow A^{\mathcal{T}}_{(k-1)j;k'i'}, k' = \{1, ..., (k-2)\}, \forall i'$
end if
{Making $A^{\mathcal{T}}$ symmetric:}
$A^{\mathcal{T}}_{k'i';ki} \leftarrow A^{\mathcal{T}}_{ki;k'i'}, \forall k', \forall i'$
end for

3.3 Simultaneity Constraint

We assume that an instrument can play only a single note at a time. This assumption is undoubtedly valid for the vocal sources. For instruments, this assumption is not valid for harmony-based music systems where same instrument plays many notes simultaneously to form chords, but is valid for melody-based music systems where an instrument plays only one note at a time. This constraint implies that $N_s = N_p$.

The final link-weight matrix A is formed by combining $A^{\mathcal{S}}$ and $A^{\mathcal{T}}$ as,

$$A = (1-\alpha)A^{\mathcal{S}} + \alpha A^{\mathcal{T}} \tag{12}$$

where, $\alpha \in [0, 1]$. The simultaneity constraint is imposed as,

$$A_{ki;kj} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Another implication of the simultaneity constraint in graph clustering is mentioned in the next section, viz., Section 4.

4. GRAPH CLUSTERING

For clustering, several techniques are used like agglomerative clustering, probabilistic clustering, spectral clustering etc. In this work, we use the probabilistic framework for graph clustering as proposed by [1]. The clustering problem is seen as an optimization problem which aims at maximizing the log-likelihood function,

$$\mathcal{L}(\Omega, A) = \sum_{s} \sum_{k,l,i,j} \ln P(\omega^s_{ki}, \omega^s_{lj} | A_{ki;lj}) \tag{14}$$

where, $\omega^s_{ki} \in \Omega$ is the cluster membership function which measures the degree of affinity of the unit (t_k, p_i) to the sth source cluster. We can see that ω^s_{ki} corresponds to $P_{t_k}(s|p_i)$. $P(\omega^s_{ki}, \omega^s_{lj}|A_{ki;lj})$ is modeled as a Bernoulli distribution,

$$P(\omega^s_{ki}, \omega^s_{lj}|A_{ki;lj}) = (A_{ki;lj})^{\omega^s_{ki}\omega^s_{lj}}(1 - A_{ki;lj})^{1-\omega^s_{ki}\omega^s_{lj}} \tag{15}$$

This equation ensures that if the link weight $A_{ki;lj}$ is large, then there is a large probability of ω^s_{ki} and ω^s_{lj} to be close to unity, meaning that the two nodes get clustered into the same source s. On putting this in Equation (14) and maximising with respect to ω^s_{ki}, we get,

$$\frac{\partial \mathcal{L}(\Omega, A)}{\partial \omega^s_{ki}} = \sum_{lj} \omega^s_{lj} \ln \frac{A_{ki;lj}}{1 - A_{ki;lj}} \tag{16}$$

The updated cluster membership function is estimated using soft-assign ansatz as

$$\hat{\omega}^s_{ki} = \frac{\exp[\partial \mathcal{L}/\partial \omega^s_{ki}]}{\sum_s \exp[\partial \mathcal{L}/\partial \omega^s_{ki}]} \tag{17}$$

This equation also takes care of the fact that ω^s_{ki} represents $P_{t_k}(s|p_i)$ and hence ought to be normalized with respect to s.

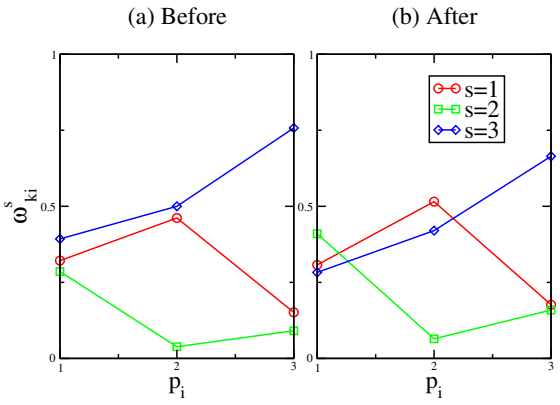


Figure 2. ω^s_{ki} vs p_i for different values of s and a fixed t_k. (a) All p_i are strongly linked to the cluster $s = 3$; (b) Equation (18) imposes the simultaneity constraint.

Under this formulation, it is quite possible that both ω^s_{ki} as well as ω^s_{kj}, for some $j \neq i$, have large values. However, the simultaneity constraint prevents this from happening. This is achieved by considering the point-cluster relation in a two-way sense, i.e. how simultaneous (t, p) units link to one cluster as well as how many clusters relate to one such unit. The simultaneity constraint is imposed in a non-rigid way by simply updating the cluster membership function as,

$$\omega^s_{ki} \leftarrow \frac{\omega^s_{ki}}{\sqrt{\sum_j \omega^s_{kj}}}, \tag{18}$$

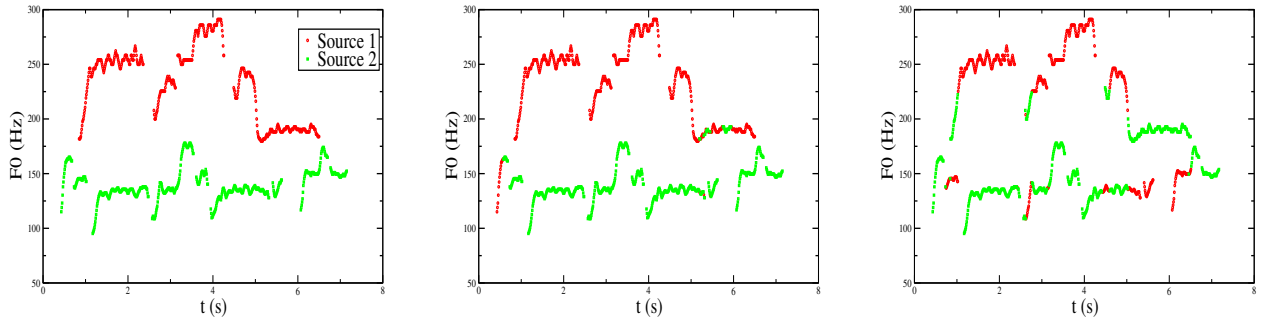
followed by normalization with respect to s, so as to maintain its probabilistic interpretation as $P_t(s|p)$. To understand this equation, let us consider the binary clustering case with $N_s = 2$. For a certain k, if $\omega^{s_1}_{kj} > \omega^{s_2}_{kj}, \forall j$, it implies that both the simultaneous units, $(t_k, p_j), j = \{1, 2\}$, are strongly associated with the source s_1. This situation implies that $\sum_j \omega^{s_1}_{kj} > 1$ and $\sum_j \omega^{s_2}_{kj} < 1$. The simultaneity constraint however imposes that only one p should be associated with one s. Hence, we attenuate $\omega^{s_1}_{kj}, \forall j$ and amplify $\omega^{s_2}_{kj}, \forall j$ by simply dividing with $\sqrt{\sum_j \omega^s_{kj}}$. This effect is also achieved for larger values of N_s, as graphically depicted in the Figure 2.

5. EXPERIMENTS

5.1 Implementation of Complete Algorithm

Given a polyphonic song along with all the active pitch values indexed by p at time frame index t, our task is to cluster the (t, p) units into the underlying sources. Instead of completely unsupervised clustering, we make the task semi-supervised by randomly choosing a few (3 here) time indices and labeling the (t, p) units at those times with their respective sources. This is the only little annotation work which has to be manually performed in case of absence of any other training data.

The source dictionaries $P(a|s, z)$ are randomly initialized and pre-trained using this little annotated data from



(a) Ground Truth (b) Proposed Algorithm (c) Algorithm [8]

Figure 3. Example of Source Clustering for a mixture of Ken-Gen singer pair. (a) Ground truth (b) the proposed algorithm and (c) the algorithm of [8]

the same song. These source dictionaries are then used as seeds for the complete algorithm as explained below.

1. PLCA decomposition of the entire song is performed, using the EM algorithm Equations (2)-(6), iterated 10 times.

2. The spectral and temporal link-weight matrices are computed as explained in Section 3.

3. The probabilistic source labels ω^s_{ki} are initialized as $P_{t_k}(s|p_i)$ and are updated using graph clustering as explained in Section 4. Then, $P_t(s|p)$ is updated as

$$P_{t_k}(s|p_i) \leftarrow \frac{P_{t_k}(s|p_i) + \omega^s_{ki}}{2}$$

because generally ω^s_{ki} is found to have extreme values (0 or 1) due to its exponential update rule.

4. These steps 1 to 3 are repeated for a fixed number of times (3 here).

Finally, the source index is estimated using maximum aposteriori estimator along with the simultaneity constraint as

$$\hat{\mathbf{s}}_t = \arg\max_{\mathbf{s}_t}\{\sum_p P_t(\mathbf{s}_t(p)|p)\}. \quad (19)$$

Here, $\mathbf{s}_t$ is the vector formed by permutations of the vector $[1, ..., N_s]$, so that the final instrument labels are $\hat{s}_{t,p} = \hat{\mathbf{s}}_t(p)$.

Notably, the $P_t(s|p)$ for the given (t, p) units are fixed to the given values throughout the algorithm. The various parameters were chosen heuristically. Number of dictionary components for each source is set as $N_z = 2$ and the number of mel-frequency band filters $N_a = 20$. In subsection 3.2, we set $\sigma = 20$ and $\epsilon = 0.6$.

5.2 Evaluation

The clustering of vocal sources is more challenging than that of most of the musical instruments. The reason is that for a certain F0, the spectrum of the latter remains mostly the same but that of the former varies significantly due to the variety of voiced sounds (or vowels) which can be produced by a human singer.

The proposed algorithm was evaluated with the help of vocal songs from the MIR-1k database [2], which is publicly available. The evaluation dataset consisted of audio recordings of 4 different singers - 2 males (Abj, Gen) and 2 females (Ken, Hey). Total 60 single-channel audio mixtures of polyphony order $N_s = 2$ were formed by linearly adding the monophonic audio waveforms, with a total duration of about 350 seconds.

The proposed approach was compared with that proposed by Spiertz and Gnann [8], whose implementation is available as open source. Their method is oriented to produce separated signals instead of instrument clusters. To estimate their performance on instrument identification, we estimated $P(s|p, t)$ by comparing the spectra using the cosine similarity measure, defined in Equation (8).

$$\hat{P}^{SG}(s|p, t) = \mathcal{D}_{\cos}(\mathbf{Y}_{p,t}, \mathbf{S}_{s,t}) \quad (20)$$

Here, $\mathbf{Y}_{p,t}$ and $\mathbf{S}_{s,t}$ are the magnitude spectral vectors of the pth output channel and the sth monophonic source used to construct the input, respectively, at time frame t.

For evaluation metrics, all the (t, p) units are clustered into N_s number of groups. Each obtained cluster is linked to a ground truth source in a way which maximizes the classification accuracy. The classification accuracy is measured as the fraction of (t, p) units correctly labeled with the ground truth source.

The evaluation scores are presented in Table 1. We can see that the proposed semi-supervised algorithm performs better than the unsupervised algorithm. Although this comparison is not well balanced, but still it gives an idea of the performance. Also, we can see that for our algorithm, the male-female voice differentiability is larger than the other two cases, viz., male-male and female-female.

This algorithm works quite well with short audio excerpts, as we tested it for upto 10s long excerpts, i.e. the size of A was upto 2000×2000. But for long songs,

Singers pair	Classification Accuracy in %	
	Proposed algorithm	Algorithm [8]
Abj-Hey	79.5 (16.5)	62.7 (15.0)
Ken-Hey	72.9 (13.7)	68.3 (13.5)
Gen-Hey	86.2 (13.0)	65.4 (11.2)
Abj-Ken	79.5 (16.8)	74.3 (9.9)
Abj-Gen	68.1 (13.5)	66.0 (10.0)
Ken-Gen	87.2 (9.9)	71.4 (14.8)

Table 1. Evaluation Results with standard deviations in brackets

the size of the link-weight matrix becomes very large and hence the computations become extensive. Thus, it is wise to break the long song into shorter segments (< 10s in duration). In such a case, one may pre-train the dictionaries for each audio segment using the few labeled (t, p) units, even if they fall in different audio segments, and perform the proposed algorithm over one segment at a time.

6. CONCLUSION

This paper proposed a novel algorithm for instrument identification in polyphonic music. The major advantage of the algorithm is that it is semi-supervised as it does not require any training data in the form of individual source audios. It can initialize from annotations only at a few time instants. The method uses source-filter based PLCA for decomposing the magnitude spectra of the polyphonic audio. This PLCA based representation is used to form a graph which is clustered into the underlying sources with the help of various perceptual cues, including the spectral and temporal cues. Novel ways of modeling these cues were proposed. The temporal constraints take care of not only the continuity of the pitch contour individually but also its interaction with other pitch contours. To model the simultaneity constraints, a new equation is introduced which takes care of the point-cluster relation in a two-way sense, so that the simultaneously occuring points are assigned to different clusters.

The experimental results show that the proposed algorithm performs better than a state of the art unsupervised source separation algorithm adapted for instrument identification. The proposed framework can also be used with supervised schemes so as to enhance the performance by incorporating various acoustic cues.

7. REFERENCES

[1] A. Robles-Kelly, and E. R. Hancock: "A Probabilistic Spectral Framework for Grouping and Segmentation," *In Pattern Recognition*, Vol. 37, No. 7, pp. 1387–1405, 2004.

[2] C.-L. Hsu, and J.-S. R. Jang: "On the Improvement of Singing Voice Separation for Monaural Recordings Using the MIR-1K Dataset," *IEEE Transactions on Audio, Speech and Language Processing*, Vol. 18, No. 2, pp. 310–319, 2010.

[3] C. Yeh, A. Roebel, and X. Rodet: "Multiple fundamental frequency estimation and polyphony inference of polyphonic music signals," *IEEE Transactions on Audio, Speech and Language Processing*, Vol. 18, No. 6, pp. 1116-1126, 2010.

[4] D. A. Reynolds: "An overview of automatic speaker recognition technology," *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vol. 4, pp. 4072–4075, 2002.

[5] J. M. Becker, M. Spiertz, and V. Gnann: "A probability-based combination method for unsupervised clustering with application to blind source separation," *Latent Variable Analysis and Signal Separation (LVA/ICA)*, pp. 99-106, 2012.

[6] J. Wu, E. Vincent, S. A. Raczynski, T. Nishimoto, N. Ono, and S. Sagayama: "Polyphonic pitch estimation and instrument identification by joint modeling of sustained and attack sounds," *IEEE Journal of Selected Topics in Signal Processing*, Vol. 5, No. 6, pp. 1124–1132, 2011.

[7] L. G. Martins, J. J. Burred, G. Tzanetakis, and M. Lagrange: "Polyphonic Instrument Recognition Using Spectral Clustering," *Proceedings of International Society for Music Information Retrieval Conference (ISMIR)*, 2007.

[8] M. Spiertz, and V. Gnann: "Source-filter based clustering for monaural blind source separation," *Proceedings of International Conference on Digital Audio Effects*, 2009.

[9] M. Weintraub: "A theory and computational model of monaural auditory sound separation," *Ph. D. dissertation,* Stanford University, 1985.

[10] P. Smaragdis, B. Raj, and M. Shashanka: "A probabilistic latent variable model for acoustic modeling," *Advances in models for acoustic processing, NIPS*, 2006.

[11] R. Jaiswal, D. FitzGerald, D. Barry, E. Coyle, and S. Rickard: "Clustering NMF basis functions using shifted NMF for monaural sound source separation," *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 245–248, 2011.

[12] T. Heittola, A. Klapuri, and T. Virtanen: "Musical instrument recognition in polyphonic audio using source-filter model for sound separation," *Proceedings of International Society for Music Information Retrieval Conference (ISMIR)*, 2009.

[13] V. Arora, and L. Behera: "On-line melody extraction from polyphonic audio using harmonic cluster tracking," *IEEE Transactions on Audio, Speech and Language Processing,* Vol. 21, No. 3, pp. 520–530, 2013.

Oral Session 6:

Listeners

AN EXPERIMENT ABOUT ESTIMATING THE NUMBER OF INSTRUMENTS IN POLYPHONIC MUSIC: A COMPARISON BETWEEN INTERNET AND LABORATORY RESULTS

Michael Schoeffler, Fabian-Robert Stöter, Harald Bayerlein, Bernd Edler, Jürgen Herre
International Audio Laboratories Erlangen
michael.schoeffler@audiolabs-erlangen.com

ABSTRACT

Internet experiments in the fields of music perception and music information retrieval are becoming more and more popular. However, not many Internet experiments are compared to laboratory experiments, the consequence being that the effect of the uncontrolled Internet environment on the results is unknown. In this paper the results of an Internet experiment with 1168 participants are compared to those of the same experiment with 62 participants but previously conducted in a controlled environment. The comparison of the Internet and laboratory results enabled us to make a point on whether the Internet can be used for our experiment procedure. The experiment aimed to investigate the listeners ability to correctly estimate the number of instruments being played back in a given excerpt of music. The participants listened to twelve short classical and pop music excerpts each composed using one to six instruments. For each music excerpt the participants were asked how many instruments they could hear and how certain they were about their estimation.

1. INTRODUCTION

In psychoacoustics, a sequence of sounds grouped by the auditory system is known as an "auditory stream" which was coined by Bregman and Campbell [2]. In the past decades, a lot of experimental work related to "auditory streams" has been done [1]. A majority of these experiments were psychoacoustically motivated, e. g. the stimuli used were mostly of simple type like sinusoids or noises. Especially from a psychoacoustic point of view, music is a very complex sound signal which contains high-level information (e. g. instrumentation and song lyrics). When listening to music, this high-level information is also mentally processed by humans. For developing auditory models, it could be helpful to know the maximum number of instruments humans are able to estimate when listening to music.

For many types of music perception experiments such as estimating the number of instruments, it is essential that the selected participants represent a large population. As recent research in cross-cultural music perception and cognition reveals: The perception of music is dependent on the origin of people [13]. Besides the cultural background, other aspects like their profession might have an influence when estimating the number of instruments being played back, e. g. musicians might recognize instruments much more easily since they are in touch with instruments in their everyday life. One of the advantages of Internet experiments (also called web-based experiments or web experiments) is that it is easier to gather participants with different backgrounds and from different regions than in laboratory experiments. For a summary of benefits and disadvantages of Internet experiments see [10]. In music perception a major argument against Internet experiments is that there is no control about the environment. With the spreading of mobile devices with Internet connections this argument becomes more apparent, since the environment of the participants can range from a quiet place at home to a noisy place outside.

By comparing the results of the Internet experiment presented in this paper to the results of the same experiment but previously conducted in a controlled environment [14], we contribute to answering the research question, whether the Internet can be used for experiments in music perception. Furthermore, subpopulations like headphones-users and loudspeaker-users are examined whether they lead to more reliable responses.

2. RELATED WORK

The ability of estimating the number of instruments is probably related to the ability of auditory stream segregation. An overview of auditory stream segregation in general is given by Bregman [1] and Wang and Brown [15].

In 1989, Huron conducted a musically motivated experiment related to stream segregation [8]. In his experiment he asked the participants for the number of voices in excerpts of organ music. Huron defined a voice as a single "line" of sound, more or less continuous, that maintains a separate identity in a sound field or musical texture (an overview of voice definitions is given by Cambouropoulos [3]). Huron came to the conclusion that the number of correctly identified voices is up to three. Based on Hurons work, we carried out an experiment where we asked musi-

cians and non-musicians how many instruments they could hear in short pieces of music [14]. In contrast to Huron we addressed a more general case where voices are played by different instruments and not only by organ. In Huron's experiment the responses of the participants were time-varying for each stimulus. Another difference is that in Huron's experiment all six participants except one had a musical background whereas our experiment was also attended by a large group of non-musicians.

It has been becoming more and more popular to use the Internet for experiments in music or auditory perception. One of the first auditory experiments were conducted by Welch and Krantz [16] in 1996. A web experiment related to MIREX tasks using *Amazon Mechanical Turk* has been conducted by Lee [9]. An experiment with a large attendance was done by Salganik *et al.* [12] in 2006. They had over 14.000 participants and examined the social influence on participants in an artificial music market. An overview of recent Internet experiments is given by Reips [11].

3. METHOD

3.1 Stimuli

For the stimuli generation, MIDI files of music pieces from the RWC database [7] were selected. The MIDI files were modified so that each file has a specific number of instruments being played back. Since the number of instruments being played back had to be as constant as possible, a so-called "instrumental stationarity" for each music piece was calculated. The "instrumental stationarity" is a measure that shows whether all instruments are played the whole time for a given start position and length. See [14] for the detailed equation of "instrumental stationarity" and more information about the stimuli generation. The RWC files were manually filtered a priori to exclude items dominated by lead instruments or singing voices. From the remaining items thirteen excerpts of MIDI files were selected with the help of the "instrumental stationarity" having one to six simultaneously playing instruments. Table 1 shows the selected excerpts including their instrumentation. The duration of each excerpt is around seven seconds. By cutting at note offsets we varied the lengths of the excerpts to make them semantically more meaningful. Six items (notated as C0**) belong to the classical Western music genre whereas the other items are of mixed genre. The MIDI excerpts were humanized and rendered by a sequencer software utilizing state-of-the-art commercial sampling products into WAV files. The rendered files were processed with convolutive reverb to match the original recordings. In informal listening tests the quality of the renderings was evaluated. In addition, participants did not give negative feedback about the artificialness of the items during the laboratory experiment. Additionally a loudness normalization was applied according to EBU-R128 [5]. To avoid spatial cues the files were downmixed to mono at 16 bit/44.1 kHz.

RWC ID	Start [s]	Dur. [s]	Instrumentation	$\sum$
J021	46.5	6.6	Piano, Contrabass (pizz.) and Trumpet	3
C001	0.0	9.0	Bassoon	1
G047	35.3	8.3	Violoncello	1
C016	0.9	7.6	Viola and Violoncello	2
G068	132.4	6.6	Violin and Flute	2
C018	240.4	5.4	French Horn, Piano and Violin	3
G046	0.3	7.9	Contrabass, Piano and Violoncello	3
C013	5.6	6.0	Flute, Viola, Violin and Violoncello	4
G036	0.0	6.5	Acoustic Guitar, Electric Bass, Piano and Violin	4
C012	112.0	6.0	Contrabass, Flute, Viola, Violin and Violoncello	5
G037	67.1	7.0	Acoustic Guitar, Contrabass (pizz.), Flute, Piano and Tenor Sax	5
C001	147.8	6.0	Bassoon, Clarinet, Contrabass, French Horn, Oboe and Violin	6
G028	17.5	6.5	Electric Bass, Electric Guitar, Flute, Piano, Trombone and Trumpet	6

Table 1. Selected items from the RWC Music Database [7]. Item *J021* was used as training item.

3.2 Participants

Participation in the experiment was done by visiting the experiment's website [1] . The experiment was promoted in mailing lists, forums, social networks and by personal invitations. Most of the forums and mailing lists were audio-related. No material incentive was given to participants. For motivating the participants a high score was added to the experiment.

In total 1310 website visitors attended the experiment. We identified 115 of them as participants who did the experiment more than once by using a browser fingerprinting method (for more details in browser fingerprinting, see [6]). In this case, only the first trial is used in the results analysis. Our browser fingerprinting method created a hash value by using the visitor's browser settings, e. g. screen resolution and installed plugins. In addition, we excluded 27 participants since they gave at least one non-serious response. We defined responses with zero instruments (25 participants) and responses with more than 12 instruments (two participants) as invalid. After the screening we had 1168 valid participants.

The participants were asked by a questionnaire whether they have a professional background in audio, play at least one instrument (including singing) and are familiar with listening tests. Detailed information about the participants are described in Table 2. Headphones were used by 571 participants and loudspeakers were used by 597 participants.

3.3 Materials and Apparatus

The main functionality of the experiment's website was written in HTML5 and JavaScript. The website was tested for all major web browsers and optimized for mobile devices and desktop computers. The default file format for the stimuli was WAV. Since some browsers (e. g. Internet Explorer) did not support WAV, MP3 (encoded with 256 kbits/s CBR with Fraunhofer Encoder) was used as alternative file format. The alternative file format was only

[1] `http://www.audiolabs-erlangen.com/experiments/wice/`

Total	Age group		Musician		Professional	
1168	0	[0-12]	-		-	
	110	[13-19]	74	[yes]	12	[yes]
					62	[no]
			36	[no]	0	[yes]
					36	[no]
	889	[20-39]	463	[yes]	128	[yes]
					335	[no]
			426	[no]	46	[yes]
					380	[no]
	143	[40-59]	98	[yes]	32	[yes]
					66	[no]
			45	[no]	8	[yes]
					37	[no]
	26	[60+]	13	[yes]	6	[yes]
					7	[no]
			13	[no]	2	[yes]
					11	[no]

Table 2. Information about the participants.

used when WAV was not supported by the browser.

3.4 Procedure

The experiment started on February the 15th, 2013 and lasted until March the 15th, 2013.

At first, participants filled out a short questionnaire. They were asked which audio setup they are using, whether they regularly play any musical instruments or do singing, have a background in professional audio, are familiar with listening tests and which age group they belong to.

After filling out the questionnaire, the participants did a short training. The purpose of the training was to familiarize the participants with the user interface and to give them the option to adjust the volume. The training had one stimulus with three instruments being played. The instruments were piano, bass and trumpet. The participants were told on the training page how many and which instruments are played back. During the training it was possible to listen to the stimulus unlimited times.

Followed by the training, the participants had to estimate the number of instruments being played in twelve stimuli. The experiment question was "How many different instruments do you hear?". Participants could listen to each stimulus up to three times. In addition, they were asked how certain they were in their response. They could choose between "uncertain", "certain" and "very certain". The user interface is shown in Figure 1.

After the participants estimated the number of instruments for all twelve stimuli, they were given a score based on their performance. Besides their personal score, a percentile rank showed how each participant performed compared to all the other participants.

4. RESULTS

The independent variables are the number of instruments being played back (Num_{Inst}), whether a participant is musical (*Musical*), professional in audio (*Professional*) and which setup was used (*Setup*). A participant is defined as

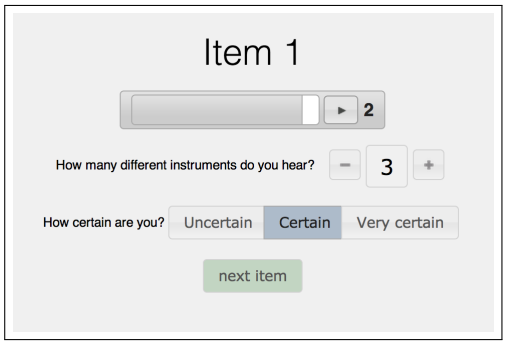


Figure 1. Experiment User Interface.

	Num_{Inst}						
Resp	$I=1$	$I=2$	$I=3$	$I=4$	$I=5$	$I=6$	n
$R=1$	2025	373	5	18	13	12	2446
$R=2$	298	1642	810	736	451	382	4319
$R=3$	12	277	1343	1145	1093	1069	4939
$R=4$	1	43	158	386	645	680	1913
$R=5$	0	1	18	48	120	155	342
$R=6$	0	0	1	3	12	30	46
$R>6$	0	0	1	0	2	8	11
	14016 responses (1168 participants · 12 items)						
Probability of $Resp_{\mathrm{Correct}}$	0.87	0.70	0.57	0.17	0.05	0.01	

Table 3. Responses from the participants. The cells with a gray background represent correct estimations.

musical ($Musical = true$) when he or she is regularly playing an instrument (including singing). The same applies to being professional ($Professional = true$) which is set when the participant responded that he or she is a professional in audio. The responses for the setup used can either be headphones ($Setup = `headphones'$) or loudspeaker ($Setup = `loudspeaker'$). The dependent variable is the participant's estimation of the number of instruments being played back (*Resp*). A correct estimation is defined as

$$Resp_{\mathrm{Correct}} = \begin{cases} 0 & \text{if } Num_{\mathrm{Inst}} \neq Resp \\ 1 & \text{if } Num_{\mathrm{Inst}} = Resp \end{cases}. \qquad (1)$$

Table 3 shows the responses of the participants for all stimuli.

For testing hypotheses, a logistic regression model with the response variable $Resp_{\mathrm{Correct}}$ and the predictor variables Num_{Inst}, *Musical*, *Professional* and *Setup* was calculated. The estimated coefficients, p-values and average marginal effects are shown in Table 4. Average marginal effects in the regression model describe the increase in probability for correctly estimating the number of instruments when the predictor variable is increased by one level. Compared to the other coefficients the average marginal effect of $Setup = `headphones'$ is very low. By using headphones instead of loudspeakers it is 1.35% more likely to estimate the number of instruments correctly.

As expected, participants who play an instrument or do singing ($Musical = true$) performed slightly better

Coefficient	Estimate	Std. Error	z-value	p-value	Average Marginal Effects
(Intercept)	1.5015	0.06836	21.963	$<$ 2e-16	0.1886
$Num_{\mathrm{Inst}} = 2$	-1.0293	0.07651	-13.453	$<$ 2e-16	-0.1293
$Num_{\mathrm{Inst}} = 3$	-1.6021	0.07463	-21.466	$<$ 2e-16	-0.2012
$Num_{\mathrm{Inst}} = 4$	-3.5674	0.08380	-42.572	$<$ 2e-16	-0.4481
$Num_{\mathrm{Inst}} = 5$	-4.8759	0.11290	-43.188	$<$ 2e-16	-0.6125
$Num_{\mathrm{Inst}} = 6$	-6.3061	0.19428	-32.459	$<$ 2e-16	-0.7922
Musical $= true$	0.5266	0.04932	10.676	$<$ 2e-16	0.0661
Professional $= true$	0.3306	0.06234	5.303	1.14e-07	0.0415
Setup = '$headphones$'	0.1071	0.04823	2.220	0.0264	0.0135

(Dispersion parameter for binomial family taken to be 1)
Null deviance: 18816 on 14015 degrees of freedom
Residual deviance: 11036 on 14007 degrees of freedom
AIC: 11054
Number of Fisher Scoring iterations: 7
McFadden's Pseudo R-squared: 0.413

Table 4. Logit regression model for response variable $Resp_{\mathrm{Correct}}$ calculated from the data obtained by the Internet experiment.

Coefficient	Estimate	Std. Error	z-value	p-value	Average Marginal Effects
(Intercept)	2.0226	0.11718	17.260	$<$ 2e-16	0.2590
$Num_{\mathrm{Inst}} = 2$	-1.0135	0.07424	-13.652	$<$ 2e-16	-0.1298
$Num_{\mathrm{Inst}} = 3$	-1.5914	0.07223	-22.033	$<$ 2e-16	-0.2038
$Num_{\mathrm{Inst}} = 4$	-3.4786	0.08031	-43.316	$<$ 2e-16	-0.4454
$Num_{\mathrm{Inst}} = 5$	-4.8058	0.10919	-44.015	$<$ 2e-16	-0.6154
$Num_{\mathrm{Inst}} = 6$	-6.2481	0.19033	-32.827	$<$ 2e-16	-0.8000
Environment = 'web'	-0.1435	0.10569	-1.358	0.174	-0.0184

(Dispersion parameter for binomial family taken to be 1)
Null deviance: 19826 on 14759 degrees of freedom
Residual deviance: 11819 on 14753 degrees of freedom
AIC: 11833
Number of Fisher Scoring iterations: 7
McFadden's Pseudo R-squared: 0.404

Table 5. Logit regression model for response variable $Resp_{\mathrm{Correct}}$ calculated from the data obtained by the Internet experiment and the laboratory experiment.

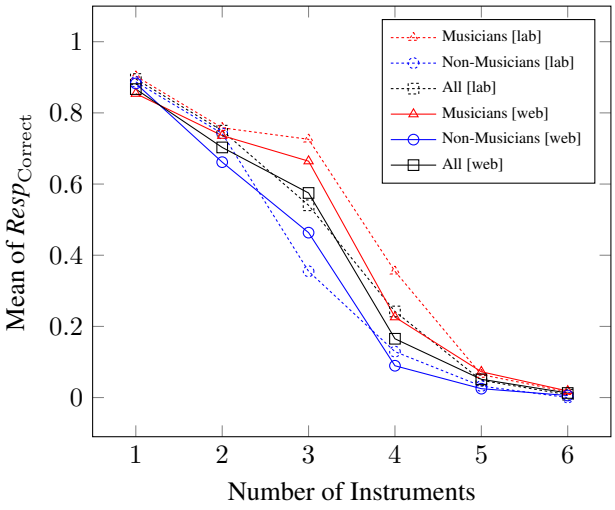


Figure 2. Probability of $Resp_{\mathrm{Correct}}$ grouped by Internet experiment (web) and laboratory experiment (lab). Solid lines represents the results of the Internet experiment and dashed lines represents the results of the laboratory experiment.

than non-musicians. According to the average marginal effect their chance of estimating the number of instruments correctly is 6.61% more likely for all stimuli. A similar increase for estimating the number correctly (4.15%) can be seen for participants being a professional in audio (*Professional* $= true$).

The average marginal effects of Num_{Inst} indicates up to which point humans are able to correctly estimate the number of instruments being played back. The average marginal effect of $Num_{\mathrm{Inst}} = 2$ shows that it is 12.93% less likely to estimate correctly when listening to two instruments instead of one instrument. Furthermore, in case of three instruments being played back the probability of estimating the wrong number increases to 20.12%. The highly negative average marginal effect of -0.4481 for $Num_{\mathrm{Inst}} = 4$ indicates that it is becoming very unlikely for humans to estimate the number of instruments correctly compared to estimating the number of one to three instruments.

For a detailed analysis of the differences between the Internet experiment and the laboratory experiment in a controlled environment, a second logit regression model was calculated. This logit regression model includes the data of the previous experiment which has responses of 62 participants. Besides Num_{Inst} an additional predictor variable *Environment* was added which can have the values 'web' or 'lab' (described in Table 5). The second logit regression model reveals that there are no significant differences ($p = 0.174$) between the two experiments. The low average marginal effect of -0.0184 also confirms that the type of the conducted experiments is applicable for an Internet environment.

Figure 2 depicts the mean probability for correctly estimating the number of instruments grouped by the environment. Since in [14] the differences between musicians and non-musicians were emphasized, their data is also depicted in Figure 2. As the logit regression model indicated, the difference in the performance of the participants between the Internet experiment and the laboratory experiment are very low. The participants in the laboratory experiment were about 4.6% better in average for all stimuli than the participants of the Internet experiment. When looking into the differences between musicians and non-musicians, the outcome for the Internet experiment and laboratory experiment differ slightly. In the laboratory experiment musicians performed about 31.6% better than non-musicians and in the Internet experiment musicians performed about 20.85% better.

The probability of correctly estimating the number of instruments does not consider how close an estimation is to the actual number of instruments. This means that a participant who estimates wrong by one instrument for all items has the same $Resp_{\mathrm{Correct}}$ like a participant who is always wrong by two instruments. Despite this work fo-

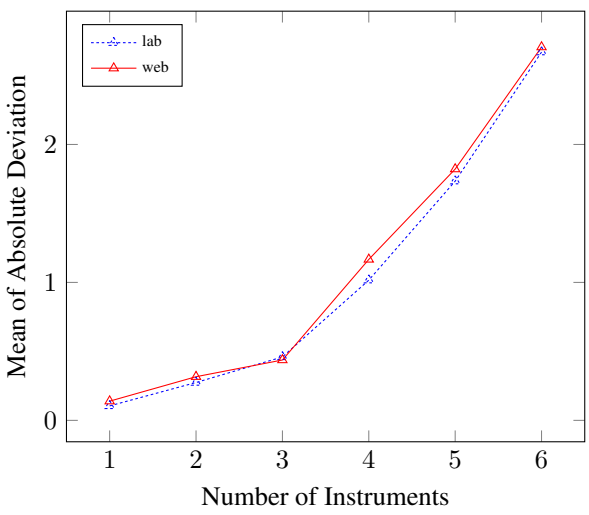


Figure 3. The mean of the absolute deviation grouped by Internet Experiment (web) and laboratory experiment (lab).

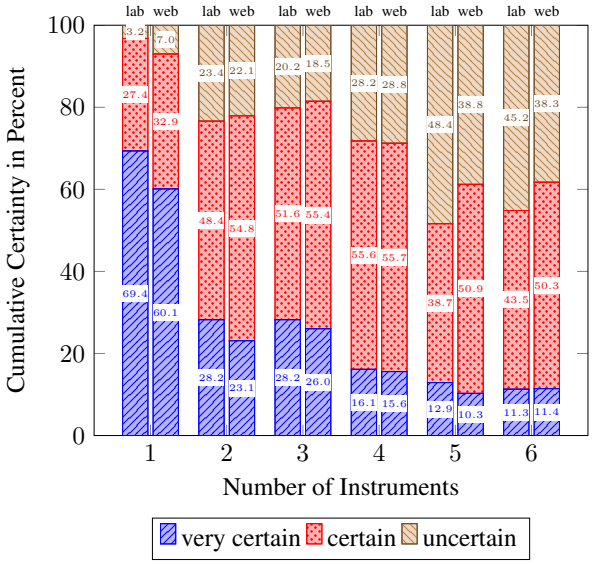


Figure 4. Differences in certainty between the Internet experiment (web) and laboratory experiment (lab).

cuses on the correct estimation, the differences in the absolute deviation to the correct number of instruments was also analyzed. The absolute deviation is defined as

$$Dev_{\mathrm{Abs}} = |Num_{\mathrm{Inst}} - Resp|. \quad (2)$$

Figure 3 depicts the mean of Dev_{Abs} for the laboratory experiment and the Internet experiment. A knee point can be seen for $Num_{\mathrm{Inst}} = 3$ where the slope of Dev_{Abs} changes.

To confirm the marginal differences between the Internet experiment and laboratory experiment for Dev_{Abs}, a linear regression model was calculated (see Table 6). Compared to the predictor variable Num_{Inst}, the coefficient of *Environment* is very low.

Coefficient	Estimate	Std. Error	z-value	p-value
(Intercept)	0.08535	0.02706	3.154	0.00161
$Num_{\mathrm{Inst}} = 2$	0.17683	0.01881	9.401	$<$ 2e-16
$Num_{\mathrm{Inst}} = 3$	0.30122	0.01881	16.014	$<$ 2e-16
$Num_{\mathrm{Inst}} = 4$	1.02154	0.01881	54.309	$<$ 2e-16
$Num_{\mathrm{Inst}} = 5$	1.67967	0.01881	89.297	$<$ 2e-16
$Num_{\mathrm{Inst}} = 6$	2.56667	0.01881	136.453	$<$ 2e-16
Environment = '*web*'	0.05481	0.02482	2.208	0.02724
Residual standard error: 0.6597 on 14753 degrees of freedom				
Multiple R-squared: 0.6603, Adjusted R-squared: 0.6602				
F-statistic: 4779 on 6 and 14753 DF, p-value: $<$ 2.2e-16				

Table 6. Linear regression model for Dev_{Abs} calculated from the data obtained by the Internet experiment and the laboratory experiment.

Another response variable that was obtained from the participants was the certainty of their estimation. Figure 4 depicts the certainty values for the Internet experiment and the laboratory experiment.

For testing whether the environment has a significant influence on the certainty of the participants (*Certainty*), a cumulative link model (also called ordered regression model) was calculated [4]. The cumulative link model is

Coefficient	Estimate	Std. Error	z-value	p-value
$Num_{\mathrm{Inst}} = 2$	-1.61273	0.05744	-28.078	$<$ 2e-16
$Num_{\mathrm{Inst}} = 3$	-1.43164	0.05701	-25.114	$<$ 2e-16
$Num_{\mathrm{Inst}} = 4$	-2.02315	0.05804	-34.858	$<$ 2e-16
$Num_{\mathrm{Inst}} = 5$	-2.47933	0.05901	-42.013	$<$ 2e-16
$Num_{\mathrm{Inst}} = 6$	-2.43573	0.05900	-41.283	$<$ 2e-16
Environment = '*web*'	-0.01008	0.07355	-0.137	0.891
Threshold coefficient	Estimate	Std. Error	z-value	
$uncertain\|certain$	-2.90465	0.08434	-34.441	
$certain\|verycertain$	-0.41072	0.08090	-5.077	

Table 7. Logit cumulative link model of certainty that was calculated from the data obtained by the Internet experiment and the laboratory experiment.

used since *Certainty* is an ordered dependent variable with the possible values 'uncertain', 'certain' and 'very certain'. The predictor variables for the ordered regression model are Num_{Inst} and *Environment*. In Table 7 is the cumulative link model for *Certainty* described. Same as $Resp_{\mathrm{Correct}}$, the environment of the experiment has no significant influence on the dependent variable *Certainty*. Considering the number of participants and the comparable low coefficient, the environment had a very low influence on *Certainty*.

5. DISCUSSION

Regarding the ability of estimating the number of instruments, the web experiment confirmed the results of the laboratory experiment [14]. Both experiments share the same outcome: The probability to correctly estimate up to about three instruments is higher than 50%.

In our previous result analysis of the laboratory experiment [14] we set the focus on the differences between musicians and non-musicians. Between the Internet experiment and the laboratory experiment, slightly different results were obtained when looking into how musicians and non-musicians performed (Figure 2). In the laboratory experiment musicians performed much better compared to

non-musicians than in the Internet Experiment. One reason seems to be that in the laboratory experiment 74.2% of the musicians had also a professional background in audio. In the Internet experiment only 27.5% of the musicians had a professional background in audio. Since audio professionals more often have to detect hardly audible differences in audio files they are more trained in this field. As mentioned before, being a musician in our context does just mean that the participant plays an instrument without any information about his expert level or the time he or she spends on practicing.

When examining the responses for items with the same number of instruments being played back, noticeable differences between the genres were found. For the stimuli with $Num_{\mathrm{Inst}} \leq 4$ the mean of $Resp_{\mathrm{Correct}}$ was 53.0% and for non-classical items 62.5%. Since the experiment was not designed for testing classical versus non-classical items, we cannot make definitive statements about whether humans are better in estimating instruments for a specific music genre. Moreover, we did not address this issue in our result analysis.

One of the main reasons for conducting the experiment was to find out which influence the Internet environment has on the results. All three regression models which included data of both experiments (Table 5, 6 and 7) revealed that the Internet environment had only very minor effects on the results. Moreover, despite the large number of participants in both experiments, the predictor variable *Environment* was even not significant in two out of three regression models.

It is often recommended to use headphones instead of loudspeakers for Internet experiments. From the relative low average marginal effect of *Setup* (Table 4), it can be derived that the type of the setup had only minor effects on the results of the Internet experiment.

Surprising was the small number of participants who had to be screened. We excluded 27 participants since they gave at least one non-serious response which is about 2.3% (1195 participants remained after excluding all trials which were not the first ones). Most of these excluded participants responded with either very high numbers (e. g. 99) or responded with zeros for their estimated number of instruments being played. We assume that especially participants who responded with zero only wanted to get an impression of the experiment.

6. CONCLUSION

The Internet experiment presented in this paper confirmed the results of a previous laboratory experiment that humans are able to correctly estimate up to around three instruments in music. Furthermore significant differences in performance between musicians and non-musicians found out by the previous experiment were confirmed. The comparison between the results of the Internet experiment and laboratory experiment revealed that only minor differences between both environments exist. Using headphones instead of loudspeaker is often held to be important when conducting listening tests over the Internet. In this experiment the audio setup used had only a minor influence on the results. According to these results, experiments in the fields of music perception or music information retrieval related to our procedure are well suited for being conducted over the Internet.

7. REFERENCES

[1] Albert S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. Bradford Books, MIT Press, Cambridge, 1990.

[2] Albert. S. Bregman and Jeffrey Campbell. Primary auditory stream segregation and perception of order in rapid sequences of tones. *Journal of experimental psychology*, 89(2):244–9, August 1971.

[3] Emilios Cambouropoulos. Voice and stream: Perceptual and computational modeling of voice separation. *Music Perception*, 26(1):75–94, 2008.

[4] Rune Haubo B. Christensen. Analysis of ordinal data with cumulative link models – estimation with the R-package ordinal, 2012.

[5] EBU. Loudness normalisation and permitted maximum level of audio signals (EBU Recommendation R 128), 2011.

[6] Peter Eckersley. How Unique Is Your Web Browser? In *Privacy Enhancing Technologies Symposium (PETS 2010)*, pages 1–18, Berlin, Germany, 2010.

[7] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical, and jazz music databases. In *Proc. ISMIR*, pages 287–288, 2002.

[8] David Huron. Voice Denumerability in Polyphonic Music of Homogeneous Timbres. *Music Perception: An Interdisciplinary Journal*, 6(4):361–382, 1989.

[9] Jin Ha Lee. Crowdsourcing music similarity judgments using mechanical turk. In *ISMIR*, pages 183–188, 2010.

[10] Ulf-Dietrich Reips. Standards for Internet-Based Experimenting. *Experimental Psychology*, 49(4):243–256, 2002.

[11] Ulf-Dietrich Reips. Using the Internet to Collect Data. In Harris Cooper, Paul M. Camic, Debra L. Long, A. T. Panter, David Rindskopf, and Kenneth J. Sher, editors, *APA Handbook of Research Methods in Psychology, Vol 2: Research designs: Quantitative, qualitative, neuropsychological, and biological*, volume 2, chapter 17, pages 291–310. American Psychological Association (APA), Washington, US, 2012.

[12] Matthew J. Salganik, Peter Sheridan Dodds, and Duncan J. Watts. Experimental study of inequality and unpredictability in an artificial cultural market. *Science (New York, N.Y.)*, 311(5762):854–6, February 2006.

[13] Catherine J. Stevens. Music perception and cognition: A review of recent cross-cultural research. *Topics in cognitive science*, 4(4):653–667, 2012.

[14] Fabian-Robert Stöter, Michael Schoeffler, Bernd Edler, and Jürgen Herre. Human ability of counting the number of instruments in polyphonic music. volume 19, page 035034. ASA, 2013.

[15] D. Wang and G. Brown. *Computational auditory scene analysis: Principles, algorithms, and applications*. Wiley-IEEE Press, 2006.

[16] Norma Welch and John H. Krantz. The World-Wide Web as a medium for psychoacoustical demonstrations and experiments: Experience and results. *Behavior Research Methods, Instruments, & Computers*, 28(2):192–196, June 1996.

SOCIAL-EQ: CROWDSOURCING AN EQUALIZATION DESCRIPTOR MAP

Mark Cartwright
Northwestern University
EECS Department
mcartwright@u.northwestern.edu

Bryan Pardo
Northwestern University
EECS Department
pardo@northwestern.edu

ABSTRACT

We seek to simplify audio production interfaces (such as those for equalization) by letting users communicate their audio production objectives with descriptive language (e.g. "Make the violin sound 'warmer.'"). To achieve this goal, a system must be able to tell whether the stated goal is appropriate for the selected tool (e.g. making the violin "warmer" with a panning tool does not make sense). If the goal is appropriate for the tool, it must know what actions need to be taken. Further, the tool should not impose a vocabulary on users, but rather understand the vocabulary users prefer. In this work, we describe SocialEQ, a web-based project for learning a vocabulary of actionable audio equalization descriptors. Since deployment, SocialEQ has learned 324 distinct words in 731 learning sessions. Data on these terms is made available for download. We examine terms users have provided, exploring which ones map well to equalization, which ones have broadly-agreed upon meaning, which term have meanings specific small groups, and which terms are synonymous.

1. INTRODUCTION

Much of the work of audio production involves finding the mapping between the terms in which a musician describes an acoustic concept (e.g. "Make the violin sound 'warmer.'") and the tools available to manipulate sound (e.g. the controls of a parametric equalizer). Often, mappings are non-obvious and require significant work to find.

We seek to simplify audio production interfaces (such as those for equalization) by letting users communicate their audio production objectives with descriptive language ("Make the violin sound 'warmer.'"). To achieve this goal, the system must be able to tell whether the stated goal is achievable for the selected tool (e.g. making the violin "warmer" with a panning tool does not make sense). It must also know what actions need to be taken, given the correct tool ("Use the parametric equalizer to boost the 2-4 kHz and the 200-500 Hz bands by 4 dB"). Further, the tool should be aware of possible variations in the mapping between word and audio among users (Bob's "warm" $\neq$ Sarah's "warm"), and the tool should be aware of which words are synonymous.

In this work, we describe SocialEQ, a project to crowdsource a vocabulary of audio descriptors, grounded in perceptual data that can be mapped onto concrete actions by an equalizer (EQ) to effect meaningful change to audio files. SocialEQ has, to date, been taught 324 distinct words in 731 learning sessions. Through our analysis of the data collected, we address the following questions:

1. What audio descriptors are actionable by an audio equalizer?
2. How widely-agreed-upon is the meaning of an EQ descriptor?
3. What EQ descriptors are true audio synonyms?

2. BACKGROUND AND RELATED WORK

There is currently no universal dictionary of audio terminology that is defined both in terms of subjective experiential qualities and measurable properties of a sound. Tools built from text co-occurrence, lexical similarity and dictionary definitions (e.g. WordNet [13]) are fundamentally different in their underpinnings and do not address the issue of how words map to measurable sound features.

There are some terms relating to pitch (high, low, up, down) and loudness (soft, loud) that have relatively well-understood [6, 18] mappings onto measurable sound characteristics. Some terms of art used by recording engineers [8] describe effects produced by recording and production equipment, and are relatively easy to map onto measurable properties. These include "compressed" (i.e., a compressor has been applied to reduce the dynamic range of the audio) and "clipped" (i.e., the audio is distorted in a way characteristic of an overloaded digital recorder). These terms are not, however, widely understood by either musicians or the general public [21].

Numerous studies have been performed over the last fifty years in the hopes of finding universal sound descriptors that map onto a set of canonical perceptual dimensions [4, 11, 20, 22]. Also, in the last decade or so, many researchers coming from backgrounds such as recording engineering [8], music composition [19] and computer science [16] have studied the space of terminology, seeking a

universal set of English terms for timbre. Researchers in acoustic signal processing and perception continue to seek English taxonomies for timbre descriptors. Some seek general taxonomies of sound descriptors [16]. Others are focused on timbre adjectives for particular instruments [2,9].

Such studies typically start by determining a set of natural descriptors by performing a survey. These descriptors are then used in a second study, where participants evaluate sounds in terms of the descriptors. These are then mapped onto machine-measurable parameters, such as spectral tilt, sound pressure level, or harmonicity. Commonalities in descriptor mappings are found between participants in the studies and then some small set of descriptive terms are proposed as relatively universal.

In audio engineering, there has been work directly mapping equalizer parameters to commonly used descriptive words using a fixed mapping [12, 14]. A problem with these approaches is that mappings are very time-consuming to develop, the former requires a painstaking training process for each user, and the latter is brittle with respect to the variability that exists across users.

In contrast, [15] establishes the effectiveness of an approach to automatically learning mappings of audio adjectives onto actionable controllers in a small study using 4 researcher-selected descriptors and 19 participants. This study was, however, quite small in the number of participants and the number of adjectives learned.

Despite the efforts of all these groups, a universal map of descriptive terms for audio remains an unachieved goal. We believe this is because the terms used range from widely-agreed-upon (e.g. *loud*) to ones that have agreement within groups but not between groups (e.g. *warm*) to idiosyncratic terms meaningful solely to individuals (e.g. *splanky*).

In this work, rather than seek a set of universal underlying dimensions of timbre or the universal meaning of a descriptive terms in all auditory contexts, we seek an understanding of descriptors in a specific context: audio equalization.

Figure 1. SocialEQ ratings screen. The user rates how well each audio example corresponds to the descriptive term they selected at the prior stage. Users rate an example by dragging the associated circle to the desired horizontal position. Vertical position does not matter.

Figure 2. The equalization curve and control slider learned for "bright" in a single session.

3. THE SOCIALEQ TASK

`SocialEQ.org` is a web-based application that learns an audio equalization curve associated with a user-provided audio descriptor. To deploy the software to a large audience for our data collection, we have implemented it as a web application using Adobe Flex and Drexel University's Audio Processing Library for Flash (ALF) [17].

After agreeing to participate, we ask participants to "enter a descriptive term in the language in which you are most comfortable describing sound (e.g. 'warm' for English, 'claro' in Spanish, or 'grave' in Italian), pick a sound file which we will modify to achieve the descriptive term, then click on 'Begin'."

Participants are then given the choice of three different source audio files to manipulate: "Electric Guitar", "Piano", and "Drums". All files were sampled at 44.1 kHz and 16 bits. The files all had constant sound (i.e. no breaks in the audio) and were presented in loops that were 8, 10, and 14 seconds long, respectively.

Once a participant selects a descriptive term and a sound file, they are asked to "rate how 'your word' that sound is" using the interface shown in 1. The participant then rates 40 examples of the chosen audio file, each modified with an equalization curve. Fifteen curves are repeats of prior examples, to test for rating consistency. Figure 1 shows the interface for rating examples.

From these rated examples, the system learns the relative boost/cut to apply each of 40 frequency bands. These bands have equivalent rectangular bandwidth (ERB) [3] derived center frequencies. We use the method from [15] to learn the EQ curve from user responses. This method treats each frequency band as independent and correlates changes in user ratings with changes in gain for that band.

Positive correlation indicates a boost, and negative correlation a cut. The result is a 40-band EQ curve for the descriptor learned from the participant. The system uses the learned equalization curve to build a slider that lets the participant manipulate the sound in terms of the descriptor (Figure 2).

After teaching SocialEQ an audio descriptor and trying the control slider, participants were asked to complete a question survey that assessed their background, listening environment, and experience using SocialEQ.

4. DATA COLLECTION

For a data collection of this size, an on-site data collection was not feasible. We instead recruited participants through Amazon's Mechanical Turk. We had 633 participants who participated in a total of 1102 training sessions (one session per learned word). We paid participants $1.00 (USD) per session, with the possibility of up to a $0.50 bonus, determined by the consistency of their equalized audio example ratings. While the quality of loudspeakers could not be controlled, over 92% of the participants reported listening over either headphones or large speakers (rather than small/laptop speakers).

4.1 Inclusion Criteria for Sessions

Before analyzing the results, we first removed sessions by participants who seemed to not put effort into the task. The mean time to teach the system the definition of a single descriptor was 292 seconds (SD=237). We removed all sessions where the participant completed the task in less than 60 seconds. We also removed all sessions where the participant gave the default rating for more than 5 out of the 40 examples. We also removed any session where the participant responded "no" to the survey question: "Was the listening environment quiet?".

Recall that 15 of the 40 examples were repeats in any session. This let us test for consistency of user responses to audio examples when teaching SocialEQ. We measured consistency using Pearson correlation between the ratings of the test and repeated examples. The median consistency across sessions was 0.41 (95% CI [0.39, 0.44]).

Only sessions with consistency above 0 were retained. This left 481 participants who taught the machine in 731 sessions (Individual participants were allowed to teach more than one descriptor to the system. The maximum number of descriptors a participant taught was 9.).

5. RESULTS

Since we have data on hundreds of words taught by hundreds of participants, we are not able to describe it all in detail here. We have made the data available for use by the research community at `http://socialeq.org/data`.

5.1 Definitions

descriptor: An adjective (e.g. *warm*) taught to the system in at least one session.

session: A participant teaches SocialEQ a descriptor, tries the learned controller, and completes the survey.

relative spectral curve (RSC): A set of relative gains (i.e. boosts or cuts) on the 40 ERB frequency bands. To make RSC comparable, every RSC is normalized by putting gain values in standard deviations from the mean value of the 40 bands.

user-concept: The RSC learned in a single session (e.g. the session where Bob teaches 'warm' to SocialEQ).

descriptor definition The set of user-concepts that all share a descriptor. A definition may be vague or precise depending on how much agreement there is between user-concepts that share the descriptor. Figure 3 shows *deep* and *sharp*.

5.2 The descriptors

In the 731 sessions, there were 324 unique descriptors. The descriptors taught most frequently are listed in Table 1. Of the 324 words, 91 occurred two or more times. The most popular descriptor was *warm*, but note that there was a bias for participants to teach the system *warm* due to its use as the example in the instructions (see Section 3).

Table 1. The 10 most common descriptors contributed

Rank	Descriptor	Sessions
1	warm	57
2	cold	25
3	soft	24
4	loud	22
5	happy	19
6	bright	16
7	harsh	15
8	soothing	14
9	heavy	11
10	cool	11

5.3 Representing equalization concepts

In this paper, we make the assumption that participants judged each equalization example relative to the unprocessed source rather than judging the absolute spectrum of each equalization example. We therefore have chosen to represent equalization concepts in terms of relative changes in each frequency band rather than the resulting spectrum after equalization. This allows us to compare equalization concepts from varying source material played on varying loudspeakers. In Figure 3, we show the distribution of RSCs collected for two example descriptors: *deep* and *sharp*. Each column shows the distribution of learned values for the corresponding ERB-spaced frequency band.

Note that except for one outlier participant, there was fairly high agreement for the meaning of *sharp*. This is interesting, since most musicians are taught that *sharp* relates to relative pitch, rather than the spectral characteristics of a sound. Our data indicate *sharp* also has other connotations that relate to timbre.

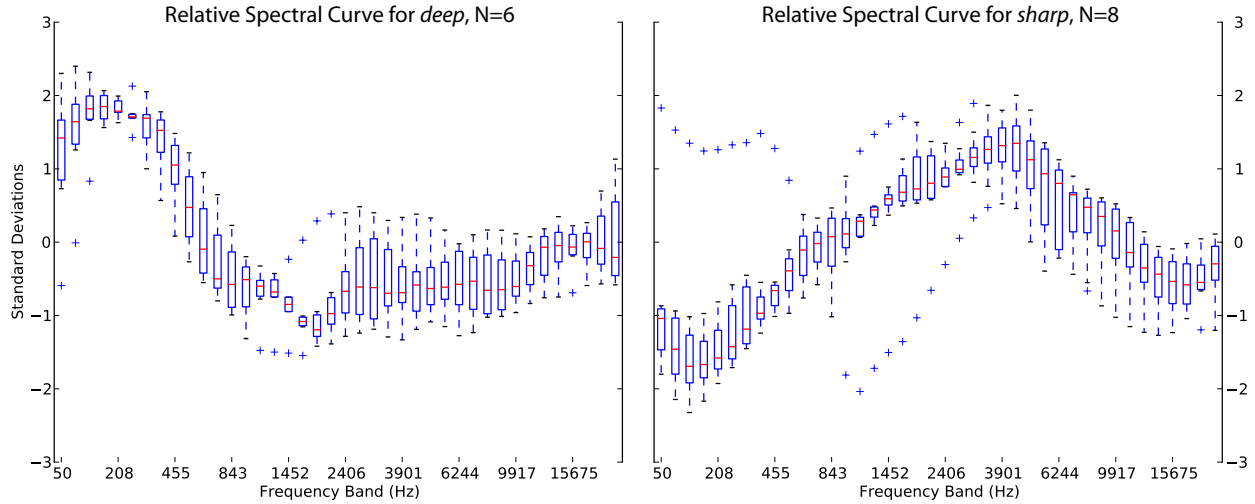


Figure 3. Per frequency band boxplots of RSCs for the descriptors *deep* and *sharp*, learned in 6 and 8 sessions respectively. In each ERB-spaced frequency band, the center line is the median, the box represents 50% of the data, the whiskers represent the remaining 50%. The pluses represent outliers.

5.4 Actionable equalization descriptors

Table 2. Top 10 descriptors taught to the system by at least 4 people, ranked by *mean slider rating*, which ranges from -3 (Strongly Disagree) to 3 (Strongly Agree)

Rank	Word	Mean Response
1	relaxing	2.75
2	quiet	2.60
3	hot	2.50
4	hard	2.50
5	heavy	2.36
6	smooth	2.33
7	deep	2.33
8	bright	2.31
9	soothing	2.31
10	mellow	2.29

As stated in Question 1 in Section 1, one one goal of this paper is to determine what audio descriptors describe goals achievable by an audio equalizer (i.e. the audio descriptor is an equalization descriptor). One way to answer this is to simply look at the *mean slider rating*: the mean response to the survey statement, "The final (control) slider captured my target audio concept." Participants were asked to respond on a 7-level Likert scale coded from -3 (Strongly Disagree) to 3 (Strongly Agree). Table 2 shows the 10 descriptors with the highest mean response to this question that were contributed by at least 4 participants.

The learning approach used by SocialEQ [15] has an inherent bias toward learn smooth equalization curves and has difficulty learning curves with narrow boosts or cuts or frequency relationships that are non-linear or dependent. Therefore, *mean slider rating* is a sufficient but not necessary condition to determine whether the descriptor is actionable by an equalizer.

5.5 Agreed upon equalization descriptors

Table 3. Top 10 descriptors ranked by the *agreement score* described in Section 5.5

Rank	Word	Agreement Score
1	tinny	0.294
2	pleasing	0.222
3	low	0.219
4	dry	0.210
5	metallic	0.195
6	quiet	0.188
7	deep	0.164
8	hollow	0.160
9	light	0.131
10	warm	0.130

The second question we would like to answer is "How widely-agreed-upon is the meaning of an EQ descriptor?". For some words, the meaning of a descriptor, as embodied in the RSC learned in a particular session, may vary significantly from person to person. We want to find which descriptors vary the least from person to person, or rather which descriptors have the most widely-agreed-upon meanings. To answer the question we looked at the total variance (i.e. the trace of the covariance matrix) of RSCs within a descriptor definition. This can be written simply as the sum of the variance in each RSC frequency-band for the user-concepts in a descriptor definition:

$$\text{trace}(\Sigma)_{descriptor} = \frac{1}{N}\sum_{k=0}^{39}\sum_{n=0}^{N-1}(x_{n,k} - \mu_k)^2 \quad (1)$$

where N is the number of user-concepts in the descriptor definition, k is the index of the frequency band, x is the RSC for user-concept n, and μ_k is the mean of frequency

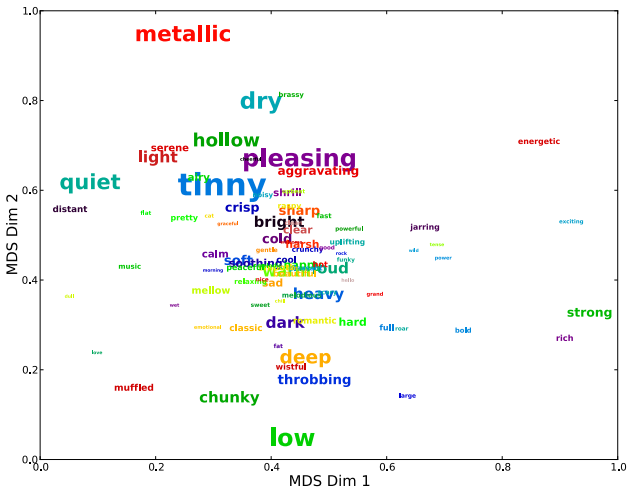


Figure 4. 2-dimensional multi-dimensional scaling (MDS) plot of the descriptors. Their font size positively correlates to their agreement score from Equation 2.

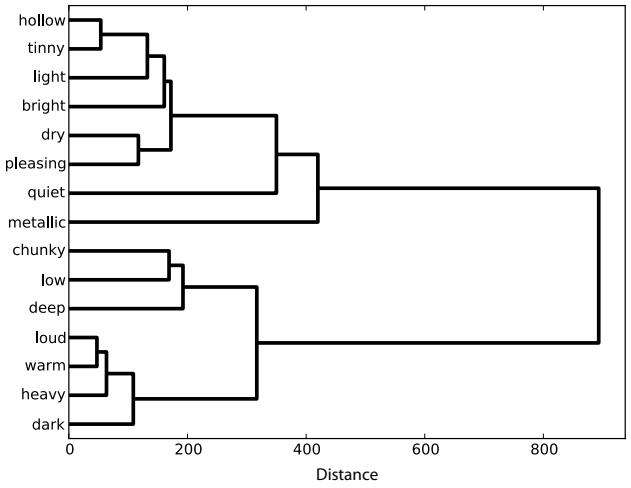


Figure 5. Hierarchical clustering of the 15 descriptors with the highest agreement scores.

band k over the N user-concepts in the descriptor definition.

If we then divide the natural logarithm of the number of user-concepts ($\log(N)$) by this value, i.e.:

$$agreementscore = \frac{log(N)}{\text{trace}(\Sigma)_{descriptor}} \quad (2)$$

we have an *agreement score* that takes into account both total variance and the popularity of the descriptor. We used $\log(N)$ to linearize the number of user-concepts since the frequency with which a descriptor was taught the system was distributed similarly to Zipf's law [10]. When we rank the descriptors by this score, we discover which descriptors have more agreement amongst the participants. The top ten descriptors ranked by this score are shown in Table 3.

5.6 Audio descriptor synonyms

To answer the last question, "What EQ descriptors are true audio synonyms", we compared the descriptor definitions using a distance function.

Table 4. Synonyms of high agreement descriptors (see Table 3) found through comparing descriptor definitions

descriptor	synonyms
light	tinny, crisp
tinny	hollow, crisp, light, shrill, bright, cold, raspy
deep	throbbing, dark
hollow	tinny, dry, shrill, pleasing

To compare learned descriptor definitions, we wanted a distance measure that would: 1) allow for varying number of user-concepts per descriptor definition; 2) allow for multi-modal distributions; and 3) take into account the uncertainty of the descriptor definition. Therefore, we modelled each descriptor definition as a probability distribution over the user-concepts for the descriptor, and we then compared descriptor definitions using an approximation of the symmetric KL-divergence.

The steps to calculate the distance between two descriptor definitions are:

1. Model each user-concept as a Gaussian distribution, $\mathcal{N}(\mu_i, \Sigma_i)$, where μ_i is the RSC of the user-concept and Σ is a diagonal covariance matrix in which the variance for each frequency-band is set by $\sigma^2_{i,k} = (\sigma_k - \sigma_k r_i)^2$ where σ_k is the sample standard deviation of frequency-band k for RSCs of *all* descriptors, and r_i is the ratings consistency for the session that learned user-concept i. Here we are using the ratings consistency as a measure of the uncertainty of the user-concept, mapping a consistency range of $[0, 1]$ to a per-frequency-band variance range of $[\sigma_k, 0]$.

2. Then model each descriptor definition as follows:

$$P(x) = \frac{1}{N} \sum_{i=0}^{N-1} \mathcal{N}(\mu_i, \Sigma_i) \quad (3)$$

   where $\mathcal{N}(\mu_i, \Sigma_i)$ is the distribution for the i^{th} user-concept.

3. We then use a symmetric Monte Carlo approximate KL-divergence [7] to compare two descriptor definition models.

Using this distance measure, we computed the distance between every pair of descriptor definitions. With these distances, we can map and visualize the relationships of descriptor definitions. In Figure 4, we placed the descriptor definitions in a two-dimensional space by using metric multi-dimensional scaling [1], each descriptor definition is scaled by its agreement score so that well-defined descriptors are larger. To get a better sense of the relationships of the 15 high agreement descriptor definitions listed in Table 3, we performed agglomerative hierarchical clustering using the "group average" algorithm [5] and plotted the dendrogram in Figure 5. From this you can see two clusters

formed with descriptors one might typically associates as opposites: *bright/dark*, *quiet/loud*, *light/heavy*. From this we can also see relationships of descriptors which may not have been obvious such as *pleasing* being closely associated with *dry*.

In Table 4, we list a few high agreement descriptors along with their synonyms through comparing descriptor definitions. We considered two words synonyms if their distance was within the first percentile of all the pairwise distances. Also, only descriptors definitions that consisted of at least two user-concepts and had at least a 1.0 *mean slider rating* (as described in Section 5.4) were included.

6. CONCLUSION

In this work, we analyzed the equalization descriptors that were taught to a web-based equalization learning system, SocialEQ. We developed methods to determine which descriptors map well to equalization, which have broadly-agreed upon meaning, and which are synonymous. During analysis we found both expected and unexpected relationships and definitions of descriptors, but more importantly we developed the ground work of an intelligent audio production system that responds to the descriptive language of the user. With a large collection of equalization descriptors and the techniques described in this paper, we have a map of equalization descriptors which a system could use to determine whether a goal is achievable with an equalizer and whether it needs to learn an individual's equalization concept or can use an agreed upon concept.

This work was supported by grant by National Science Foundation Grant No. IIS-1116384.

7. REFERENCES

[1] I. Borg and P. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Verlag, 2005.

[2] A. Disley and D. Howard. Spectral correlates of timbral semantics relating to the pipe organ. *Speech, Music and Hearing*, 46:25–39, 2004.

[3] B. Glasberg and B. Moore. Derivation of auditory filter shapes from notched-noise data. *Hearing Research*, 47(12):103–138, 1990.

[4] J. Grey. Multidimensional perceptual scaling of musical timbres. *The Journal of the ASA*, 61(5):1270–1277, 1977.

[5] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer New York, 2001.

[6] H. Helmholtz and A. Ellis. *On the sensations of tone as a physiological basis for the theory of music*. Dover, New York, 2d english edition, 1954.

[7] J. Hershey and P. Olsen. Approximating the kullback leibler divergence between gaussian mixture models. In *Proc. ICASSP, 2007*.

[8] D. Huber and R. Runstein. *Modern recording techniques*. Focal Press/Elsevier, Amsterdam ; Boston, 7th edition, 2010.

[9] E. Lukasik. Towards timbre-driven semantic retrieval of violins. In *Proc. of International Conference on Intelligent Systems Design and Applications, 2005*.

[10] C. Manning, P. Raghavan, and H. Schtze. *Introduction to information retrieval*. Cambridge University Press Cambridge, 2008.

[11] S. McAdams, S. Winsberg, S. Donnadieu, G. Soete, and J. Krimphoff. Perceptual scaling of synthesized musical timbres: Common dimensions, specificities, and latent subject classes. *Psychological Research*, 58(3):177–192, 1995.

[12] S. Mecklenburg and J. Loviscach. subjeqt: controlling an equalizer through subjective terms. In *Proc. of CHI '06 Extended Abstracts on Human Factors in Computing Systems, 2006*.

[13] G. Miller. Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39–41, 1995.

[14] D. Reed. Capturing perceptual expertise: a sound equalization expert system. *Knowledge-Based Systems*, 14(12):111–118, 2001.

[15] A.T. Sabin, Z. Rafii, and B. Pardo. Weighting-function-based rapid mapping of descriptors to audio processing parameters. *Journal of the AES*, 59(6):419–430, 2011.

[16] M. Sarkar, B. Vercoe, and Y. Yang. Words that describe timbre: a study of auditory perception through language. In *Proc. of Language and Music as Cognitive Systems Conference, 2007*.

[17] J. Scott, R. Migneco, B. Morton, C. Hahn, P. Diefenbach, and Y. Kim. An audio processing library for mir application development in flash. In *Proc. ISMIR, 2010*.

[18] R. Shepard. Geometrical approximations to the structure of musical pitch. *Psychological Review*, 89(4):305–333, 1982.

[19] D. Smalley. Spectromorphology: explaining sound-shapes. *Organised Sound*, 2(02):107–126, 1997.

[20] L. Solomon. Search for physical correlates to psychological dimensions of sounds. *The Journal of the ASA*, 31(4):492–497, 1959.

[21] E. Toulson. A need for universal definitions of audio terminologies and improved knowledge transfer to the audio consumer. In *Proc. of The Art of Record Production Conference, 2003*.

[22] A. Zacharakis, K. Pastiadis, G. Papadelis, and J. Reiss. An investigation of musical timbre: Uncovering salient semantic descriptions and perceptual dimensions. In *Proc. of the ISMIR, 2011*.

TASTE OVER TIME: THE TEMPORAL DYNAMICS OF USER PREFERENCES

Joshua L. Moore, Shuo Chen, Thorsten Joachims
Cornell University, Dept. of Computer Science
{jlmo|shuochen|tj}@cs.cornell.edu

Douglas Turnbull
Ithaca College, Dept. of Computer Science
dturnbull@ithaca.edu

ABSTRACT

We develop temporal embedding models for exploring how listening preferences of a population develop over time. In particular, we propose time-dynamic probabilistic embedding models that incorporate users and songs in a joint Euclidian space in which they gradually change position over time. Using large-scale Scrobbler data from Last.fm spanning a period of 8 years, our models generate trajectories of how user tastes changed over time, how artists developed, and how songs move in the embedded space. This ability to visualize and quantify listening preferences of a large population of people over a multi-year time period provides exciting opportunities for data-driven exploration of musicological trends and patterns.

1. INTRODUCTION

Embedding methods are a class of models that learn positions for discrete objects in a metric space. Such models are widely employed in a variety of fields, including natural language processing and music information retrieval (MIR). In MIR, these methods find use in recommendation and playlist prediction, among other problems. Embedding methods offer advantages in two main aspects. First, they are often very easy to interpret: the resulting space can be easily visualized and inspected in order to explain the behavior of the model. Second, they can be applied to discrete objects without features, learning feature representations of the objects as part of model training.

In this work, we explore the use of embedding methods as a tool for identifying trends and patterns in multi-year listening data of Last.fm users. In particular, we propose novel time-dynamic embedding models that generate trajectories of musical preferences by jointly embedding listeners and the songs they play in a single metric space. In order to do this, we extend existing probabilistic playlists models [1, 2] by adding time dynamics, allowing users and songs to change position on a multi-year scale. By examining these models, we can draw conclusions about the behavior of listeners and musical artists over time. Based on these findings, we conjecture that these time-dynamic embedding methods provide exciting opportunities for data-driven exploration of musicological trends and patterns. To facilitate such research, scalable software implementations of our embedding methods are available at http://lme.joachims.org.

2. RELATED WORK

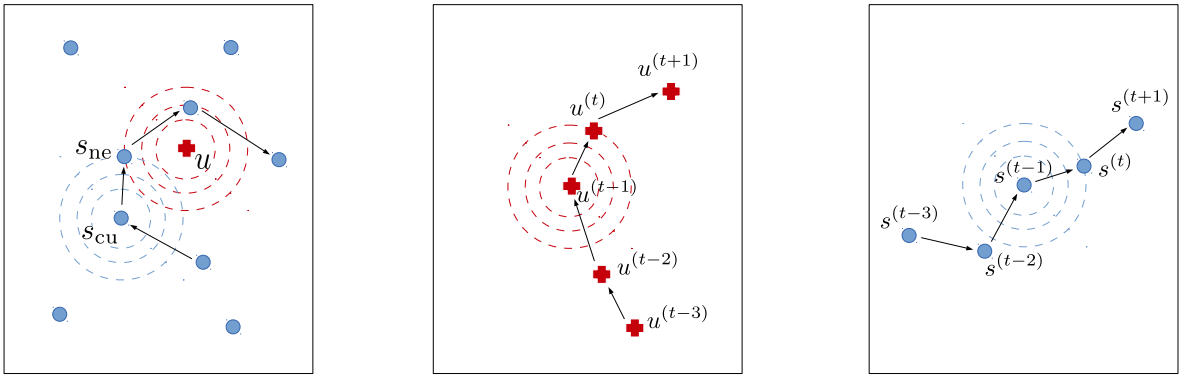
Embedding music into a low-dimensional space is useful for visualization and automatic playlist generation. There are numerous existing algorithms, such as Multi-Dimensional Scaling [7] and Local Linear Embedding [4], which have been applied to large corpora of songs and artists.

Our work is motivated by recent work of Moore et al. [2, 5] and Aizenberg et al. [1] on using historical playlist data, but we focus on long-term temporal dynamics. This is different from the short-term dynamics considered by Aizenberg et al., namely, the time of day of a track play by a station. This time dependency is employed to factor out the influence of different format blocks at a radio station based on the time of day (i.e. a college station may play classical music from 6 AM to 9 AM and jazz from 9 AM to noon). In our work, we allow the positions of users or songs to vary smoothly over the long-term, learning a representation for each three-month time step from the beginning of 2005 to the end of 2012. Second, both of these related works focus on automatic playlist prediction. In this paper, we instead use our model as a data analysis tool to explore long-range trends in the behavior of users, songs, and artists.

Weston et al. [9] use music embedding for a variety of MIR tasks including tag prediction and determining song similarity. Their embedding algorithm works by optimizing a rank-based loss function (e.g., AUC, precision at k) over training data for a given task. Our work differs from this in that our embedding results from a probabilistic sequence model that is learned from the track histories of users. In addition, the work by Weston et al. does not attempt to model the temporal dynamics.

Dror et al. [3] explore the use of temporal dynamics in collaborative filtering for music. However, the use of time dynamics in their work is mainly restricted to modeling biases for songs and users, which does not permit the visualization and analysis applications enabled by our work.

Finally, Shalit et al. [8] applies a dynamic topic model to audio features of songs for the purpose of modeling and



(a) (b) (c)

Figure 1: Illustrations of the embedding models. Blue dots and red crosses represent songs and users respectively. (a) Static playlist model. A playlist is represented by songs that are linked by arrows. The next song s_{ne} is decided by both current song s_{cu} and the user u (The popularity term also has its effect, which is not shown here). (b) The drifting of a user u over timesteps in the user-dynamic model. At each timestep, a random walk governed by a Gaussian distribution is taken. (c) Similar drifting of a song s over timesteps in the song-dynamic model.

discovering musical influence. While this work does not explicitly involve embedding songs, users or artists, it is a good example of the use of temporal dynamics in analysis of music data. In addition, their topic model requires audio features to represent each song. This is in contrast to our model where features are not required.

3. MODEL

In this section we detail the probabilistic embedding models we propose for temporal embedding of users and songs. Starting from a static playlist model (Section 3.1) similar to [2, 5], we incorporate a macroscopic temporal model under which the embedding can change over time, providing trajectories for users and songs through embedding space. In particular, we propose a user-dynamic embedding model (Section 3.2) in which users move against over a map of songs, as well as a song-dynamic embedding model (Section 3.3) in which songs move against a map of users. For both models, we briefly outline how they can be trained using maximum likelihood (Section 3.4).

3.1 Embedding Songs and Users for Playlist Modeling

Given a collection of songs $\mathcal{S} = \{s_1, \ldots, s_{|\mathcal{S}|}\}$ and a collection of users $\mathcal{U} = \{u_1, \ldots, u_{|\mathcal{U}|}\}$, each user's listening history can be described as a sequence of songs $p = (p^{[1]}, ..., p^{[k_p]})$ of length k_p, where each $p^{[i]} \in \mathcal{S}$. We refer to this (multi-year) sequence as the "playlist" of that user. The collection D of all user playlists is the training data for our embedding method.

Following the approach proposed in [2], we model a user's playlist using a first-order Markov model, but also augment it with a user model similar to [1]. As a result, the probability $\Pr(s_{\text{ne}}|s_{\text{cu}}, u)$ of the next song in a playlist depends only on the current song and the user. The overall probability of a playlist is therefore

$$\Pr(p|u) = \prod_{i=1}^{k_p} \Pr(p^{[i]}|p^{[i-1]}, u). \quad (1)$$

Note that transition triples $(s_{\text{ne}}|s_{\text{cu}}, u)$ (i.e., reads as "user u listened to the current song s_{cu}, then listened to the next song s_{ne}") are a sufficient statistic for this model.

Our goal is to embed each song and user into a d-dimensional latent Euclidean space such that song-song and song-user distances model the transition probabilities $\Pr(s_{\text{ne}}|s_{\text{cu}}, u)$. This provides such distances with a clear semantic meaning. More specifically, we want to learn a mapping $X(\cdot)$ that maps every song s or user u into that space, namely $X(s), X(u) \in \mathbb{R}^d$. The dimensionality d is manually specified. Alternatively, X can be considered as a $(|\mathcal{S}| + |\mathcal{U}|) \times d$ embedding matrix, the rows of which correspond to the position of songs and users in the latent space. We will not distinguish the two interpretations of X in the rest of the paper if it is clear from the context.

The specific model we propose for relating distances to transition probabilities is

$$\Pr(s_{\text{ne}}|s_{\text{cu}}, u) = \frac{e^{-\Delta(s_{\text{ne}}, s_{\text{cu}})^2 - \Delta(s_{\text{ne}}, u)^2 + b_{\text{idx}(s_{\text{ne}})}}}{Z(s_{\text{cu}}, u)}, \quad (2)$$

where $\Delta(x, y) = ||X(x) - X(y)||$ is the Euclidean distance between two embedded items (either song or user) in the latent space. $b_{\text{idx}(s)}$ is a scalar bias term that is added to model the popularity of each song, where $\text{idx}(s)$ returns the index for song s. For example, $\text{idx}(s_i) = i$. $Z(s_{\text{cu}}, u)$ is the partition function that normalizes the distribution. It is defined as

$$Z(s_{\text{cu}}, u) = \sum_{i=1}^{|\mathcal{S}|} e^{-\Delta(s_i, s_{\text{cu}})^2 - \Delta(s_i, u)^2 + b_i}. \quad (3)$$

Panel (a) of Figure 1 illustrates how song and user positions in the embedding space relate to the transition probability $\Pr(s_{\text{ne}}|s_{\text{cu}}, u)$. The red cross is the position of the

user, and the blue dot labeled s_{cu} is the current song in the playlist. The probability of playing some song s_{ne} next depends on its sum of the squared distances to the current song and the user, plus its inherent popularity $b_{s_{\text{ne}}}$. This means that the transition to the next song s_{ne} is more likely if (1) the next song is close to the current song in the latent space, (2) the next song is close to the user('s taste) in the latent space, or (3) the next song is popular. We focus our experiments on two-dimensional embeddings, since this provides us with an X that can easily be visualized. However, higher-dimensional embeddings are possible as well.

3.2 User-dynamic Embedding Model

Combining Equations (1) and (2) models a playlist as a stochastic process on a microscopic level (i.e., on the timescale of minutes). In addition, we also model changes in user preferences as a stochastic process on a macroscopic level. In the following experiments, each macroscopic timestep $t \in \mathcal{T}$ ($\mathcal{T}$ is the set of all timesteps) denotes a quarter of a year, and notation like 20083 denotes "third quarter of year 2008".

Let us first consider a macroscopic stochastic process where positions of users are changing over time, while the position of the songs are fixed in the latent space. Denoting with $u^{(t)}$ the position of user u in embedding space at timestep t, the overall trajectory of a user is $u^{(*)} = (u^{(1)}, u^{(2)}, ...)$. At each timestep t, the microscopic transition probability $\Pr(s_{\text{ne}}|s_{\text{cu}}, u^{(t)})$ now depends on the users current position, and the conditional probability of the next song is

$$\Pr(s_{\text{ne}}|s_{\text{cu}}, u^{(t)}) = \frac{e^{-\Delta(s_{\text{ne}},s_{\text{cu}})^2 - \Delta(s_{\text{ne}},u^{(t)})^2 + b^{(t)}_{\text{idx}(s_{\text{ne}})}}}{Z(s_{\text{cu}}, u^{(t)})}. \quad (4)$$

Note that even though the positions of songs are fixed, we still give each song a time-varying popularity term $b_i^{(t)}$.

To restrict users from drifting too much from one timestep to the other, we model a users trajectory as a Gaussian random walk. Panel (b) of Figure 1 illustrates such a random walk. Concretely, this means that the user's next position $u^{(t)}$ is a Gaussian step $\mathcal{N}(u_i^{(t-1)}, \frac{1}{2\nu_{\text{user}}} I_d)$ from the current position $u^{(t-1)}$. Here, I_d is the d-dimensional identity matrix, and ν_{user} is the variance (which can be viewed as a regularization coefficient that influences step sizes). This Gaussian distribution makes it more likely that the user's positions at two consecutive timesteps are close to each other.

Considering both the stochastic process over transition triples and the stochastic process describing the users' trajectories, the overall user-dynamic embedding model can be trained via maximum likelihood. The resulting optimization problem is

$$\max_{\substack{X \in \mathbb{R}^{(|\mathcal{S}|+|\mathcal{T}||\mathcal{U}|)\times d} \\ b \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{S}|}}} \prod_{(s_{\text{ne}}|s_{\text{cu}},u^{(t)}) \in D} \Pr(s_{\text{ne}}|s_{\text{cu}}, u^{(t)}) \cdot \prod_{i=1}^{|\mathcal{U}|} \prod_{\substack{t \in \{\tau | (\tau \in \mathcal{T}) \\ \wedge (\tau - 1 \in \mathcal{T})\}}} e^{-\nu_{\text{user}} \Delta(u_i^{(t-1)}, u_i^{(t)})^2}, \quad (5)$$

where the song and time-dependent user positions are optimized to maximize the likelihood of the observed playlists.

3.3 Song-dynamic Embedding Model

Similar to the user-dynamic embedding model, we also consider a song-dynamic embedding model which fixes the position of users and allows songs to drift over time. In this model, the probability of each transition triple is

$$\Pr(s_{\text{ne}}^{(t)}|s_{\text{cu}}^{(t)}, u) = \frac{e^{-\Delta(s_{\text{ne}}^{(t)},s_{\text{cu}}^{(t)})^2 - \Delta(s_{\text{ne}}^{(t)},u)^2 + b^{(t)}_{\text{idx}(s_{\text{ne}})}}}{Z(s_{\text{cu}}^{(t)}, u)}. \quad (6)$$

After introducing an analogous Gaussian random walk for songs over different timesteps (as illustrated in Panel (c) of Figure 1), we get the training problem

$$\max_{\substack{X \in \mathbb{R}^{(|\mathcal{T}||\mathcal{S}|+|\mathcal{U}|)\times d} \\ b \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{S}|}}} \prod_{(s_{\text{ne}}^{(t)}|s_{\text{cu}}^{(t)},u) \in D} \Pr(s_{\text{ne}}^{(t)}|s_{\text{cu}}^{(t)}, u) \cdot \prod_{i=1}^{|\mathcal{S}|} \prod_{\substack{t \in \{\tau | (\tau \in \mathcal{T}) \\ \wedge (\tau - 1 \in \mathcal{T})\}}} e^{-\nu_{\text{song}} \Delta(s_i^{(t-1)}, s_i^{(t)})^2}, \quad (7)$$

where users and time-dependent song positions are optimized.

From a technical perspective, it is conceivable to train an embedding model with both users and songs varying their position over time, which will output an embedding matrix X of $(|\mathcal{T}|(|\mathcal{S}| + |\mathcal{U}|))$ rows. We briefly explored this model, but found it difficult to interpret the resulting trajectories. We therefore focus on the restricted models in our empirical evaluation.

3.4 Training of Probabilistic Embedding Models

The maximum likelihood optimization problems in Equations (7) and (5) are of substantial scale. Previous sequence models were trained using stochastic gradient methods [1, 2, 5]. However, those training algorithm does not scale well, since the complexity of each training iteration is quadratic in the number of terms in the partition function (in our case $|\mathcal{S}|$). In related work on (non-temporal) sequence modeling for natural language [6], we developed an approximate, sampling-based training algorithm which estimates the partition function on the fly. This training procedure has complexity which is only linear in the number of terms in the partition function, and we adopt this algorithm for training. A software package implementing the training algorithm is available online at `http://lme.joachims.org`.

4. EXPERIMENTS

Our experiments revolve around a *Last.fm* data set which we crawled using the site's API [1]. The crawl was conducted over the course of several weeks in the fourth quarter of 2012. Although it is unused in this work, we were initially also interested in the social network data, so we

[1] http://www.last.fm/api

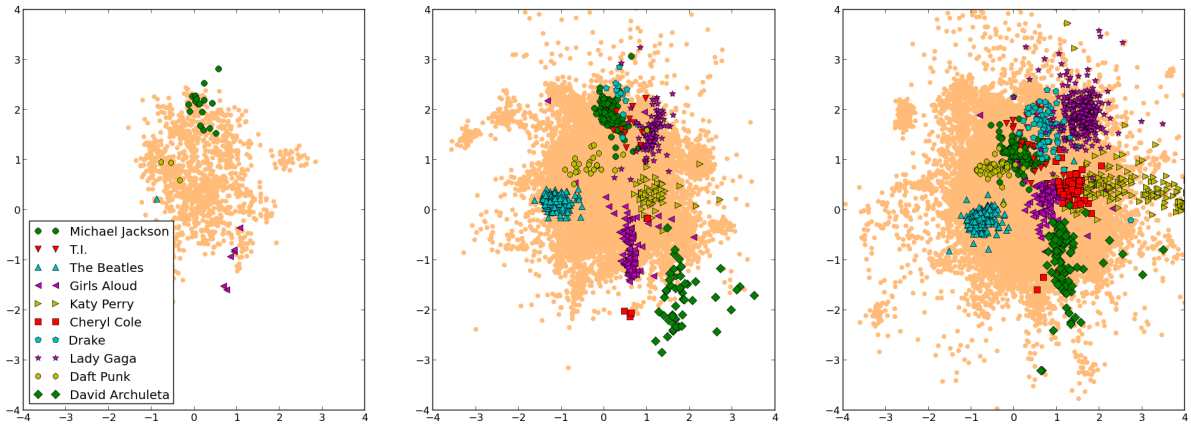


Figure 2: The song-dynamic model's song space plotted (from left to right) at 20051, 20091, and 20124

crawled through the social network using the top listener for each of the 10 top artists on the site at the time as seeds. For each user, we crawled the user's complete timestamped track history and friends list. We later augmented this data with the age, gender, and country of each user (for those for which it was available). We also crawled the tags for some of the songs, although we do not take advantage of this data in this work.

The result contains over 300,000,000 track plays by roughly 4700 users, with over 550,000 unique tracks. This data contains many noisy track names, so we pruned the data further by only considering tracks with at least 1000 plays and discarding users with no remaining track history after infrequent songs are discarded. This yields the set of track histories used in the experiments, which contains 4,551 users, 32,401 unique tracks, and roughly 200,000,000 track plays. We used this to create our "per-user playlist" data by splitting the track histories into playlists of consecutive songs that were played within 15 minutes of each other. Finally, we quantized the timestamps to divide each user's track history into year quarters, ranging from first quarter, 2005 until fourth quarter, 2012, for a total of 32 timesteps. From this point on, we will refer to the nth quarter of year $yyyy$ as $yyyyn$, such as 20051 for 2005 first quarter.

We considered models with 2 dimensions in this work for the sake of simplicity and ease of visualization. In order to find good values for ν_{song} and ν_{user}, we further divided the data by placing each fifth song transition into a validation set and the rest into the training set. We then used these to validate for the optimal values of these parameters. The user-dynamic model performed best with a low value of ν_{user}, with its optimal value at 0.01. In contrast, the song-dynamic model performed best with strong regularization, and the optimal ν_{song} was found to be 2.0.

4.1 Demographics of users

The demographics of the data set reflect characteristics of the average Last.fm user. For each demographic category, we report percentages based on the number of users reporting in that category. 83% reported an age, 89% reported a country, and 91% reported gender. In our data, about

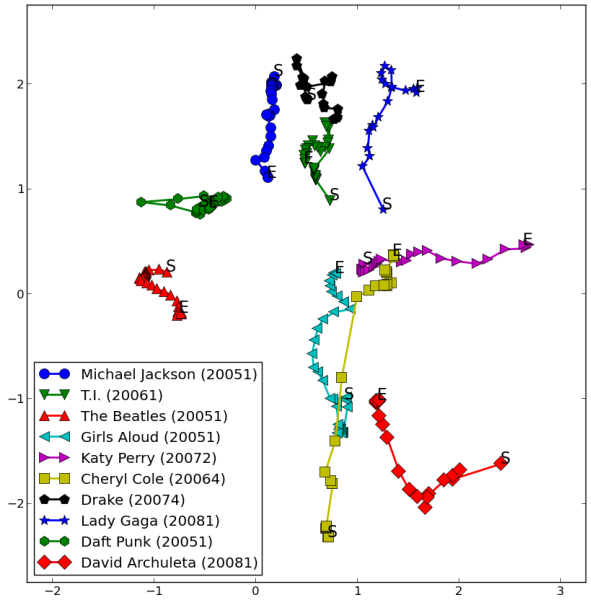


Figure 3: Artist trajectories over time. The legend gives the first quarter in which each artist was observed

78% of the users are male, and about 88% are between the ages of 15 and 25 (roughly evenly split between the two groups) as of the crawl in 20124. The median user age is 20, and the average is about 20.8. Due to the social network crawl and a coincidence of the seed users, roughly 57% of our users are from Brazil. The country distribution has a fairly long tail, with only 84% coming from the 10 most popular countries, and 91% coming from the 20 most popular countries. The ten most well-represented countries in the data set are Brazil (57%), US (8%), UK (4%), Poland (3%), Russia (2.6%), Germany (2.3%), Spain (1.6%), Mexico (1.6%), Chile (1.3%), and Turkey (1.1%).

4.2 Song-dynamic Model

In the song-dynamic model, songs can move over time through a map of users. Among other things, the resulting trajectories give insight into how the appeal of songs and artists changed over time.

In Figure 2, we show the embedding of the songs at the start, middle, and end of the time sequence (i.e., timesteps

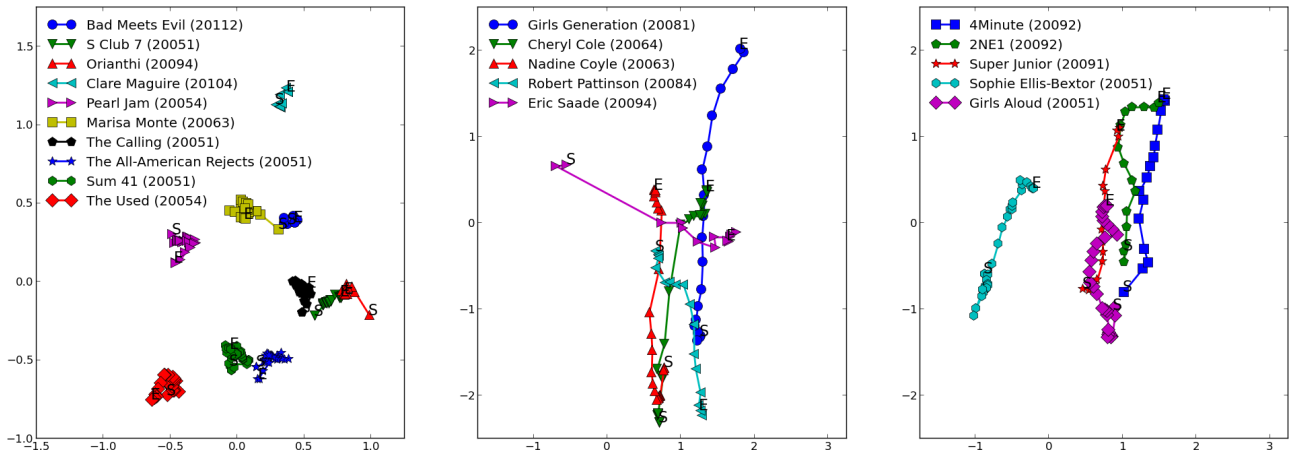


Figure 4: The 10 artists with the smallest variance in position over time (left) and the 10 with the largest variance in position over time (top 5 in center, next five at right). The first timestep at which each artist was observed is listed in parentheses.

20051, 20091, and 20124). A song is plotted once it has been played at least once, which explains why the space becomes more dense over time. The locations of users are not plotted to reduce clutter. Generally speaking, the density of users is greatest around the origin and then decreases outwards. In this sense more popular music lies in the center, but note that we also capture popularity through the specific song bias parameter.

Are similar songs embedded at similar locations? To illustrate the semantic layout of the embedding space, we highlight the songs of some reference artists. Note that the songs of the reference artist cluster even though our embedding method has no direct information about artists. This verifies that the model can indeed learn about songs similarity merely from the listening patterns. We also note that our intuitive notion of artist similarity generally matches the distance at which our model positions them in embedding space.

How do songs and artists move? Figure 2 also shows that the songs of some artists move in the embedding space, while others remain more stationary. The artists' changes are aggregated into trajectories and displayed in Figure 3. Each dot in Figure 3 indicates the mean location of the songs of one artist at a specific time step. This plot enables us to see more clearly some events and trends in the music world that influence the model.

First, note that Michael Jackson's trajectory starts off clumped together in the same space, moving very little. Then, after some number of timesteps, it starts moving quickly towards the center. Upon closer inspection, the turning point in this trajectory turns out to line up exactly with the death of Michael Jackson in June, 2009.

Similarly, the Beatles start to drift slightly away from the center as many other artists enter the model. Then, they make an abrupt turn back towards the center. This aligns with the release of the Beatles' full catalog on iTunes in the 20104 after being totally unavailable via digital distribution before then.

Daft Punk also starts to drift away from the center until the release in December, 2010 of the motion picture *Tron: Legacy*, which featured a popular soundtrack by the duo.

We can also see Girls Aloud and Cheryl Cole (of Girls Aloud) drift from the edges rapidly towards the center in correlated paths, and the emergence of David Archuleta, an American Idol runner-up in May, 2008. All follow a similar trajectory in user space, indicating that the users that previously listened to Girls Aloud are listening to David Archuleta a few years later.

We can also see artists like Katy Perry and Lady Gaga drift away from the center after the peak of their popularity, and we see Drake drift towards the center in what can partly be explained by a shift in his style from something more hip hop oriented to a somewhat more poppy style.

What does the variance of a trajectory indicate? The trajectories are useful not only for visualization, but also as the basis for further aggregating and quantifying the behavior of an artist. Figure 4 shows the artists with the smallest and largest variance in position over time. The specific criterion used here for a given artist is the average distance over timesteps from the artist's embedding at that timestep to the mean vector of that artist's representation over all time steps. To avoid obscure artists that would be difficult to interpret without further background knowledge, we only consider artists who appeared in the track histories of at least 10% of the users.

The left-hand panel in Figure 4 shows the 10 artists with small variance. Many of these are well-established artists that probably undergo little change in style or fan base.

The panel in the middle and on the right-hand side of Figure 4 show the 10 artists with the largest variance. Many of these are popular artists that have a large change in appeal – i.e., those that go from being relatively obscure to quite popular.

The variance of a trajectory in only one possible statistic that summarizes a path. We conjecture that other summary statistics will highlight other aspects of an artist's development, providing additional criteria for exploratory data

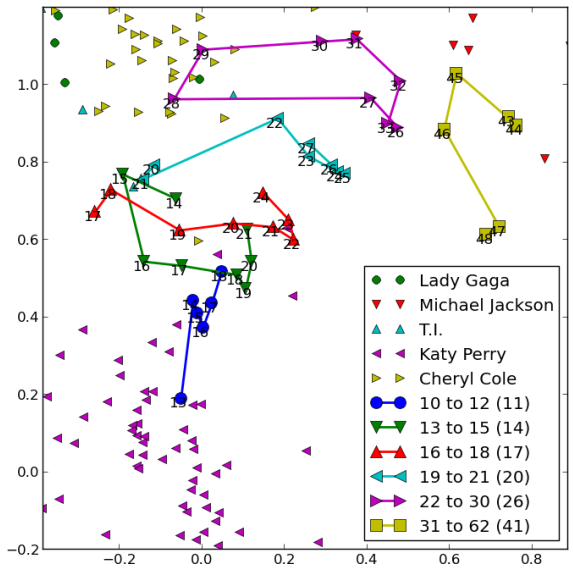


Figure 5: Trajectories of users with age, grouped by age in 2005. Each point is labeled with the average age of the group at that time. The legend also gives the average age in 2005 of the users in that group (in parentheses).

analysis.

4.3 User-dynamic model

The user-dynamic model is dual to the song-dynamic model, in that it models trajectories of users on a map of songs. While the trajectories of indiviual users provide an intersting tool for reflection, they are difficult to interpret for outsiders. We therefore only show aggregate user paths.

One such aggregation is shown in Figure 5. Here, we can see the behavior of users when aggregated by age. Specifically, the users are grouped by age in 2005 in order to separate the effect of a person's absolute age from the effect of the change in the average listener's taste profile.

Distinctive differences in trajectory can be seen, with the youngest group moving to north, away from Katy Perry and many other more "sugary" pop artists, and towards more dance and R&B oriented pop artists as well as the hip hop cluster which is further north, outside the figure.

The other age groups see more lateral moves and tend to be further north, even when age is fixed. The oldest age groups (where 22 to 30 and 31 to 62 were aggregated with a larger interval due to a smaller number of users in these age ranges) start very far north, and the 31 to 62 group mostly hovers around the eastern part of the figure. Outside of the figure and to the right are where many older rock bands such as the Rolling Stones and the Beatles lie, and this oldest age group is also closer to them.

5. CONCLUSIONS

We presented novel probabilistic embedding methods for modeling long-term temporal dynamics of sequence data. These models jointly embed users and songs into a metric space, even when no features are available for either one. Users and/or songs are allowed to change position over time, which enables the analysis of long-term dynamics of user tastes and artist appeal and style. The ability to visualize the learned embeddings is a key feature for easy interpretability and open-ended exploratory data analysis. We conjecture that such embedding models will provide interesting tools for analyzing the growing body of listening data. Furthermore, the embedding models described in the paper can easily be adapted and extended to include further information (e.g., social network data), providing many directions for future work.

5.1 Acknowledgements

This work was supported by NSF grants IIS-1217485, IIS-1217686, and IIS-1247696. The first author is supported by an NSF Graduate Research Fellowship. We would also like to thank the anonymous reviewers for their feedback, and Brian McFee for helpful discussions and technical advice.

6. REFERENCES

[1] N. Aizenberg, Y. Koren, and O. Somekh. Build your own music recommender by modeling internet radio streams. In *Proceedings of the 21st international conference on World Wide Web*, pages 1–10. ACM, 2012.

[2] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims. Playlist prediction via metric embedding. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 714–722. ACM, 2012.

[3] G. Dror, N. Koenigstein, and Y. Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 165–172. ACM, 2011.

[4] V. Jain and L. Saul. Exploratory analysis and visualization of speech and music by locally linear embedding. *ICASSP*, 2004.

[5] J. L. Moore, S. Chen, T. Joachims, and D. Turnbull. Learning to embed songs and tags for playlist prediction, 2012.

[6] J. L. Moore and T. Joachims. Fast training of probabilistic sequence embedding models with long-range dependencies. *Arxiv pre-print*, 2013.

[7] J. Platt. Fast embedding of sparse music similarity graphs. *NIPS*, 2004.

[8] U. Shalit, D. Weinshall, and G. Chechik. Modeling musical influence with topic models. In *ICML*, 2013.

[9] J. Weston, S. Bengio, and P. Hamel. Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval. *Journal of New Music Research*, 40(4):337–348, 2011.

EXPLORING THE RELATION BETWEEN NOVELTY ASPECTS AND PREFERENCES IN MUSIC LISTENING

Andryw Marques
Universidade Federal de Campina Grande
andrywmr@lsd.ufcg.edu.br

Nazareno Andrade
Universidade Federal de Campina Grande
nazareno@computacao.ufcg.edu.br

Leandro Balby
Universidade Federal de Campina Grande
lbmarinho@computacao.ufcg.edu.br

ABSTRACT

The discovery of new music, e.g. song tracks and artists, is a central aspect of music consumption. In order to assist users in this task, several mechanisms have been proposed to incorporate novelty awareness into music recommender systems. In this paper, we complement these efforts by investigating how the music preferences of users are affected by two different aspects of novel artists, namely familiarity and mainstreamness. We collected historical data from Last.fm users, a popular online music discovery service, to investigate how these aspects of novel artists relate to the preferences of music listeners for novel artists. The results of this analysis suggests that the users tend to cluster according to their novelty related preferences. We then conducted a comprehensive study on these groups, from where we derive implications and useful insights for developers of music retrieval services.

1. INTRODUCTION

The discovery of new songs and artists that one likes is a central aspect of music consumption. To a higher or lesser degree, all music listeners look for novelty over time. Due to the huge music collections presently available to listeners, it becomes increasingly difficult to sift for novel and relevant music. There is thus a potential for efficient and novelty-aware retrieval services that assist listeners.

Both commercial systems and research efforts have made progress in incorporating novelty in music retrieval systems (e.g.. [7,10,11]). In general, novelty concerns an item previously unknown to a consumer. For example, a movie still not watched or a band still not listened to. However, it is also possible to extend this unifaceted view of novelty. For example, an item that is both previously unknown and has a different style from all other previously known items brings to a consumer an extra dimension of novelty. The distinction of extra dimensions in novelty is relevant as consumers might have preferences for novel items along multiple of such dimensions: some users may prefer novel items that are similar (or familiar) to their current preferences, but not mainstream (or popular), and vice-versa. Although sometimes mentioned in the literature, this multifaceted view of novelty remains largely unexplored in the Music Information Retrieval field.

In this work we conduct an exploratory analysis of the impact of different dimensions of novelty in the preferences of music listeners. Our ultimate goal is to elucidate what kinds of novelties would be preferred by music listeners, thus paving the way for more efficient and informative music retrieval services. More concretely, we examine how two important aspects of music novelty – familiarity and mainstreamness – affect the preferences of music listeners to the new artists they discover.

Our analysis relies on historical music listening data from 17,000 Last.fm users (Section 3), and on a model of listening behavior and novelty aspects we propose (Section 4). On the one hand, our results suggests that there is no global correlation between familiarity or mainstreamness and novelty relevance for the listeners in our sample. On the other hand, when considered individually, most listeners do have a significant correlation between either familiarity or mainstreamness of novel artists heard and the relevance they see in these artists (Section 5). This suggests that novelty-based personalized music retrieval systems likely have more chances of success than non-personalized ones.

Another consequence of this result is that listeners form groups concerning their novelty-related preferences. This observation motivates us to perform a cluster analysis on the listeners. This analysis unveils seven archetypical profiles that explain how different groups have preferences for the different aspects of novelty we consider (Section 6).

2. RELATED WORK

Several studies point to the relevance of considering novelty in music recommendation systems. In general, different efforts agree that it is necessary to evaluate such systems not only by their accuracy and recall, but also by how many new and relevant items are introduced to a user [4,5]. In materializing this view, several studies in the literature discuss how to build recommender systems that introduce novelty adequately [7, 10, 11].

Specifically about novelties, Vargas et al. [8] put forward a formal framework to analyze the relation between

users, items, and novelty in a recommendation system. The authors make a distinction between choice (a user picks an item), discovery (a user is introduced to an novel item) and relevance (a user likes an item).

Related to the discovery, novel items can be defined in a twofold manner [1]. On the one hand, novel items have (usually content-based) characteristics not shared by items previously declared as relevant by the user [7, 11]. On the other hand, novel items are defined in terms of popularity among users, mainly non popular items. This attribute of items is influential in their discovery potential [2, 4], thus interplaying with their novelty.

With respect to the reaction to novelty in music, a related field is that of the reaction to new opinions (in general). Munson and Resnick [6] ran experimental studies that suggest that online users can be clustered into three distinct subgroups with respect to their preferences for new opinions: diversity-seeking, challenge-averse and support-seeking.

In general, although past work has utilized the concept of novelty for recommending music and other types of items, and has sometimes dealt with the relation between novelty and diversity or popularity, the literature currently lacks studies specifically formalizing novelty aspects and relating them to the relevance of novel items. This study contributes to fill this gap, formalizing the concepts of familiarity and mainstreamness of novel music artists in relation to a listener. According to Vargas et al.'s framework, we relate these two aspects with the relevance of items chosen by listeners so as to inform the design of discovery mechanisms. Finally, our results with respect to different types of listeners with respect to their reaction to novelty bears similarities to Munson and Resnick's results.

3. DATA COLLECTED

Our dataset is comprised of two parts: the first contains the subjects whose behavior will be analyzed, together with their historical listening habits, and the second concerns metadata about the artists listened by the chosen subjects. All data was collected from Last.fm through its publicly available data access API.

For the remainder of this study, we differentiate between the *experiment period*, set as the six months between March to September 2012, the *observation period*, which includes the experiment period and also the following six months, and the *prior listening history*, which is the year preceding the experiment period. The distinction between experiment and observation period is used to control the bias in relevance measuring, and is further discussed in Section 4.4. Figure 1 illustrates the periods on a timeline.

3.1 Subject data

Our goal is to identify and collect data about a sample of Last.fm users which has been exposed to novel artists during a period of interest. Starting with the profile of the first author, a snowball sampling procedure was conducted on the Last.fm friendship network until 100,000 users were identified. Next, data about the artists listened to by the users prior and during the experiment period were collected. The former is collected as the set of the 200 artists most listened by the user since he or she joined Last.fm, plus the union of the sets of 100 most listened artists by the user in each week of his or her prior listening history. The data collected about the listening habits during the observation period is the union of the sets of the 100 most listened artists by the user in each week in this period.

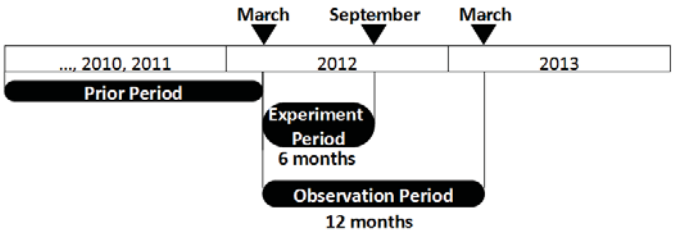


Figure 1: Time periods used in the experiment.

After this data collection, a filtering process was conducted, with the goal of selecting the users who (i) have highly active listening habits, (ii) are likely to have informed a large portion of these habits to the system we collected the data from, (iii) have experienced a number of novelties in their listening that enable us to investigate the relation between novelty characteristics and user preferences, and (iv) likely gear most of their music listening, instead of using a Last.fm's radio created by a recommendation algorithm [1].

For (i) and (ii) we kept only users who had at least 100 artists in their prior listening history, and which were active in at least three quarters of the weeks in the observation period, having at least 100 song executions in a week to be considered active. For (iii), we selected the users with at least 10 novel artists in the experiment period that were listened to at least 10 times each. For (iv), we infer that a user does not chiefly use the Last.fm recommendations if that user listened 15 or more songs of a same artist in a week. Our assumption is that this denounces a level of gearing incompatible with radio licensing. Finally, we filtered out also a small number of registered users with unrealistically high music listening frequency (more than 16 hours a day during the observation period). This process produced a sample with 17,183 Last.fm users.

3.2 Artist data

For each artist present in the subjects' listening data, artist popularity and tag data was collected from Last.fm. Popularity data was collected as the number of users who listened the artist as informed by Last.fm on March 10th 2013. A tag is a label given by a user to the artist, ranging, in Last.fm, from music genres (eg. rock, samba) to mood (eg. sad, lively) and other contextual metadata (eg. summer, lovesong). Each tag has a popularity, reflecting how often users have assigned the tag to a given artist. After

[1] http://www.lastfm.com/listen

collecting tag data from all artists in the subjects listening data, we filter out unpopular tags, defined as having popularity lower than 0.15 of the most popular tag for the artist. Furthermore, tags denoting personal circumstances, such as *seen live* and *favorite* were manually removed.

4. CHARACTERIZING LISTENER PROFILES AND NOVELTIES

To examine how different characteristics of novel items affect their relevance for listeners, we resort on three constructs we now describe in turn: a model of a listener profile, a set of dimensions on which we consider novel items may vary, and a definition of relevance.

4.1 Novel Items

For our experiment, the items considered are artists, and an artist is novel for a subject if two conditions are met: (a) this artist is absent from the subject's listening history prior to the experiment period and (b) the artist was listened to at least five times in a week by this subject during the experiment period. The experiment period is used for identifying novel artists listened by the subject, while the whole observation experiment period is used for an unbiased evaluation of how much attention the subject payed to these novel items (a process detailed in Section 4.4).

With this definition, we identify 652,511 novel items in our data, with a subject discovering on average 38.3 (std. dev. 31.2) novel artists during our measurement.

4.2 Listener Profiles

We model each listener profile as a set of clusters of artists in his or her listening history prior to the experiment period. These clusters are obtained applying the DBScan clustering algorithm [3] to the set of artists known to the listener are not considered to be novel to the user in our experiment.

Artists are represented as vectors, where each vector component represents the number of times a given tag was assigned to the artist being considered. More formally, let A be the set of artists, and T the set of tags. Let $f : A \times T \rightarrow \mathbb{R}$ denote the frequency of which a given tag $t \in T$ was assigned to a given artist $a \in A$.[2] Now, the vector representing an artist $a \in A$ is defined as $\vec{a} := (f(a,t_1), f(a,t_2), \ldots, f(a,t_{|T|}))$ The cosine similarity between two artists is defined, as usual, as:

$$\cos(\vec{a},\vec{a}') := \frac{\langle \vec{a},\vec{a}' \rangle}{\|\vec{a}\|\|\vec{a}'\|} \quad (1)$$

The clusters are then computed based on the cosine similarity between the vector representations of artists as mentioned above. The two parameters of the DBScan algorithm – the minimum number of points for a cluster to be considered and the minimum similarity for two points in a cluster – were empirically defined as 3 and 0.875 (resp.) after consulting an online community of music aficionados [3] about which of several clustering solutions better described members' taste profiles. Profiles obtained through this process for our subjects have an average of 5.5 (std. dev. 2.9) clusters found among the artists present in the subject's listening history.

4.3 Characteristics of novelties

We consider novel items/artists have two other aspects of novelty besides being unknown to a listener: familiarity and mainstreamness. The former captures the notion that characteristics of the music from an artist may or may not be familiar to a listener. The latter models the intuition that the artist, although not listened, may be known to the listener through other media and the general public acclaim, ie. mainstream.

4.3.1 Familiarity

Familiarity is defined based on the similarity between an artist and those that compose a listener profile. This models the intuition that for a jazz fan, novel heavy metal artists are likely to sound less familiar than a novel artists from a subgenre of jazz. Formally, let $P := \{C_1, ..., C_n\}$ denote a listener's profile, formed by the clusters of artists C_i. Also, let $\vec{c_i}$ be the centroid of the cluster C_i, and p_i be the proportion of all song executions by the listener in his or her prior listening history that were from artists in C_i. Then, the familiarity between an artist a and a listener profile P is the weighted arithmetic mean of the similarity between the artist and the centroids $\vec{c_i}$, with p_i as weights:

$$\text{fam}(a,P) = \frac{\sum_{i=1}^{n} cos(\vec{a},\vec{c_i}) \times p_i}{\sum_{i=1}^{n} p_i} \quad (2)$$

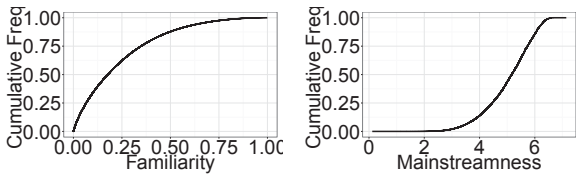
Figure 2(a) displays the cumulative distribution of the familiarity of all novel artists in our experiment to their listener profiles. Notice that there is an overall skew towards more familiar artists.

4.3.2 Mainstreamness

For our experiment, mainstreamness is defined as the log of the overall popularity of the artist in Last.fm. This definition aims to capture how likely it is that the artist and an opinion about the artist are known to a subject before the subject has carefully listened to the artist. This knowledge likely comes from other medias, such as mentions in newspapers or television, and from the subject's social network. Our hypothesis is that some users may have a preference for mainstream novel items known to be liked by the general public. Because the artists popularity distribution observed is highly skewed, our analysis uses the log of this popularity in all results reported. Figure 2(b) displays the cumulative distribution of the mainstreamness of all novel artists in our experiment. This distribution has a skew towards highly mainstream artists.

[2] http://www.lastfm.com/api

[3] A facebook group of music releases named 'Revista Billboard'



(a) Familiarity. (b) Mainstreamness.

Figure 2: Cumulative distribution of Familiarity and Mainstreamness for all novel artists in the experiment period.

4.4 Preferences for novel items

We measure subjects' preferences for different types of novel artists gauging how much or how often each novel artist was listened to during the observation period. An artist/item is thus said more relevant than another if the former was listened to more times or more frequently than the latter during a time span.

To put forward this approach we consider two metrics for item relevance in our analysis. Let δ be the period between the first time a subject listened to a novel artist and the end of our observation period, then (a) the *total attention* a subject gives to an artist is the total number of executions of songs of this artist during δ divided by the number of weeks in δ; and (b) the *period of attention* of a subject to a novel artist is the fraction of the number of weeks in δ in which the subject listened at least once to the artist. Because novel artists are found over a period of time, some of these artist have a smaller time window in our experiment period in which they can be listened to. We tackle this issue with two measures. First, taking δ as the denominator of attention metrics limits the potential bias in the counting of song executions or weeks of attention. Second, as mentioned in Section 4.1, only novelties discovered in the experiment period are considered in the analysis, but the whole observation period is used to count song executions and weeks of attention. This gives each novel item a minimum of six months of observation in our traces, which limits possible biases caused by too short observations.

Figure 3 shows the cumulative distribution of the total attention log and period of attention for each novelty found by the subjects in our experiment. In both cases, attention is concentrated on a small proportion of the novel items discovered.

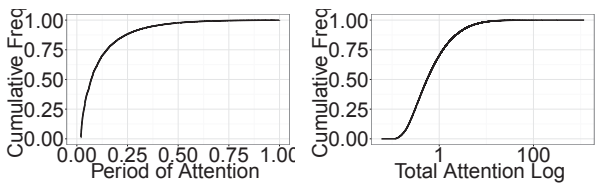


Figure 3: Cumulative distribution of the Total Attention Log and Period of Attention for each novelty

Table 1: Correlation (Spearman's coefficient) between novelty characteristics and relevance, analyzing all novel items together.

Dimensions	PoA	F	M
Total Attention	0.74	0.04	0.04
Period of Attention (PoA)	-	0.04	0.05
Familiarity (F)	-	-	0.01
Mainstreamness (M)	-	-	-

5. PREFERENCES FOR NOVELTY CHARACTERISTICS

Our central research question is to understand how considering different aspects of novelty can enhance our understanding about listener preferences. To address this question, we first evaluate whether there is a correlation between novelty characteristics – familiarity and mainstreamness – and relevance – total attention and attention period.

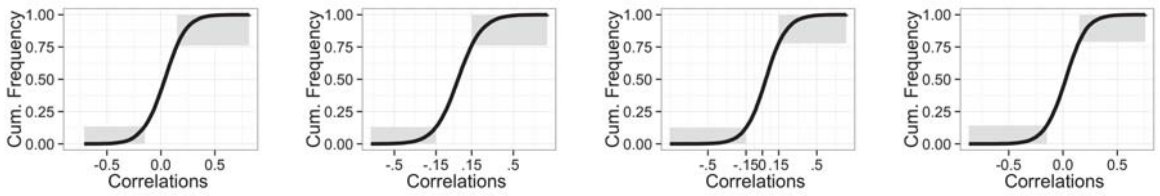
Table 1 shows that analyzing all novel items together, there seems to be no relevant correlation between novelty characteristics and relevance that is valid for all subjects. On the other hand, analyzing the correlation between novelty characteristics and relevance for the novel items of each subject individually, there is a different pattern. For each pair of variables we examine, there is a significant correlation (above 0.15 or below -0.15) between the variables for approximately one third of all subjects (Figure 4). In aggregate, 70% of all subjects have in their data a significant correlation between at least one of the novelty characteristics and a measure of relevance.

Taken together, our results point that *while there is no overall common behavior regarding subjects' preferences for different types of novelty, most users have some preference for a type of novelty in their listening behavior.*

6. LISTENER GROUPS WITH RESPECT TO NOVELTY

The analysis of the correlation between novelty types and preferences indicates that there are different types of listener in our data. In this section we apply clustering analysis to delve further in the identification and analysis of such groups. For that, we use only the set of subjects for which at least one of the correlations analyzed in the previous section is not in the [-0.15; 0.15] interval (total=12,114).

Our analysis uses the Ward hierarchical clustering method [9] considering normalized versions of two dimensions that describe each subject's preferences for novelty aspects and two dimensions that measure the subject's usual musical taste. The dimensions related to preferences are (a) the correlation between total attention devoted to a novel artist and the artist's familiarity for the listener and (b) the correlation between total attention to a novel artist and the artist's mainstreamness. To represent usual tastes, we use (c) the average mainstreamness of the artists in the listener's profile, and (d) the number of clusters present in the subject's listening profile. Upon examination, the cor-



(a) Period of Attention and Mainstreamness (b) Period of Attention and Familiarity (c) Total Attention and Familiarity (d) Total Attention and Mainstreamness

Figure 4: Distribution of the Spearman correlations coefficient between novelty characteristics and relevance in each listener's data. Shaded areas highlight the portion of subjects with correlation higher than 0.15 or lower than -0.15.

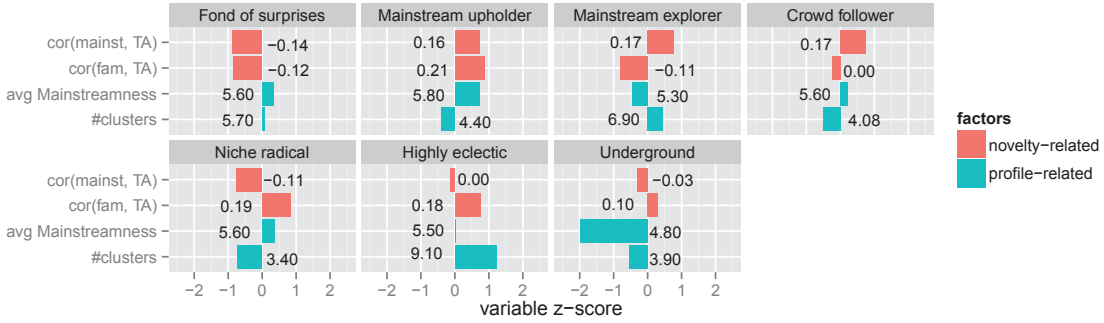


Figure 5: Centroids of the seven clusters found in the analysis. Variables as normalized as the z-score (horizontal axis), where zero represents the average across all listeners, and the unit of variation is a standard deviation in the considered metric. In the vertical axis, fam stands for familiarity, mainst for maisntreamness and TA for total attention. The numbers on the chart are the denormalized values.

relations between novelty aspects and period of attention for novel artists were discarded as redundant dimensions for the clustering analysis.

Applying this approach points to satisfying clustering solutions with respect to intra and intergroup heterogeneity that have between 5 and 8 clusters. Further examination considering the descriptive power of the solutions leads to the choice of the 7-cluster solution, whose centroids are shown in Figure 5. Analyzing these centroids, we labelled the groups as follows:

1. *Fond of surprises*: The largest cluster (n=2,738), contains subjects with a distinct preference for novelty that is both unfamiliar and of lower mainstreamness.

2. *Mainstream upholder*: Subjects with a listening history marked by mainstream artists, and which clearly prefer novelty of higher mainstreamness and familiarity (n=1,552).

3. *Mainstream explorer*: Listeners with noted preference for novel artists from the mainstream, but who value new artists that are not familiar. Seem to be exploring and discovering relevant artists among the mainstream that were previously outside their profile, which is itself formed by a high number of clusters (n=1,535).

4. *Crowd follower*: Subjects with a limited number of clusters in their listening profile, and with a clear preference for mainstream novelty (n=2130).

5. *Niche radical*: Listeners who habitually focus on a small number of clusters of artists, and which have a strong preference for novelty familiar and of below-average mainstreamness, likely in one or more niches (n=1,172).

6. *Highly eclectic*: Somewhat the opposite of niche radicals, these are subjects whose listening profiles had high number of artist clusters, and who value novelty already familiar (n=1,874).

7. *Underground*: Subjects who usually listen to artists far from the mainstream, have a relatively homogeneous listening profile, and prefer novelty close to this profile (n=1,143).

Interestingly, the largest single cluster in our results — and simultaneously one of the most distinct of them — is that of subjects fond of surprises. These are listeners who exhibit a clear preference for diversity and novelty among lesser known artists. Our results point that subjects in this group are interested in discovering new artists in the long tail of popularity, and at the same time exploring unfamiliar genres, even though their profile is, on average formed by mainstream artists. As an example of fond of surprises, a listener has a profile composed by *Indie* artists, but he preferred as novel artists some *Country* unpopular artists.

Diametrically opposite to subjects in this cluster, those in the niche follower group seem eager to discover more

music similar to that they have listened in the past. Moreover, what was listened in the past is focused on a small area of the artists space, and novelty is more appreciated if it has low mainstreamness.

Not surprisingly, taken together, subject groups related to a marked preference for mainstream novel artists (mainstream upholders, mainstream explorers and crowd followers) form a large portion (43%) of our sample. Nevertheless, we see a distinction between listeners who have a markedly mainstream listening history prior to the experiment (mainstream upholders), and those who don't. As an example of mainstream upholder, a listener has a profile composed by popular *Indie* artists and he liked others popular *Indie* artists as novel items.

Mainstream explorers seem to be subjects who discovered a set of new mainstream artists during the experiment period that are different from their previous profile. As an instance of this group, a listener has a profile composed of *Sludge Metal* and *Hardcore* artists, but he preferred some popular *Hip-Hop* artists as novel items. Crowd followers, on the other hand, are subjects with a simpler profile previous to the experiment, and which enjoined novelty according to mainstreamness, irrespective of similarity to their previous profiles.

The underground cluster seems to capture listeners who devote their attention to a set of artists extremely different from the mainstream. Finally, the highly eclectic cluster models the profile of listeners who transit among many artist clusters, but which have marked preferences for novelty based on this diverse profile.

7. DISCUSSION AND IMPLICATIONS

The main objective of this study is to investigate the use of multiple dimensions of novelty to further understand listeners' preferences regarding novel artists. This understanding, in turn, is aimed at improving the design of music information retrieval systems.

Our main findings are threefold. First, there is no overall correlation between familiarity of mainstreamness and relevance in the novel artists discovered by our subjects. Second, when considered individually, most subjects had a clear correlation between either familiarity or mainstreamness and the relevance seen in novel artists. Third, it is possible to cluster listeners with some correlation between novelty aspects and preferences in seven groups that reveal how different audiences approach novelty.

Considered together, these results point to the need of a personalized approach in assisting a listener to discover relevant novel artists. Moreover, this personalization should be related not only to which other artists have been enjoyed before, but should be also aware that some listeners have clear preferences regarding how mainstream or familiar novel artists are. The information that these dimensions are relevant to model listeners' behavior, and that it is highly variable among a listener population should be considered in the design of future mechanisms. On another perspective, the groups we find to describe the archetypical behaviors among our subjects can be used to develop interfaces or mechanisms that target different types of users.

There are a number of opportunities on our study and directions in which future work could expand it. Notedly, considering data from a population outside Last.fm is necessary to evaluate the generalizability of our results. Also, our clustering is aimed at a descriptive analysis. Finding the listener clustering that provides maximum efficacy gain to a predictive model is likely a fruitful avenue of work. Finally, the validation of the use of familiarity and mainstreamness as dimensions of novelty in an experiment of musical recommendation is necessary to further validate the use of multiple dimensions to address novelty.

8. REFERENCES

[1] A. Bellogin, I. Cantador and P. Castells "A Study of Heterogeneity in Recommendations for a Social Music Service," *Proc. of the 1st HetRec*, pp. 1–8, 2010.

[2] Ò. Celma, P, Herrera: "A new approach to evaluating novel recommendations," *Proc. of the ACM RecSys 2008*, pp. 179–186, 2008.

[3] M. Ester, H. Kriegel, J. Sander and X. Xu: "A density-based algorithm for discovering clusters in large spatial databases with noise," *Proc.of the ACM KDD, pp. 226-231*, pp. 226–231, 1996.

[4] J. L. Herlocker, J. A. Konstan, L. G. Terveen and J. T. Riedl "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, Vol. 22, No. 1, pp. 5–53, 2004.

[5] S. M. McNee, J. Riedl and J. A. Konstan "Being accurate is not enough: how accuracy metrics have hurt recommender systems," *Proc. of CHI '06*, pp. 1097–1101, 2006.

[6] S. Munson and P. Resnick "Presenting diverse political opinions: how and how much," *Proc. of CHI*, pp. 1457–1466, 2010.

[7] M. Nakatsuji, Y. Fujiwara, A. Tanaka, T. Uchiyama, K. Fujimura, T. Ishida: "Classical music for rock fans?: novel recommendations for expanding user interests," *Proc. of the 19th ACM CIKM*, pp. 949–958, 2010.

[8] S. Vargas and P. Castells: "Rank and relevance in novelty and diversity metrics for recommender systems," *Proc. of the ACM RecSys 2011*, pp. 109–116, 2011.

[9] J. H. Ward Jr. "Hierarchical Grouping to Optimize an Objective Function," *Journal of the American Statistical Association*, Vol. 58, No. 301, pp. 236-244, 1963.

[10] Y. C. Zhang, D. O. Séaghdha, D. Quercia and T. Jambor: "Auralist: introducing serendipity into music recommendation," *Proc. of the WSDM*, pp. 13–22, 2012.

[11] C. Ziegle, S. M. McNee, J. A. Konstan, G. Lausen: "Improving recommendation lists through topic diversification," *Proc. of the 14th WWW*, pp. 22–32, 2005.

Poster Session 3

SPECTRAL CORRELATES IN EMOTION LABELING OF SUSTAINED MUSICAL INSTRUMENT TONES

Bin Wu[1], **Simon Wun**[1], **Chung Lee**[2], **Andrew Horner**[1]
[1]Department of Computer Science and Engineering,
Hong Kong University of Science and Technology, Hong Kong
[2]The Information Systems Technology and Design Pillar,
Singapore University of Technology and Design, 20 Dover Drive, Singapore 138682
{bwuaa, simonwun}@cse.ust.hk, im.lee.chung@gmail.com, horner@cse.ust.hk

ABSTRACT

Music is one of the strongest inducers of emotion in humans. Melody, rhythm, and harmony provide the primary triggers, but what about timbre? Do the musical instruments have underlying emotional characters? For example, is the well-known melancholy sound of the English horn due to its timbre or to how composers use it? Though music emotion recognition has received a lot of attention, researchers have only recently begun considering the relationship between emotion and timbre. To this end, we devised a listening test to compare representative tones from eight different wind and string instruments. The goal was to determine if some tones were consistently perceived as being happier or sadder in pairwise comparisons. A total of eight emotions were tested in the study. The results showed strong underlying emotional characters for each instrument. The emotions Happy, Joyful, Heroic, and Comic were strongly correlated with one another. The violin, trumpet, and clarinet best represented these emotions. Sad and Depressed were also strongly correlated. These two emotions were best represented by the horn and flute. Scary was the emotional outlier of the group, while the oboe had the most emotionally neutral timbre. Also, we found that emotional judgment correlates significantly with average spectral centroid for the more distinctive emotions, including Happy, Joyful, Sad, Depressed, and Shy. These results can provide insights in orchestration, and lay the groundwork for future studies on emotion and timbre.

1. INTRODUCTION

Music is one of the most effective forms of media for conveying emotion. A lot of work has been done on emotion recognition in music, considering such factors as melody [3], rhythm [18], and lyrics [10]. However, little attention has been given to the relationship between emotion and timbre. Does the timbre of a particular musical instrument arouse specific emotions? For example, the English horn often plays a sad and melancholy character in orchestral music – is the melancholy character due to the instrument's timbre, the melody composers feel inspired to write for the instrument, or both? If listeners just heard an isolated tone from the English horn without a melodic context, would it sound more melancholy than other instruments? This paper addresses this fundamental question.

Some previous studies have shown that emotion is closely related to timbre. Scherer and Oshinsky found that timbre is a salient factor in the rating of synthetic tones [17]. Peretz *et al.* showed timbre speeds up the discrimination of emotion categories [15]. Bigand *et al.* reported similar results from their study of emotional similarities between one-second musical excerpts [4]. It was also found that timbre is essential to musical genre recognition and discrimination [2, 19].

Little attention has been given to the direct connection between emotion and timbre, though a study by Eerola *et al.* was an excellent start [6]. They carried out listening tests to investigate the correlations of emotions with temporal and spectral sound features. The study confirmed strong correlations between some features, especially attack time and brightness, and the emotional dimensions valence and arousal for one-second isolated instrument tones.

Valence and arousal refer to how positive and energetic a music stimulus sounds [21]. Despite the widespread use of these emotional dimensions in music research, composers may find them vague and difficult to interpret for composition and arrangement purposes. In our study, to make the results intuitive for composers, the listening test subjects compared sounds in terms of emotional categories such as Happy and Sad. The use of emotional categories has been shown to be generally congruent with results obtained using the dimensional model in music [7].

Moreover, we equalized the attacks and decays of the stimuli so that the temporal features attack time and decay time would not be factors in the subjects' judgment. This modification allowed us to isolate the effects of spectral features, such as the average spectral centroid, which strongly correlates with the perceptual brightness of sound.

The next section describes the listening test in detail. We report the listening test results in Section 3. Section 4

discusses applications of the results and questions arising from our study.

2. EXPERIMENTAL METHOD

2.1 Stimuli

The stimuli consisted of eight sustained tones from several wind and bowed string instruments: the bassoon (Bs), clarinet (Cl), flute (Fl), horn (Hn), oboe (Ob), saxophone (Sx), trumpet (Tp), and violin (Vn). They were obtained from the McGill and Prosonus sample libraries, except for the trumpet tone, which had been recorded at the University of Illinois at Urbana-Champaign School of Music. All the tones were used in a discrimination test carried out by Horner *et al.* [9], and six of them were also used by McAdams *et al.* [14].

The tones were used in their entirety, including the full attack, sustain, and decay sections. They were nearly harmonic and had fundamental frequencies close to 311.1 Hz (Eb4). The original fundamental frequencies deviated by up to 1 Hz (6 cents), and were synthesized by additive synthesis at 311.1 Hz. They were stored in the format of 16-bit samples at a 22050- or 44010-Hz sampling rate (depending on the number of harmonics with significant amplitudes).

Duration, loudness, and harmonic frequency deviations were equalized so that these factors would not influence the results. Furthermore, to isolate the effects of spectral features, the attacks and decays of the tone were equalized by time-stretching the actual attacks and decays. As a result, the stimuli were standardized to last for 2 seconds with attacks and decays 0.05 seconds long.

2.2 Subjects

32 subjects without hearing problems were hired to take the test. These undergraduate students ranged in age from 19 to 24. Half of them had music training (that is, at least five years of practice on an instrument).

2.3 Emotion Categories

The subjects compared the stimuli in terms of eight emotion categories: Happy, Sad, Heroic, Scary, Comic, Shy, Joyful, and Depressed. These terms were selected for their relevance to composition and arrangement by one of the authors, who had received formal composition education. Their ratings according to the Affective Norms for English Words [5] are shown in Figure 1 using the Valence-Arousal model. It is worth noting that Happy, Joyful, Comic, and Heroic form one cluster, and that Sad and Depressed form another cluster.

2.4 Listening Test

Every subject made pairwise comparisons of all the eight instruments. During each trial, the subjects heard a pair of tones from different instruments and were prompted to choose which tone more strongly aroused a given emotion. Figure 2 shows a screenshot of the listening test program.

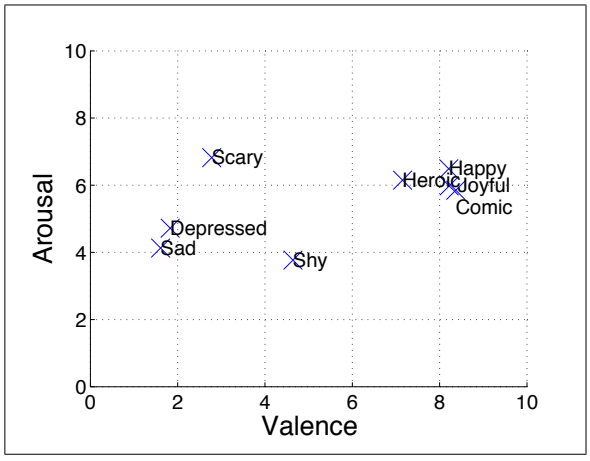


Figure 1. Russel's Valence-Arousal emotion model. Valence refers to how positive an emotion is. Arousal refers to how energetic an emotion is.

Each combination of two different instruments was presented in four trials for each emotion, and the listening test totaled $C_2^8 \times 4 \times 8 = 896$ trials. The overall trial presentation order was randomized.

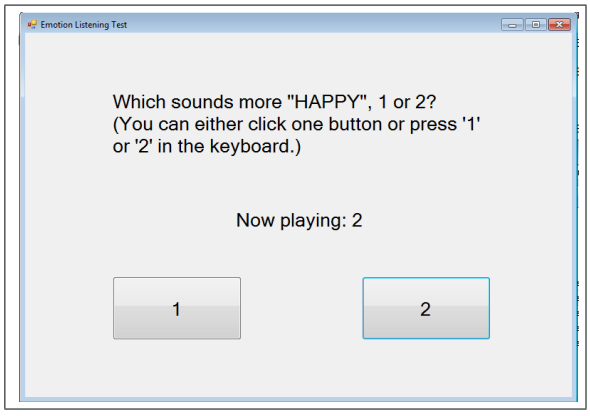


Figure 2. Listening test interface.

Before the first trial, the subjects read online definitions of the emotion categories from the Cambridge Academic Content Dictionary [1]. The listening test took about 1.5 hours, with breaks every 30 minutes.

The subjects were seated in a "quiet room" with less than 40 dB SPL background noise level. Residual noise was mostly due to computers and air conditioning. The noise level was reduced further with headphones. Sound signals were converted to analog by a Sound Blaster X-Fi Xtreme Audio sound card, and then presented through Sony MDR-7506 headphones at a level of approximately 78 dB SPL, as measured with a sound-level meter. The Sound Blaster DAC utilized 24 bits with a maximum sampling rate of 96 kHz and a 108 dB S/N ratio.

3. RESULTS

3.1 Quality of Responses

The subjects' responses were first screened for inconsistencies. A subject's consistency was defined in the four comparisons of a pair of instruments A and B for a particular emotion as follows:

$$consistency_{A,B} = \frac{max(v_A, v_B)}{4} \quad (1)$$

where v_A and v_B are the number of votes the subject gave to the two instruments respectively. A value of $consistency = 1$ represents perfect discrimination, whereas 0.5 represents random guessing. The mean of the average consistency of all subjects was 0.79.

Predictably the subjects were only fairly consistent because of the emotional ambiguities in the stimuli. We assessed the quality of subject responses further using a probabilistic approach. One probabilistic model, successful for image labeling, was adapted to suit this study [20]. The original model took the difficulty of labeling and the ambiguities in image categories into account. This was done to estimate the annotators' expertise and the quality of their responses. Those who made low-quality responses were unable to discriminate between image categories and were considered random pickers. In our study, we verified that the two least consistent subjects made responses of the lowest quality. They were excluded from the results.

We measured the level of agreement among the remaining subjects with an overall Fleiss' Kappa statistic. It was calculated at 0.22, which can be interpreted as indicating fair agreement [12].

3.2 Emotional Judgment

Figure 3 depicts the emotion "voting" results on a gray scale. Each row shows the percentage of positive votes an instrument received when compared to the other instruments for one particular emotion. The lighter the color of a cell, the more positive votes its "row instrument" received when compared to its "column instrument". For example, the bassoon was nearly always judged happier than the horn but usually not as happy as the clarinet.

The subjects gave clear-cut votes to distinctive emotions such as Sad and Depressed, for which the voting patterns have a lot of contrast. On the other hand, there were considerable ambiguities in the emotion comparisons for Scary.

The voting patterns for Sad, Depressed, and Shy were similar, suggesting that these emotions correlate with each other. Likewise, Happy and Joyful form another group of correlated emotions, which apparently includes Heroic and Comic.

We ranked the instruments in order of the number of positive votes they received for each emotion, and derived scale values using the Bradley-Terry-Luce (BTL) model [11]. Table 1 lists the rankings of the instruments, which can be visualized on a BTL scale in Figure 4. The instrument rankings for correlated emotions are similar. The horn and flute ranked high for the sad emotions, whereas the violin, trumpet, and clarinet ranked high for the happy emotions. Note that the oboe nearly always ranked in the middle.

The comparisons between instruments close in rank (e.g., the trumpet and the clarinet) were generally difficult. The votes received by a pair of such instruments could be close, corresponding to the grayest areas in Figure 3. By contrast, instruments ranking in different extremes (e.g., the horn and clarinet) could receive quite different numbers of votes, which correspond to bright and dark areas in Figure 3.

Table 2 shows the spectral characteristics of the test tones such as average spectral centroid. Table 3 shows close ties between test tone ranking and the average spectral centroid, as measured by the Spearman correlation coefficient. Emotional judgment correlated significantly with average spectral centroid, except for the less distinctive emotion Scary. For example, a high-centroid instrument is likely to sound happier, and a low-centroid instrument sadder. Emotional judgment did not have statistically significant correlations with the other spectral characteristics that we tested.

4. DISCUSSION

The pairwise correlations between emotions in our listening test are basically consistent with the pairwise relations between emotional words in the Valence-Arousal model in Figure 1. Both show Scary as the biggest outlier. Both show Happy, Joyful, Heroic, and Comic in a one cluster, and Sad and Depressed in another group. The biggest difference is that Shy is included in the sad group in this study, but it is emotionally-neutral in the Valence-Arousal model.

The results were consistent with those of Eerola's Valence-Arousal results for musical instrument tones [6]. Both show that musical instrument timbres carry cues about emotional expression that are easily and consistently recognized by listeners. Both show that spectral centroid/brightness is a significant component in music emotion.

The main application motivating this study is to provide guidelines for composers/arrangers in orchestration, especially for computer games, film, and stage music that needs to reflect characterization and dramatic action. The results provide some clear guidelines in achieving the desired emotional impact. Composers/arrangers can choose timbres that reinforce the desired emotion of their melody to achieve the strongest emotional impact. For example, composers could combine a joyful melody with instruments that have inherently joyful timbres. On the other hand, instrumental music, like opera arias, often include mixed emotions representing the complexity of characters and dramatic situations. The results of this study also provide a good starting point for achieving mixed emotions. For example, composers/arrangers might create a feeling of overall joyfulness mixed with an undertone of sadness by skillfully combining an otherwise joyful melody with the normally sad horn for a sophisticated mix of emotions.

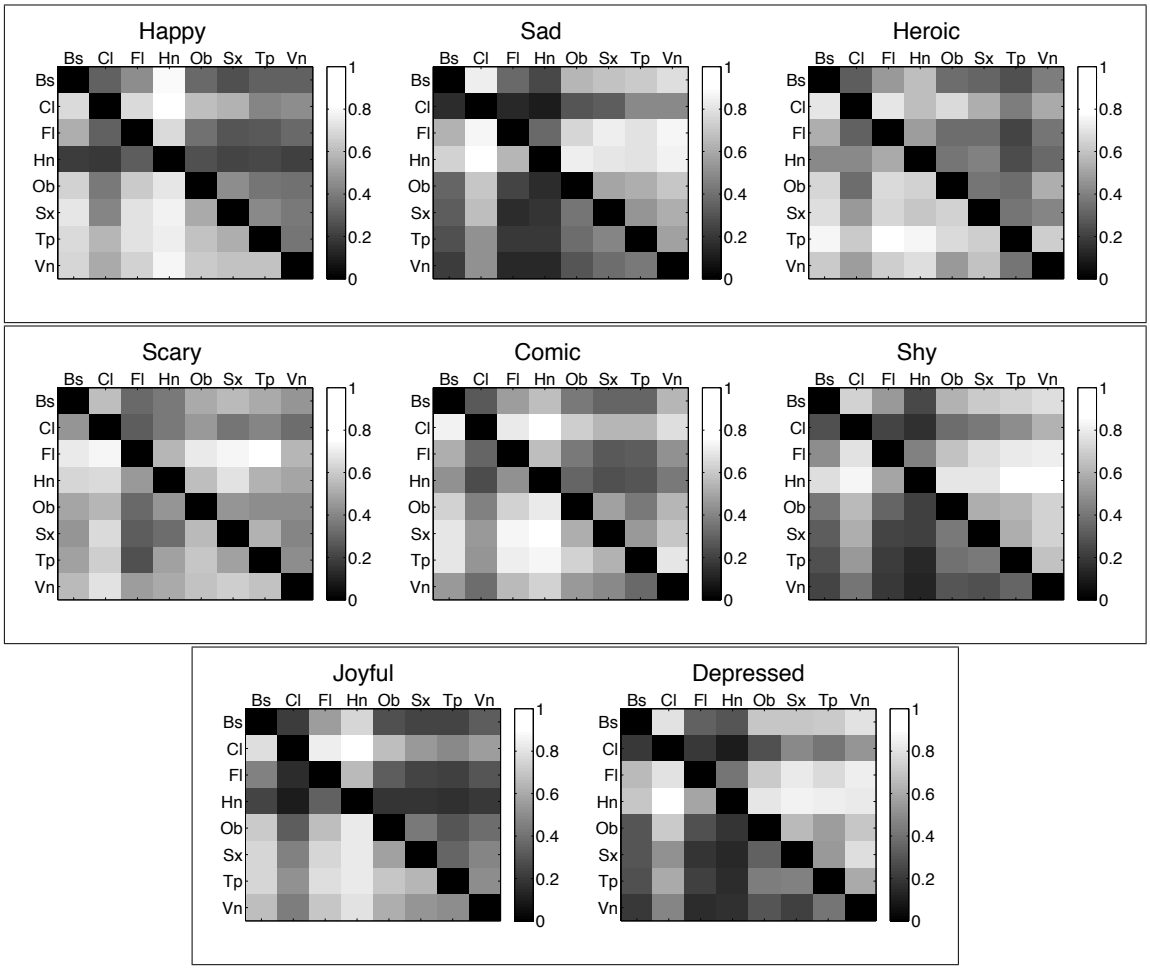


Figure 3. Comparison between instruments for each emotion. The lighter the color of a cell, the more positive votes its "row instrument" received when compared to its "column instrument".

In addition, instrument rankings for the different emotions may serve as a reference for combining instruments that blend together or contrast with each other emotionally.

Moreover, there are many other useful emotional labels to test. For example, one listener commented that the horn was not so much sad as mysterious. Is it the most mysterious timbre? What makes the instrument sound mysterious? Is it due to the reverberant reflections it makes? We could test the effect of reverberations by using tones recorded in an anechoic chamber and comparing them to those recorded in halls with varying reverberation times. In general, how does reverberation influence emotion? Does it make the sound more mysterious, majestic, or heroic? What is the effect of reverberation time and reverberation amount on emotion?

From an audio engineering and sound reproduction point of view, how do spectral alterations influence emotion? For example, how does the spectral distortion of playback equipment influence emotion? How about the effect of audio coding on emotion? For example, past work has shown that MP-3 compression with low bit-rates can cause a large reduction in brightness in the saxophone without affecting other instruments [13]. Is the emotional impact changed in a corresponding way? We expect it to be less joyful/happy – is it?

For future work, it will be also fascinating to see how emotion varies with different pitches, dynamic levels, brightness, and articulations. Do these parameters change perceived emotion in a consistent way, or does it vary from instrument to instrument? For example, we know that increased brightness makes a tone more dramatic (more happy or more angry), but is the effect more pronounced in some instruments and less so in others? For example, if the violin, which is the happiest instrument, is played softly with less brightness, is it still happier than the horn, which is the saddest instrument, if the horn is played loudly with maximum brightness? At what point are they equally happy? Can we normalize the instruments to equal happiness by simply adjusting brightness or other attributes? In the same way that we can normalize brightness by filtering or spectral tilting, can we normalize happiness by filtering or spectral tilting (or pitch or dynamic level)? How do the

Ranking \ Emotion	Happy	Sad	Heroic	Scary	Comic	Shy	Joyful	Depressed
1	Vn (5.23)	Hn (8.80)	Tp (3.38)	Fl (2.95)	Cl (3.44)	Hn (8.09)	Cl (7.30)	Hn (8.70)
2	Tp (4.53)	Fl (6.30)	Cl (2.23)	Hn (2.05)	Tp (3.23)	Fl (4.95)	Tp (6.94)	Fl (6.11)
3	Cl (4.42)	Bs (3.75)	Sx (1.93)	Vn (2.02)	Sx (2.84)	Bs (3.85)	Vn (5.35)	Bs (4.08)
4	Sx (3.91)	Ob (2.20)	Vn (1.95)	Tp (1.43)	Ob (2.20)	Ob (2.55)	Sx (5.15)	Ob (2.37)
5	Ob (3.10)	Sx (1.70)	Ob (1.65)	Bs (1.40)	Vn (1.63)	Sx (2.01)	Ob (3.79)	Sx (1.68)
6	Fl (2.02)	Tp (1.47)	Hn (1.05)	Sx (1.34)	Bs (1.37)	Tp (1.53)	Bs (2.17)	Tp (1.64)
7	Bs (2)	Vn (1.08)	Bs (1.03)	Ob (1.28)	Fl (1.30)	Cl (1.49)	Fl (1.84)	Cl (1.20)
8	Hn (1)	Cl (1)	Fl (1)	Cl (1)	Hn(1)	Vn (1)	Hn (1)	Vn (1)

Table 1. Rankings of instruments for each emotion from strongest to weakest with Bradley-Terry-Luce scale values in brackets.

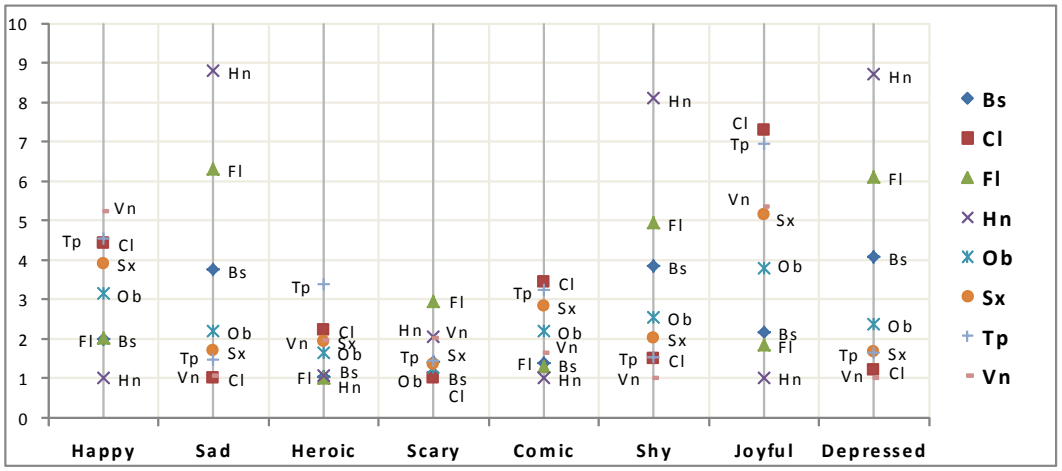


Figure 4. Bradley-Terry-Luce scale values of the instruments for each emotion.

happy spaces of the violin overlap with other instruments in terms of pitch, dynamic level, brightness, and articulation? In general, how does timbre space relate to emotional space?

Emotion gives us a fresh perspective on timbre, helping us to get a handle on its perceived dimensions. It gives us a focus for exploring its many aspects. Just as timbre is a multidimensional perceived space, emotion is an even higher-level multidimensional perceived space deeper inside the listener.

5. ACKNOWLEDGMENTS

This work has been supported by Hong Kong Research Grants Council grants HKUST613111 and HKUST613112.

6. REFERENCES

[1] "happy, sad, heroic, scary, comic, shy, joyful and depressed". *Cambridge Academic Content Dictionary*. Online: http://goo.gl/v5xJZ (17 Feb 2013).

[2] Jean-Julien Aucouturier, François Pachet, and Mark Sandler. The way it sounds: timbre models for analysis and retrieval of music signals. *IEEE Transactions on Multimedia*, 7(6):1028–1035, 2005.

[3] Laura-Lee Balkwill and William Forde Thompson. A cross-cultural investigation of the perception of emotion in music: Psychophysical and cultural cues. *Music perception*, pages 43–64, 1999.

[4] Emmanuel Bigand, Sandrine Vieillard, François Madurell, Jeremy Marozeau, and A Dacquet. Multidimensional scaling of emotional responses to music: The effect of musical expertise and of the duration of the excerpts. *Cognition & Emotion*, 19(8):1113–1139, 2005.

[5] Margaret M Bradley and Peter J Lang. Affective norms for english words (ANEW): Instruction manual and affective ratings. *Psychology*, (C-1):1–45, 1999.

[6] Tuomas Eerola, Rafael Ferrer, and Vinoo Alluri. Timbre and affect dimensions: Evidence from affect and similarity ratings and acoustic correlates of isolated instrument sounds. *Music Perception: An Interdisciplinary Journal*, 30(1):49–70, 2012.

[7] Tuomas Eerola and Jonna K Vuoskoski. A comparison

Features \ Instrument	Bs	Cl	Fl	Hn	Ob	Sx	Tp	Vn
Average Spectral Centroid	3.3806	6.3843	3.4909	2.4229	4.2713	4.1218	4.2561	4.3502
Spectral Flux	3.9274	5.8697	12.659	7.0653	6.1665	7.7108	4.4892	12.187
Spectral Entropy	0.48388	0.53137	0.57382	0.38468	0.44436	0.49527	0.57133	0.54972
Spectral Spread	12.628	14.121	12.518	6.4839	7.4289	11.429	8.1665	10.464
Irregularity	0.17081	1.083	0.44738	0.18566	0.63774	0.85751	0.058612	0.5557
Roughness	0.5379	0.54442	0.017542	0.30809	0.11009	0.69387	0.033896	8.9796

Table 2. Spectral characteristics of the instrument test tones.

Timbre \ Emotion	Happy	Sad	Heoric	Scary	Comic	Shy	Joyful	Depressed
Average Spectral Centroids	**0.8333****	**-0.9048****	**0.6429***	-0.5714	**0.7619****	**-0.8810****	**0.8571****	**-0.8810****
Spectral Flux	0.0714	0.1667	-0.3095	0.5238	-0.381	0.0714	-0.2857	0.0714
Spectral Entropy	0.5714	-0.381	0.1905	0.3095	0.2619	-0.4048	0.4048	-0.4048
Spectral Spread	0.119	-0.3571	-0.0238	-0.3095	0.3095	-0.2619	0.3333	-0.2619
Irregularity	0.2619	-0.4524	0.2381	-0.5952	0.4286	-0.381	0.3571	-0.381
Roughness	0.381	-0.5238	0.3095	-0.3571	0.2381	-0.5714	0.381	-0.5714

Table 3. Spearman correlation between emotions and spectral features. **: $p < 0.05$; * : $0.05 < p < 0.1$.

of the discrete and dimensional models of emotion in music. *Psychology of Music*, 39(1):18–49, 2011.

[8] Cardillo G. Fleiss' kappa: compute the fleiss' kappa for multiple raters. `http://goo.gl/0DFKV`, 2007.

[9] Andrew Horner, James Beauchamp, and Richard So. Detection of random alterations to time-varying musical instrument spectra. *The Journal of the Acoustical Society of America*, 116:1800–1810, 2004.

[10] Yajie Hu, Xiaoou Chen, and Deshun Yang. Lyric-based song emotion detection with affective lexicon and fuzzy clustering method. In *Proceedings of ISMIR*, volume 10, 2009.

[11] David R Hunter. Mm algorithms for generalized bradley-terry models. *Annals of Statistics*, pages 384–406, 2004.

[12] Fleiss L Joseph. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378–382, 1971.

[13] Chung Lee, Andrew Horner, and James Beauchamp. Impact of mp3-compression on timbre space of sustained musical instrument tones. *The Journal of the Acoustical Society of America*, 131(4):3433–3433, 2012.

[14] Stephen McAdams, James W Beauchamp, and Suzanna Meneguzzi. Discrimination of musical instrument sounds resynthesized with simplified spectrotemporal parameters. *The Journal of the Acoustical Society of America*, 105:882, 1999.

[15] Isabelle Peretz, Lise Gagnon, and Bernard Bouchard. Music and emotion: perceptual determinants, immediacy, and isolation after brain damage. *Cognition*, 68(2):111–141, 1998.

[16] James A Russell. Evidence of convergent validity on the dimensions of affect. *Journal of personality and social psychology*, 36(10):1152, 1978.

[17] Klaus R Scherer and James S Oshinsky. Cue utilization in emotion attribution from auditory stimuli. *Motivation and emotion*, 1(4):331–346, 1977.

[18] Janto Skowronek, Martin McKinney, and Steven Van De Par. A demonstrator for automatic music mood estimation. In *Proceedings of the International Conference on Music Information Retrieval, Vienna, Austria*, 2007.

[19] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *Speech and Audio Processing, IEEE transactions on*, 10(5):293–302, 2002.

[20] Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in Neural Information Processing Systems*, 22(2035-2043):7–13, 2009.

[21] Yi-Hsuan Yang, Yu-Ching Lin, Ya-Fan Su, and Homer H. Chen. A regression approach to music emotion recognition. *IEEE TASLP*, 16(2):448–457, 2008.

DESIGN AND EVALUATION OF SEMANTIC MOOD MODELS FOR MUSIC RECOMMENDATION

Mathieu Barthet[1], David Marston[2], Chris Baume[2], György Fazekas[1], Mark Sandler[1]
[1]Centre for Digital Music, Queen Mary University of London
{firstname.lastname}@eecs.qmul.ac.uk
[2]BBC R&D London, {firstname.lastname}@bbc.co.uk

ABSTRACT

In this paper we present and evaluate two semantic music mood models relying on metadata extracted from over 180,000 production music tracks sourced from I Like Music (ILM)'s collection. We performed non-metric multidimensional scaling (MDS) analyses of mood stem dissimilarity matrices (1 to 13 dimensions) and devised five different mood tag summarisation methods to map tracks in the dimensional mood spaces. We then conducted a listening test to assess the ability of the proposed models to match tracks by mood in a recommendation task. The models were compared against a classic audio content-based similarity model relying on Mel Frequency Cepstral Coefficients (MFCCs). The best performance (60% of correct match, on average) was yielded by coupling the five-dimensional MDS model with the term-frequency weighted tag centroid method to map tracks in the mood space.

1. INTRODUCTION

Research on music and emotions is a burgeoning field in the Music Information Retrieval community, generating an ever-increasing number of studies [1]. Although the nature of emotional responses to music is still a misunderstood and controversial topic [6], several analyses of music consumer needs and behaviors have highlighted the usefulness of developing search engines which can organise tracks according to the moods they express (see e.g. [8]). Even before the advent of online music technologies and the concurrent removal of physical media for music such as the CD, production music labels [1] created track compilations according to suggested moods; this music delivery model persists today (see e.g. the "Mood for Love" compilation from West One Music Group [2]). Production music has, by tradition, been called "mood music" as it is often composed to convey mostly constant emotional responses facilitating its association with a narrative, as opposed to commercial music which often presents larger changes of perceived emotions over time. Creative producers searching for background music for television, film, or radio often use keywords which evoke emotions (e.g. "happy music for kids"), or emotion-eliciting situations (e.g. "basketball music"). Unfortunately, due to the complex and subjective nature of music emotions there is no consensual agreement between production music libraries on the genesis and organisation of mood-related metadata. This hinders the recommendation of tracks based on mood, especially on music search engines which aggregate several production music catalogues. In this paper we propose two different methods to represent the relationships between mood tags (semantic mood models) based on analysis of mood-related metadata extracted from 183,176 production music tracks from I Like Music (ILM)'s aggregated catalogue (29 different production music libraries). We then assess how well the various models perform in a task of music recommendation when combined with five different tag summarisation techniques. In a companion study, the mood model providing the best accuracy (a five-dimensional model derived from multidimensional scaling analysis of mood tag co-occurences) was used for audio content-based music emotion recognition using support vector regression (SVR).

2. RELATED WORK

Other data-driven semantic mood models have previously been proposed. In [9], latent semantic analysis (LSA) techniques were applied to 105 emotion words occurring in tags associated with 8,872 Last.fm tracks. The resulting low-dimensional representation obtained after MDS analysis was in line with the traditional arousal and valence dimensions from Russell's circumplex [12], and a third dimension correlated with the *spiritual* or *meditative* component of musical experience. The conclusions from [5] also found mood spaces in agreement with the AV plane after applying MDS to co-occurrence counts of artists from the Last.fm dataset whose styles were described by 146 mood tags. The model from [9] was extended in [14] in which the affective circumplex transform (ACT) was applied to LSA-MDS spaces in order to infer explicit configurations matching Russell's circumplex. The use of ACT provided

[1] Production music is recorded music for use in film, television, radio and other media.

[2] http://www.westonemusic.com/album?catno=WOM_RFM_0018

better accuracy than the baseline LSA-MDS technique according to human judgements. The semantic mood models proposed in this paper (Section 3) differ from previous approaches in several ways. First, to the best of our knowledge, no formal assessment of such models has previously been conducted in relation to music recommendation (Section 4). Second, the models are derived from mood annotations curated by music experts (production music libraries) rather than social tags which are more noisy and idiosyncratic by essence. The analysed mood tags come from the large-scale ILM dataset including a high number of unique mood-related tags (2,060). The comparative study [13] showed that mood models derived from this set of curated editorial mood tags significantly outperformed mood models derived from Last.fm social tags, based on human judgements of arousal, valence and tension. Third, the models which obtained the best performance rely on a higher number of dimensions (5-D and 10-D) than the classic 3-D emotion model.

3. DESIGN OF SEMANTIC MOOD MODELS

3.1 Data-driven Mood Model (MDS)

In the same way that a measure of similarity between tracks can be derived from tag co-occurrence counts, a measure of similarity between tags can be derived from their co-occurrence over tracks [10].

3.1.1 Processing of Mood Tags

In the case of curated editorial metadata, tracks are associated with a list of unique tags judged to be the most appropriate by professional music experts. Hence, a given tag is only attributed once to a track, unlike for social tags for which a large number of users apply tags to tracks. Initially the mood tags were cleaned by correcting misspellings (100 errors out of 2,398 mood tags), removing duplicates (338 duplicates yielding 2,060 unique tags), and stripping white spaces and punctuation marks (e.g. '.', '!'). Instead of following a bag-of-words approach, in which the meaning of certain tags consisting of a series of words can be lost (e.g. "guilty pleasure"), we collated multiple words together to further process them as single entities (using a hyphen between the words). The vocabulary used in the ILM editorial annotations is composed of conventional words and does not suffer from the idiosyncrasies of social tags which often include sentences, informal expressions (e.g. "good for dancing to in a goth bar", cited as an example in [10]), or artists' names. For this reason, we did not have to tokenise the tags with a stop-list (for instance, removing words such as "it", "and", "the"). However, we used a stemmer algorithm [3] to detect tags with similar base parts (e.g. "joyful" and "joy"), as these refer to identical emotional concepts. 1,873 mood-related stems were obtained out of the 2,060 unique mood tags. In order to reduce the size of the stem vector while maintaining the richness of the track descriptions, we only kept stems which were associated with at least 100 tracks in further analyses. This stem filtering process yielded a list of 453 stems which provided at least one mood tag for each of the 183,176 tracks from the ILM dataset. The associations between tracks and stems are provided in a document-term matrix $\mathbf{X} = \{x_{ij}\}$ where:

$$x_{ij} = \begin{cases} 1 & \text{if stem } j \text{ is associated with track } i \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The stem pairwise co-occurrences over tracks c_{ij} are then given by:

$$c_{ij} = |\{x_{\bullet i}\} \cap \{x_{\bullet j}\}| \quad (2)$$

where $\{x_{\bullet i}\}$ is the set of tracks annotated with stem i and $|\ \ |$ is the cardinality operator. The measure of dissimilarity between stems s_{ij} is computed as follows:

$$s_{ij} = 1 - \frac{c_{ij}}{Max(c_{ij})} \quad (3)$$

where $Max(c_{ij})$ is the maximum of the pairwise stem co-occurrence in the ILM dataset (26,859).

3.1.2 Multidimensional Scaling Analysis

Non-metric multidimensional scaling (MDS) analyses were then applied to the stem dissimilarity matrix, $\mathbf{S} = \{s_{ij}\}$. Four outlier stems with a null or very small co-occurrence measure compared to all the other stems were discarded so as not to bias the MDS analysis (this yielded a list of 449 stems). Figure 1 shows the evolution of Kruskal's stress1 [7] as the number of dimensions (D) increases from 1 to 13. Following a rule of thumb for MDS [3], acceptable, good and excellent representations are obtained for D = 3 (stress < 0.2), D = 5 (stress < 0.1) and D = 11 (stress < 0.05). Interestingly, five dimensions yield a good representation (elbow of the scree plot). This result suggests that more than three dimensions are required to accurately categorise mood terms in the context of production music, which contrasts with the classic three-dimensional emotion model (arousal, valence and dominance) [11]. In further analyses, we mapped the mood stems back to mood tags to uncover the meaning of the dimensions. We inspected whether the organisation of the terms along each dimension of the MDS configurations was relevant according to definable emotion-related concepts. For mood tags common to the list from the Affective Norm for English Words (ANEW) [2] (123 common tags were found out of our list of 449), we computed the correlations (Pearson's r) between the coordinates of the mood tags along each dimension of the MDS configurations and the ANEW measures of arousal, valence and dominance (see results in Table 1 in the case where D=5). Interestingly, three out of the five MDS dimensions are significantly correlated with the arousal and/or valence and/or dominance dimensions, showing that the 5-D MDS configuration captures aspects of the core emotion dimensions. However, some of the MDS dimensions are concurrently correlated with the arousal, valence and dominance dimensions. This is

[3] The PorterStemmer algorithm from the Natural Language Toolkit (NLTK) package for Python was used.

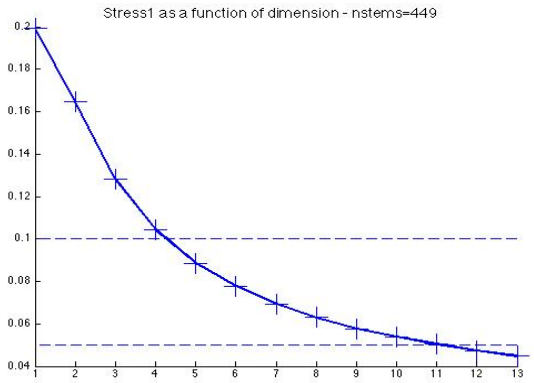


Figure 1. Kruskal's stress1 as a function of the number of dimensions in the MDS analysis of the mood stem dissimilarity matrix.

partly due to the fact that these dimensions can co-vary for certain emotions. For instance, the correlation between the ANEW valence and dominance dimensions is highly significant (r=.83, p<.001) which may explain why the second MDS dimension is correlated with both valence and dominance. A positive, although weak, correlation (r=.20, p<.05) was also found between the ANEW arousal and dominance dimensions. In contrast, no significant correlation was found between the ANEW arousal and valence dimensions. In order to disambiguate some of the MDS dimensions, and as MDS yields a solution which is invariant to rotation, we next applied a transformation to the MDS space to align it according to existing mood models.

3.1.3 Affective Circumplex Transform (ACT)

The affective circumplex transform (ACT) proposed in [14] was applied to the five-dimensional MDS configuration previously described. The goal of the ACT is to match the two first dimensions of the MDS configuration with the dimensions from Russell's arousal/valence (AV) model [12]. As in [13], we first determined mood terms common to both the MDS and Russell's AV spaces (37 terms were found in common). The ACT maintains the relative distances between mood terms in the initial MDS configuration, since it only allows translation, reflection, orthogonal rotation, and isotropic scaling. Measures of correlation between the transformed five-dimensional MDS configuration and the arousal, valence and dominance measures from the ANEW dataset are reported in Table 1. The first dimension of the MDS-ACT configuration is strongly correlated with valence (r=.56, p<.001) whereas before the ACT, no significant correlations with arousal, valence and dominance were found for that dimension. The strong relationship between valence and dominance can explain why the first MDS-ACT dimension is also correlated with dominance (r=.30, p<.001). The second and third dimensions are only correlated with arousal (r=.35, p<.001) and dominance (r=.34, p<.001), respectively. Hence, the ACT can infer an explicit representation according to the core emotion dimensions. Although no clear interpretations of the fourth and fifth dimensions have yet been found, the 5-D model yields a tag configuration which can be used to compute track distance measures, as described in the next two sections.

3.2 Tag Summarisation Methods

We devised several methods to summarise the tags of a track in a given multidimensional mood space. Let's denote $\mathbf{T} = \{t_{ij}\}$ as the tag matrix representing the coordinates of the tags i of a track across the dimensions j of the mood space. For the methods described in Sections 3.2.1 to 3.2.4, the tag summary matrix $\mathbf{Y} = \{y_{ij}\}$ is obtained by multiplying the tag matrix with a weight matrix $\mathbf{W} = \{w_i\}$.

3.2.1 Tag of Maximum Term Frequency (MTF)

This method assumes that a track is best represented by the tag from its set of tags which has the highest term frequency (TF) in the dataset. The weight w_i for the N tags of a track are as follows:

$$w_i = \begin{cases} 1 & if \quad TF(t_i) = Max_{i=(1...N)}[TF(t_i)] \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

3.2.2 Centroid (CEN)

This method summarises the tags of a track by their centroid or geometrical mean in the mood space. The tag weights are hence given by $w_i = \frac{1}{N}$.

3.2.3 Term-Frequency Weighted Centroid (TFW)

This method summarises the tags of a track by their centroid after attributing to each tag a weight proportional to its term frequency (TF): $w_i = \frac{TF(t_i)}{\sum(TF(t_i))}$. Hence the centroid is attracted by the tag with the highest term frequency.

3.2.4 Inverse Term-Frequency Weighted Centroid (ITF)

Conversely, this method attributes more weight to the tag with the lowest term frequency following the assumption that this tag may convey more specific information about the song: $w_i = \frac{1/TF(t_i)}{\sum(1/TF(t_i))}$.

3.2.5 Mean and Variance (MVA)

Rather than summarising the tags of a track by a point in the space, this method assumes that the tags can be represented by a Gaussian distribution. The tag summary matrix $\mathbf{Y}$ is given by the mean $\bar{\mathbf{T}}$ and variance $\sigma(\mathbf{T})$ of the tag matrix: $\mathbf{Y} = \{\bar{\mathbf{T}}; \sigma(\mathbf{T})\}$.

3.3 Model Derived from Mood Taxonomy (CLUST)

Popular mood key words were added to an initial selection provided by ILM to create a list of 355 mood words. Over 95% of the production music library contained at least one of these 355 words. Each of these words were placed in one of 27 categories, which became the starting point for a cluster-based model. Each category was treated as a cluster containing several mood words. Many

	Before ACT			After ACT		
Dimension	Arousal	Valence	Dominance	Arousal	Valence	Dominance
MDS Dim 1	-	-	-	-	.56***	.30***
MDS Dim 2	-	.31***	.36***	.35***	-	-
MDS Dim 3	-.33***	.47***	.19*	-	-	.34***
MDS Dim 4	-	-	-	-	-	-
MDS Dim 5	-.20 *	-.24**	-.31***	-	-	-

Table 1. Correlation (Pearson's r) between the dimensions of the five-dimensional MDS configuration and the arousal, valence and dominance dimensions, as characterised by the ANEW dataset. Only significant correlations are reported. *** p $<$ .001, ** p $<$ .01, * p $<$ .05

of these clusters could be considered to overlap in their mood; some were clearly opposites while others had little in common. To convert these clusters into dimensions, the overlapping ones were combined into single dimensions; any opposite clusters were converted into negative (-ve) values of the dimension they were opposite to; and the non-overlapping clusters were treated as new dimensions. Using this method, the 27 clusters were converted to 10 dimensions, giving each of the 355 mood words 10 dimensional mood values. The choice of allocation of words to clusters and clusters to dimensions was performed based on only one person's opinion. The choice of 10 dimensions was a compromise between combining clusters that were too dissimilar and having too sparse a model. To illustrate the process, the first three dimensions represent the following mood clusters: 1) Confident (+ve scale), Cautious & Doubtful (-ve scale); 2) Sad & Serious (+ve scale), Happy & Optimistic (-ve scale); and 3) Exciting (+ve scale), Calm (-ve scale).

As each music track is associated with several mood tags which each mapped to 10 dimensional values, tags had to be combined. The most simple and obvious way would be to take the mean of all the mood values to generate a single 10-dimensional value for a track. However, it was felt that a music track can be represented by moods that differ significantly, so combining them into a single mood would be too crude. Therefore, a method (denoted PEA) to generate two mood values per track was devised. This method uses clustering of the 10-D scores where close scores are combined together. The means of the two most significant clusters are then calculated, resulting in two 10-D mood values for each track. A weight was assigned to each value according to the size of the cluster.

3.4 Track Distance Measures in Mood Space

For the purposes of searching a database of tracks with mood values assigned to them, a distance measurement is required to find which tracks most closely match each other. For the MDS-based models (Section 3.1), distances between tracks were obtained using either the Euclidean distance between tag summary vectors (methods MTF, CEN, TFW, ITF), or the Kullback-Leibler (KL) divergence between the Gaussian representations of the tags (method MVA). As the model described in Section 3.3 allocates two 10-D mood values per track (method PEA), a weighted Euclidean measure was used which exploited the weighting values associated with each of the two 10-D mood values. This is shown in Equation (5) where $m_s(i,k)$ is the mood of the seed track (where i is the value index, and k is the dimension index), $m_t(j,k)$ is the mood expressed by the test track (where j is the value index), $w_s(i)$ is the seed track weighting, and $w_t(j)$ is the test track weighting.

$$d = \sqrt{\sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{k=0}^{9}((w_s(i)\cdot m_s(i,k)) - (w_t(j)\cdot m_t(j,k)))^2} \quad (5)$$

4. LISTENING EXPERIMENT

4.1 Corpus and Recommenders Tested

5,000 production music tracks were picked up randomly from the ILM dataset according to two constraints: (i) the durations of the tracks had to be at least 60 s (in order to discard short versions of the tracks), (ii) instrumental stems, i.e. individual tracks from multitrack recordings, were discarded. Six main genres were represented (jazz, dance, rock, electronic, folk and orchestral). 18 different recommenders were tested based on the two different mood models MDS and CLUST (Sections 3.1 and 3.3, respectively), the ACT transformation of the MDS model, MDS-ACT (Section 3.1.3), and a timbre-based model based on 20 MFFCs. Model MDS was tested with three different dimensions (3, 5 and 11) and the five different tag summarisation methods defined in Section 3.2 (MTF, CEN, TFW, ITF, MVA). Models CLUST, MFCC and MDS-ACT were tested with just one configuration each. As the ACT maintains the relative distances between mood terms in the MDS space, it doesn't affect track distances using the MTF, CEN, TFW, and ITF methods. The MDS-ACT model was hence only tested with the MVA method.

4.2 Procedure

To determine which mood model configuration used in a recommender gave the best matches according to human perception, a listening experiment was conducted. If a mood model is of any use it should ensure that a recommender generates tracks that closely match a seed track according to mood.

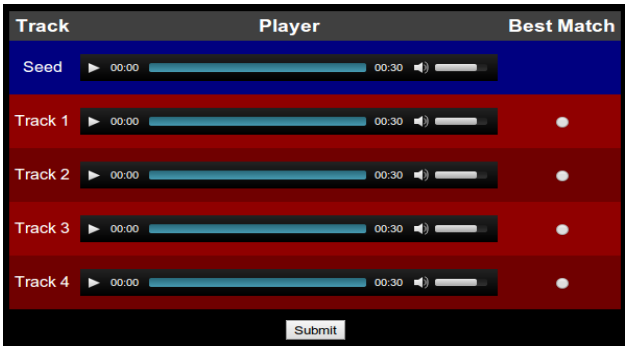


Figure 2. Survey interface.

The requirements of the test were that it should be simple to perform, not require specialist software or equipment, be accessible to enough people and not take too much time. To achieve these aims, a simple web-page was developed which made use of the audio tools in HTML5 (see Figure 2), so anyone could access the survey on an internet connection. The web-page presented the listener with a randomly selected seed track (from the 5,000 track dataset), plus four more tracks for assessment. In order to avoid potential causes of similarity between tracks due to the fact that they belong to the same album, recommended tracks were restricted to belong to a different album than that of the seed track. Of the four assessed tracks, one was from the recommender and the other three were randomly selected. The choice of recommender was also random. Participants were required to listen to each track at least once and select which one of the four tracks they felt most closely matched the seed track. They could repeat the process as many times as they wished, though they were encouraged to do at least 10 minutes' worth of testing.

4.3 Statistical Analysis

The system counted how many times the participants selected the recommenders' tracks. Therefore if a recommender generated recommendations no better than chance, the baseline score would be 25%. To determine which model gave the best performance, the confidence intervals for the scores had to demonstrate whether recommenders' scores were significantly different from each other. The score $s(m)$ for a particular recommender m is shown in the Equation 6, where $r(m,n)$ is 1 for a correct selection of the recommender m, 0 for a false selection, and n is the trial index. The number of times the recommender has been tested is $N(m)$.

$$s(m) = \frac{1}{N(m)} \sum_{n=1}^{N(m)} r(m,n) \qquad (6)$$

To calculate the confidence intervals, we decided to move away from the traditional parametric method which assumes a Gaussian distribution, and use a non-parametric bootstrapping method [4].

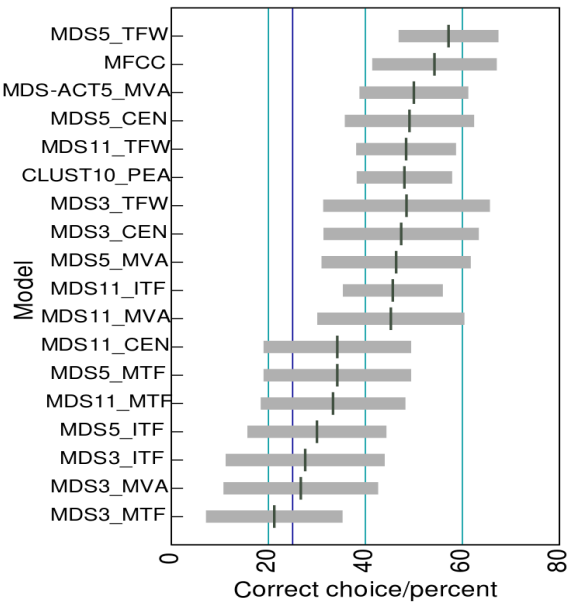


Figure 3. % of correct choice across recommenders. The recommenders are labelled using abbreviations in the format: 'ModelDimensions_TagSummarisationMethod'.

5. RESULTS AND DISCUSSION

The simple web-based survey design meant that the survey was easily accessible and simple to use, which allowed between 40-60 participants (estimated from email request list and activity) to perform the survey. There were 971 marks given in total in the survey, which was enough to determine which recommenders we should avoid using, but not enough to identify a clear leader. The scores for each model are shown in Figure 3, and are arranged in order with the best-scoring model at the top. The horizontal bars represent the confidence interval, with the fine vertical line representing the mean score. There is a line at 25% which corresponds to the random baseline score, so any scores that overlap that line can be considered to be no better than chance (given the sample size). To compare the performance of the different dimensions and of the MDS model, the scores from the versions of the model with the same dimensions were combined. To compare the methods, the scores from the same method were combined. Figure 4 shows how 3, 5 and 11 dimensions compare, and how the five tag summarisation methods perform. While the number of results per recommender ranged from 29 to 91 scores, the confidence intervals still remained quite large for making comparisons. While it would have been preferable to sample more participants to reduce the confidence intervals, the length of the experience had to be considered. While the confidence intervals do not necessarily allow a single recommender to be selected above all the others, they do still give us enough scope to eliminate the worst recommenders. The results show that the seven worst recommenders' confidence intervals overlap the 25% baseline score, although with some of those the overlap is small and a larger sample size could show they

are better than chance. For assessing the types of model, number of dimensions, method and distance measures, there was little evidence to conclude that any of those factors individually had a strong influence. However, the scores indicate the recommenders (for model MDS) with 5 dimensions could be stronger than 3 and 11 dimensions (though confidence intervals overlap). The MTF (tag with maximum term frequency) method yielded the smallest scores, which were significantly smaller than those of the strongest method, TFW (term-frequency weighted centroid).

The only recommender which does not use metadata relies on the timbre-based similarity measurement (MFCC). This recommender performed well, which can probably be explained by the fact that within the 5,000 tracks in the survey, there were still enough tracks that sounded very similar to each other (despite a certain amount of manual filtering of the full track list), and which would therefore indicate a similar mood. In practice, with a recommender searching over a million tracks, one that just returns very similar-sounding tracks may not fit the requirement of a mood-based recommender offering a more diverse selection.

Model CLUST performed as well as the best combinations of models MDS and MFCC. However, it was only tested with a single configuration, so it was not known whether the 10-dimensional two-maxima method was the best for it. It was also based on only one person's opinions of mood words, so could have benefited from a more extensive multi-person survey to refine the mood values.

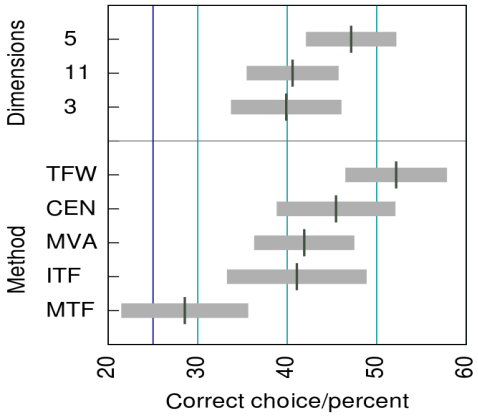


Figure 4. Dimension and method scores of model MDS.

6. SUMMARY AND CONCLUSIONS

The aim of the work was to design a mood model suitable for a recommender that could work over a very large database of music. It was felt that existing mood models with two [12] or three dimensions [11] were both too generalised for music analysis and lacking in dimensions to discriminate over large quantities of music. The survey conducted for this study assessed how various recommenders corresponded with human perception of music track matching based on perceived moods. The survey demonstrated that higher dimension models are effective in a recommender, and given the correct choice of summarisation methods and distance measures, can be tuned for better results. The results of the test showed that a five-dimensional model produced the best scores, although it was not statistically the clear leader. This work used production music associated with manually generated mood tags, and thus provided a source of information on emotional responses to music without analysis of the audio signal itself. Based on the results of the perceptual evaluation presented in this paper, a mood model (5-D MDS with term-frequency weighted centroid) was selected to develop an audio-content-based music emotion recognition (MER) system. We will follow up this study by testing to what extent this system can be used to assign mood tags in a robust manner to commercial music tracks which do not possess mood metadata.

7. ACKNOWLEDGEMENTS

This work was partly funded by the TSB project "Making Musical Mood Metadata" (TS/J002283/1). The authors would like to acknowledge the BBC Audio Research Partnership from which this collaboration was started.

8. REFERENCES

[1] M. Barthet, G. Fazekas, and M. Sandler. Multidisciplinary perspectives on music emotion recognition: Recommendations for content- and context-based models. In *Proc. of CMMR*, pages 492–507, 2012.

[2] M.M. Bradley and P.J. Lang. Affective norms for English words (ANEW): Instruction manual and affective ratings. Technical report c-2, University of Florida, Gainesville, FL, 2010.

[3] W. R. Dillon and M. Goldstein. *Multivariate Analysis*. Wiley series in probability and mathematical statistics. John Wiley & Sons, New York, 1984.

[4] T. C. Hesterberg, D. S. Moore, S. Monaghan, A. Clipson, and R. Epstein. *Bootstrap methods and permutation tests*. 2005.

[5] X. Hu. Music and mood: where theory and reality meet. In *Proc. of iConference*, 2010.

[6] P. Juslin and D. Västfäll. Emotional responses to music: The need to consider underlying mechanisms. *Behavioral and brain sciences*, 31:559–621, 2008.

[7] J.B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1–27, 1964.

[8] J. A. Lee and J. S. Downie. Survey of music information needs, uses, and seeking behaviors: preliminary findings. In *Proc. of ISMIR*, 2004.

[9] M. Levy. *Retrieval and Annotation of Music Using Latent Semantic Models*. PhD thesis, Queen Mary University of London, 2012.

[10] M. Levy and M. Sandler. A semantic space for music derived from social tags. In *Proc. ISMIR*, pages 411–416, 2007.

[11] C. E. Osgood, G. J. Suci, and P. H. Tannenbaum. *The Measurement of Meaning*. University of Illinois Press, 1957.

[12] J.A. Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161–1178, 1980.

[13] P. Saari, M. Barthet, G. Fazekas, T. Eerola, and M. Sandler. Semantic models of musical mood: comparison between crowd-sourced and curated editorial tags. In *Proc. of IEEE ICME (Affective Analysis in Multimedia workshop)*, San Jose, CA, 2013.

[14] P. Saari and T. Eerola. Semantic computing of moods based on tags in social media of music. *IEEE Transactions on Knowledge and Data Engineering*, manuscript submitted for publication available at http://arxiv.org/, 2013.

LOW-RANK REPRESENTATION OF BOTH SINGING VOICE AND MUSIC ACCOMPANIMENT VIA LEARNED DICTIONARIES

Yi-Hsuan Yang
Research Center for IT Innovation, Academia Sinica, Taiwan
yang@citi.sinica.edu.tw

ABSTRACT

Recent research work has shown that the magnitude spectrogram of a song can be considered as a superposition of a low-rank component and a sparse component, which appear to correspond to the instrumental part and the vocal part of the song, respectively. Based on this observation, one can separate singing voice from the background music. However, the quality of such separation might be limited, because the vocal part of a song can sometimes be low-rank as well. Therefore, we propose to learn the subspace structures of vocal and instrumental sounds from a collection of clean signals first, and then compute the low-rank representations of *both* the vocal and instrumental parts of a song based on the learned subspaces. Specifically, we use online dictionary learning to learn the subspaces, and propose a new algorithm called multiple low-rank representation (MLRR) to decompose a magnitude spectrogram into two low-rank matrices. Our approach is flexible in that the subspaces of singing voice and music accompaniment are both learned from data. Evaluation on the MIR-1K dataset shows that the approach improves the source-to-distortion ratio (SDR) and the source-to-interference ratio (SIR), but not the source-to-artifact ratio (SAR).

1. INTRODUCTION

A musical piece is usually composed of multiple layers of voices sounded simultaneously, such as human vocal, melody line, bass line and percussion. These components are mixed in most songs sold in the market. For many music information retrieval (MIR) problems, such as predominant instrument recognition, artist identification and lyrics alignment, separating one source from the others is usually an important pre-processing step [6, 9, 13].

Many algorithms have been proposed for blind source separation in monaural music signals [21, 22]. For the particular case of separating singing voice from music accompaniment, it has been found that characterizing the music accompaniment as a repeating structure on which varying vocals are superimposed leads to good separation quality [8, 16, 17, 23]. For example, Huang *et al.* [8] found that, by decomposing the magnitude spectrogram of a song into a low-rank matrix and a sparse matrix, the sparse component appears to correspond to the singing voice. Evaluation on the MIR-1K data set [7] shows that such a low-rank decomposition (LRD) method outperforms sophisticated, pitch-based inference methods [7, 22].

However, the low-rank and sparsity assumptions about the music accompaniment and singing voice have not been carefully studied so far. From mathematical point of view, the low-rank component corresponds to a succinct representation of the observed data in a lower dimensional subspace, whereas the sparse component corresponds to the (small) fraction of the data samples that are far away from the subspace [2, 11]. Without any prior knowledge of the data, it is not easy to distinguish between data samples originated from the subspace of music accompaniment and those from the subspace of singing voice. Therefore, the low-rank matrix resulting from the aforementioned decomposition might be actually a mixture of the subspaces of vocal and instrumental sounds, and the sparse matrix might contain a portion of the instrumental sounds such as the main melody or the percussion sounds [23].

Because MIR-1K comes with "clean" vocal and instrumental sources recorded separately at the left and right channels, in our pilot study we tried LRD using principal component analysis (PCA) [2] for the two clean sources, respectively. Result shows that, contrary to the sparsity assumption, the vocal channel can also be well approximated by a low-rank matrix. As Figure 1 exemplifies, we are able to reduce the rank of the singing voice and the music accompaniment matrices (by PCA) from 513 to 50 and 10, respectively, with less than 40% loss in the source-to-distortion ratio (SDR) [20].

Motivated by the above observation, in this paper we investigate the quality of separation as a result of decomposing the magnitude spectrogram of a song into "two" low-rank matrices plus one sparse matrix. The first two matrices represent the singing voice and music accompaniment in the subspaces of vocal and instrumental sounds, respectively, whereas the last matrix contains data samples deviated from the subspaces. Therefore, unlike existing methods, the vocal part of a song is also modeled as a low-rank signal. Moreover, different subspaces are explicitly used for vocal and instrumental sounds.

To achieve the above decomposition, we propose a new algorithm called multiple low-rank representation (MLRR),

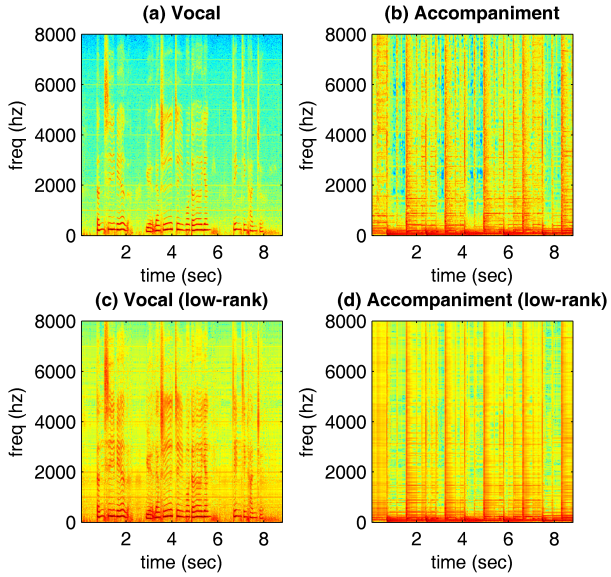


Figure 1. (a) (b) The original, full-rank magnitude spectrograms (in log scale) of the vocal and instrumental parts of the clip 'Ani_1_01' in MIR-1K [7]. (c) (d) The low-rank matrices of the vocal part (rank=50) and the instrumental part (rank=10) obtained by PCA. Such low-rank approximation only incurs 40% loss in signal-to-distortion ratio.

which involves an iterative optimization process that seeks the lowest rank representation [2, 10, 11]. Moreover, instead of decomposing a signal from scratch, we employ an online dictionary learning algorithm [12] to learn the subspace structures of the vocal and instrumental sounds in advance from an external collection of clean vocal and instrumental signals. In this way, we are able to incorporate prior knowledge about the nature of vocal and instrumental sounds to the decomposition process.

The paper is organized follows. Section 2 reviews related work on LRD its application to singing voice separation. Section 3 describes the proposed algorithms. Section 4 presents the evaluation and Section 5 concludes.

2. REVIEW ON LOW-RANK DECOMPOSITION

It has been shown that many real-world data can be well characterized by low-dimensional subspaces [11]. That is, if we put n m-dimensional data vectors in the form of a matrix $X \in \Re^{m \times n}$, X should have rank $r \ll \min(m, n)$, meaning few linearly independent columns [2]. The goal of LRD is to obtain a low-rank approximation of X in the presence of outliers, noises, or missing values [11].

The classical principal component analysis (PCA) [2] seeks a rank-r estimate A of the matrix X by solving

$$\begin{aligned} \min_{A} \quad & \|X - A\| \\ \text{subject to} \quad & \text{rank}(A) \leq r \,, \end{aligned} \tag{1}$$

where $\|X\|$ denotes the spectral norm, or the largest singular value of X. This problem can be efficiently solved via singular value decomposition (SVD) by using the r largest singular values [2].

It is well-known that PCA is sensitive to outliers. To remedy this issue, robust PCA (RPCA) [2] uses the l_1 norm to characterize sparse corruptions and solves

$$\min_{A} \quad \|A\|_* + \lambda \|X - A\|_1 \,, \tag{2}$$

where $\|\cdot\|_*$ denotes the nuclear norm (the sum of its singular values), $\|\cdot\|_1$ is the l_1 norm that sums the absolute values of matrix entries, and λ is a positive weighting parameter. The use of nuclear norm as a surrogate of the rank function makes it possible to solve (2) by convex optimization algorithms such as accelerated proximal gradient (APG) or augmented Lagrange multipliers (ALM) [10].

RPCA has been successfully applied to singing voice separation [8]. Researchers found that the resulting sparse component (i.e., $X - A$) appears to correspond to the vocal part and the low-rank one (i.e., A) corresponds to the music accompaniment. More recently, Yang [23] found that the sparse component often contains percussion sounds and proposed a back-end drum removal procedure to enhance the quality of the separated singing voice. Sprechmann *et al.* [17] considered both A and $X - A$ to be non-negative and employed multiplicative algorithms to solve the resulting robust non-negative matrix factorization (RNMF) problem. Efficient, supervised or semi-supervised variants have also been proposed [17]. Although promising result is obtained, none of the reviewed methods justified the assumption of considering singing voice as sparse.

Durrieu *et al.* [3] proposed a non-negative matrix factorization (NMF)-based method for singing voice separation that regards the vocal spectrogram as an element-wise multiplication of an excitation spectrogram and a filter spectrogram. Many other NMF-based methods that do not rely on the sparse assumption have also been proposed [14]. However, we tend to focus on LRD-based methods that have similar form as RPCA in this work. The comparison with NMF-based methods is left as a future work.

Finally, low-rank representation (LRR) [11] seeks the lowest rank estimate of data X with respect to $D \in \Re^{m \times k}$, a "dictionary" that is assumed to linearly span the space of the data being analyzed. Specifically, it solves

$$\min_{Z} \quad \|Z\|_* + \lambda \|X - DZ\|_1 \,, \tag{3}$$

where $Z \in \Re^{k \times n}$ and k denotes the dictionary size. Since $\text{rank}(DZ) \leq \text{rank}(Z)$, DZ is also a low-rank recovery to X. As discussed in [11], by properly choosing D, LRR can recover data drawn from a mixture of several low-rank subspaces. By setting $D = I_m$, the $m \times m$ identify matrix, the formulation (3) reduces to (2). Although it is possible to use dictionary learning algorithms such as K-SVD [1] to learn a dictionary from data, Liu *et al.* [11] simply set $D = X$, using the data matrix itself as the dictionary. In contrast, we extend LRR to the case of multiple dictionaries and employ online dictionary learning (ODL) [12] to learn the dictionaries, as described below.

3. PROPOSED ALGORITHMS

By extending formulation (3), we are able to obtain the low-rank representations of X with respect to multiple dic-

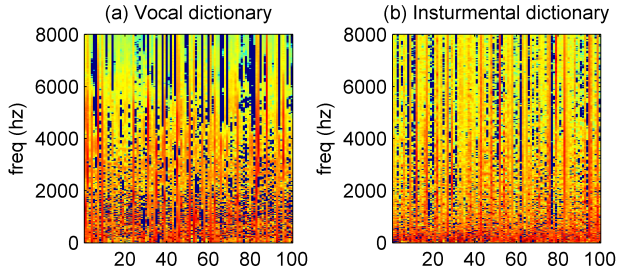


Figure 2. The spectra (in log scale) of the learned dictionaries (with 100 codewords) for (a) vocal and (b) instrumental spectra, using online dictionary learning.

tionaries $D_1, D_2, \ldots, D_\kappa$, where κ denotes the number of dictionaries. Although it is possible to use a dictionary for each musical component (e.g., human vocal, melody line, bass line and percussion), we consider the case $\kappa = 2$ and use one dictionary for human vocal and the other for the music accompaniment.

3.1 Multiple Low-Rank Representation (MLRR)

Given an input data X and two pre-defined (or pre-learned) dictionaries $D_1 \in \Re^{m\times k_1}$ and $D_2 \in \Re^{m\times k_2}$ (k_1 and k_2 can take different values), MLRR seeks the lowest rank matrices Z_1 and Z_2 by solving

$$\min_{Z_1,Z_2} \|Z_1\|_* + \beta \|Z_2\|_* + \lambda \|X - D_1Z_1 - D_2Z_2\|_1 , \quad (4)$$

where β is a positive parameter. This optimization problem can be solved by the method of ALM [10], by first reformulating (4) as

$$\begin{aligned} \min_{Z_1,Z_2,J_1,J_2,E} \quad & \|J_1\|_* + \beta \|J_2\|_* + \lambda \|E\|_1 \\ \text{subject to} \quad & X = D_1Z_1 + D_2Z_2 + E , \\ & Z_1 = J_1 , \; Z_2 = J_2 , \end{aligned} \quad (5)$$

and then minimizing the augmented Lagrangian function

$$\begin{aligned} \mathcal{L} = \; & \|J_1\|_* + \text{tr}(Y_1^T(Z_1 - J_1)) + \frac{\mu}{2}\|Z_1 - J_1\|_F^2 \\ + \; & \beta \|J_2\|_* + \text{tr}(Y_2^T(Z_2 - J_2)) + \frac{\mu}{2}\|Z_2 - J_2\|_F^2 \\ + \; & \lambda \|E\|_1 + \text{tr}(Y_3^T(X - D_1Z_1 - D_2Z_2 - E)) \\ + \; & \frac{\mu}{2}\|X - D_1Z_1 - D_2Z_2 - E\|_F^2 , \end{aligned} \quad (6)$$

where $\|\cdot\|_F$ denotes the Frobenius norm (square root of the sum of the squares of its elements) and μ is a positive penalty parameter. We can minimize (6) with respect to Z_1, Z_2, J_1, J_2, E, respectively, by fixing the other variables and then updating the Lagrangian multipliers Y_1, Y_2 and Y_3. For example, J_2 can be updated by

$$J_2^* = \text{argmin}\, \beta \|J_2\|_* + \frac{\mu}{2}\|J_2 - (Z_2 + \mu^{-1}Y_2)\|_F^2 , \quad (7)$$

which can be solved via the singular value thresholding (SVT) operator [2], whereas Z_1 can be updated by

$$Z_1^* = \Sigma_1 \left(D_1^T(X - D_2Z_2 - E) + J_1 + \mu^{-1}(D_1^TY_3 - Y_1)\right) , \quad (8)$$

where $\Sigma_1 = (I + D_1^TD_1)^{-1}$. The update rule for the other variables can be obtained in a similar way as described in [10, 11], mainly by taking the first-order derivative of the augmented Lagrangian function $\mathcal{L}$ with respect to the variable. By using a non-decreasing sequence of $\{\mu_t\}$ as suggested in [10] (i.e., using μ_t in the t-th iteration), empirically we observe that the optimization usually converges in 100 iterations. After the decomposition, we consider D_1Z_1 and D_2Z_2 as the vocal and instrumental parts of the song and discard the intermediate matrices E, J_1 and J_2.

3.2 Learning the Subspace Structures of Singing and Instrumental Sounds

The goal of dictionary learning is to find a proper representation of data by means of reduced dimensionality subspaces, which are adaptive to both the characteristics of the observed signals and the processing task at hand [19]. Many dictionary learning algorithms have been proposed, such as $k$means and K-SVD [1, 19]. In this work, we adopt the online dictionary learning (ODL) [12], a first-order stochastic gradient descent algorithm, for its low memory consumption and computational cost. ODL has been used in many MIR tasks such as genre classification [24].

Given N signals $p_i \in \Re^m$, ODL learns a dictionary D by solving the following joint optimization problem,

$$\begin{aligned} \min_{D,Q} \quad & \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{2}\|p_i - Dq_i\|_2^2 + \eta\|q_i\|_1\right) , \\ \text{subject to} \quad & d_j^Td_j \le 1 , q_i \ge 0 , \end{aligned} \quad (9)$$

where $\|\cdot\|_2$ denotes the Euclidean norm for vectors, Q denotes the collection of the (unknown) nonnegative encoding coefficients $q_i \in \Re^k$, and η is a regularization parameter. The dictionary D is composed of k codewords $d_j \in \Re^m$, whose energy is limited to be less than one. Formulation (9) can be solved by updating D and Q in an alternating fashion. The optimization of q_i involves a typical sparse coding problem that can be solved by the LARS-lasso algorithm [4]. Our implementation of ODL is based on the SPAMS toolbox [12]. [1]

Figure 2 shows the dictionaries for vocal and instrumental spectra we learned from a subset of MIR-1K, using $k_1 = k_2 = 100$. It can be found that the vocal dictionary contains voices of higher fundamental frequency. In addition, we see more energy in the so-called "singer's formant" (around 3 khz) from the vocal dictionary [18], showing that the two dictionaries capture distinct characteristics of the signals. Finally, we also observe some atoms that span almost the whole spectra in both dictionaries (e.g., the 12th codeword in the instrumental dictionary), possibly because of the need to reconstruct a signal by a sparse subset of the dictionary atoms, by virtue of the l_1-based sparsity constraint in formulation (9).

In principle, we can improve the reconstruction accuracy (i.e., smaller $\|p_i - Dq_i\|_2$ in (9)) by using larger k [12], at the expense of increasing the computational cost in solving both (9) and (5). However, as Section 4.1 shows, larger

[1] http://spams-devel.gforge.inria.fr/

k does not necessarily lead to better separation quality, possibly because of the mismatch between the goals of reconstruction and of separation.

The source codes, sound examples, and more details of this work are available online. [2]

4. EVALUATION

Our evaluation is based on the MIR-1K dataset collected by Hsu & Jang [7]. [3] It contains 1,000 song clips extracted from 110 Chinese pop songs released in karaoke format, which consists of a clean music accompaniment track and a mixture track. A total number of eight female and 11 male amateur singers were invited to sing the songs, thereby creating the clean singing voice track for each clip. Each clip is 4 to 13 seconds in length and sampled at 16 khz. Although MIR-1K also comes with human-labeled pitch values, unvoiced sounds and vocal/nonvocal segments, lyrics, and the speech recordings of the lyrics for each clip [7], these information are not exploited in this work.

Following [17], we reserved 175 clips sang by one male and one female singers ('abjones' and 'amy') for training (i.e., learning the dictionaries D_1 and D_2), and used the remaining 825 clips of 17 singers for testing the performance of separation. For the test clips, we mixed the two sources v and a linearly with equal energy (i.e., 0 db signal-to-noise ratio) to generate x, the mixture of sounds similar to the one available from commercial CDs. The goal is to recover v and a from x for each test clip separately.

Given a music clip, we first computed its short-time Fourier transform (STFT) by sliding a Hamming window of 1024 samples and 1/4 overlapping (as in [8]) to obtain the spectrogram, which consists of the magnitude part X and the phase part P. We applied matrix decomposition using X to get the separated sources. To synthesize the time-domain waveforms $\hat{v}$ and $\hat{a}$, we performed inverse STFT using the magnitude spectrogram of the separated source and the phase P of the original signal [5]. Because the separated spectrogram may contain negative values, we converted negative values to zero before inverse STFT.

The quality of separation is assessed in terms of the following measures [20], which are computed for the vocal part v and the instrumental part a, respectively,

- Source-to-distortion ratio (SDR), which measures the energy ratio between the source and the distortion (e.g., v to $v - \hat{v}$).
- Source-to-artifact ratio (SAR), which measures the amount of artifacts of the source separation algorithm such as musical noise.
- Source-to-interference ratio (SIR), which measures the interference from other sources.

Higher values of these ratios indicate better separation quality. We computed these ratios by using the BSS Eval toolbox v3.0, [4] assuming that the admissible distortion is a time-invariant filter [20]. As in [7], we compute the *normalized* SDR (NSDR) by $\text{SDR}(\hat{v}, v) - \text{SDR}(x, v)$. Moreover, we aggregate the performance over all the test clips by taking the weighted average, with weight proportional to the length of each clip [7]. The resulting measures are denoted as GNSDR, GSAR, and GSIR, respectively (the later two are not normalized). [5]

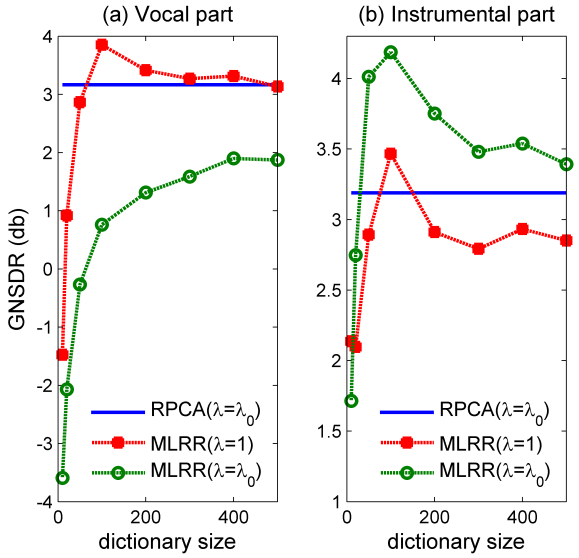


Figure 3. The quality of the separated (a) vocal and (b) instrumental parts of the 825 clips in MIR-1K in terms of global normalized source-to-distortion ratio (GNSDR).

4.1 Result

We first compared the performance of MLRR with RPCA, one of the state-of-the-art algorithms for singing voice separation [8]. We used ALM-based algorithm for both MLRR and RPCA [10]. For MLRR, we learned dictionaries from the training set and evaluate separation on the test set of MIR-1K. Although it is interested to use different dictionary sizes for the vocal and instrumental dictionaries, we set $k_1 = k_2 = k$ in this study. For RPCA, we simply evaluated it on the test set, without using the training set. The value of λ was set to either $\lambda_0 = 1/\sqrt{\max(m, n)}$, according to [2] (recall that (m, n) is the size of the input matrix X), or 1, as suggested in [11]. We only use λ_0 for RPCA because using 1 did not work. Moreover, we simply set β to 1 for MLRR. For future work it would be interesting to use different β to investigate whether we want to penalize the rank of one particular source more. [6]

Figure 3 shows the quality (in terms of GNSDR) of the separated vocal and instrumental parts using different algorithms, different values of the parameter λ and different values of the dictionary size k. We found that MLRR attains the best result when $k = 100$ for both parts (3.85 db and 4.19 db). The performance difference in GNSDR be-

[2] http://mac.citi.sinica.edu.tw/mlrr

[3] https://sites.google.com/site/unvoicedsoundseparation/

[4] http://bass-db.gforge.inria.fr/

[5] Please note that in some previous work the older version BSS Eval toolbox v2.1 was used [7, 8, 23], assuming that the admissible distortion is purely a time-invariant gain.

[6] In fact, when $\beta = 1$ one can combine Z_1 and Z_2, reducing (4) to (3), and used an LRR-based algorithm to solve the problem as well.

Table 1. Separation quality (in db) for the singing voice

Method	GNSDR	GSIR	GSAR
RPCA (λ=λ_0) [8]	3.17	4.43	11.1
RPCAh (λ=λ_0) [23]	3.25	4.52	11.1
RPCAh+FASST [23]	3.84	6.22	9.19
MLRR (k=100, λ=1)	3.85	5.63	10.7

Table 2. Separation quality for the music accompaniment

Method	GNSDR	GSIR	GSAR
RPCA (λ=λ_0) [8]	3.19	5.24	9.23
RPCAh (λ=λ_0) [23]	3.27	5.31	9.30
RPCAh+FASST [23]	3.21	5.24	9.30
MLRR (k=100, λ=λ_0)	4.19	7.80	8.22

tween MLRR (when $k = 100$) and RPCA is significant, either for the vocal or instrumental part, under one-tailed t-test (p-value$<$0.001; d.f.=1648). [7]

From Figure 3, several observations can be made. First, it can be found that using larger k does not always lead to better performance, as discussed in Section 3.2. Second, for the instrumental part, using $k = 20$ ($\lambda = \lambda_0$) already yields high GNSDR (2.74 db), whereas for the vocal part we need to use at least $k = 50$ ($\lambda = 1$). This result shows that we need more dictionary atoms to represent the space of the singing voice, possibly because the subspace of singing voice is of higher rank (cf. Figure 1). The separation quality of the singing voice is worse (i.e., lower than zero) when k is too small. Third, we saw that the vocal and instrumental parts favor different values of λ for MLRR, which deserves future study. [8]

Next, we compared MLRR with the two algorithms presented in [23], in terms of more performance measures. RPCAh is an APG-based algorithm that uses harmonicity priors to take into account the similarity between sinusoidal elements [23]; RPCAh+FASST employs Flexible Audio Source Separation Toolbox for removing the drum sounds in the vocal part [15]. Because FASST involves a heavy computational process, we set the maximal number of iterations to 100 in this evaluation. [9]

Result shown in Tables 1 and 2 indicates that, except for the GSIR for singing voice, MLRR outperforms all the evaluated RPCA-based methods [8,23] in terms of GNSDR and GSIR, especially for the music accompaniment. However, we also found that MLRR introduces some artifacts and leads to slightly lower GSAR. This is possibly because the separated sounds are linear combination of the dictionary atoms, which may not be comprehensive enough to capture every nuance of music signals.

Finally, to provide a visual comparison, Figure 4 shows the separation result for RCA (λ=λ_0), RCAh+FASST, and MLRR (k=100, λ=1) for the clip 'Ani_1_01,' focusing on low frequency parts 0–4 khz. We saw that the recovered vocal signal well captures the main vocal melody, and that components with strong harmonic structure are present in the recovered instrumental part. We also observed undesirable artifacts in the higher frequency components of MLRR, which should be the subject of future research.

[7] We have tried imposing a nonnegative constraint on the dictionary D (c.f. Eq. 9) but this did not further improve the result.

[8] It is fair to use different λ for the two sources; for example, if the application is about analyzing singing voice, one can use λ=1.

[9] We did not compare our result with another two state-of-the-art methods [17] and [16], because somehow we cannot reproduce the result for the former and because the latter did not evaluate on MIR-1K. Moreover, please note that the evaluation here is performed on 825 clips (excluding those used for dictionary learning) instead of the whole MIR-1K.

5. CONCLUSION AND DISCUSSION

In this paper, we have presented a time-frequency based source separation algorithm for music signals that considers both the vocal and instrumental spectrograms as low-rank matrices. The technical contributions we have brought to the field include the use of dictionary learning algorithms to estimate the subspace structures of music sources and the development of a novel algorithm MLRR that uses the learned dictionaries for decomposition. The proposed method is advantageous in that potentially more training data can be harvested to improve the result of separation. Although it might not be fair to directly compare the performance of MLRR and RPCA (because the former uses an external dictionary), our result shows that we can still get similar separation quality without the sparse assumption on the singing voice. However, because the separated sounds are linear combination of the atoms in the pre-learned dictionaries, there are some unwanted artifacts that are audible, which should be the subject of future work.

6. ACKNOWLEDGMENTS

This work was supported by the National Science Council of Taiwan under Grants NSC 101-2221-E-001-017, NSC 102-2221-E-001-004-MY3 and the Academia Sinica Career Development Award.

7. REFERENCES

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54(11):4311–4322, 2006.

[2] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):1–37, 2011.

[3] J.-L. Durrieu, G. Richard, and B. David. An iterative approach to monaural musical mixture de-soloing. In *Proc. ICASSP*, pages 105–108, 2009.

[4] B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[5] D. Ellis. A phase vocoder in Matlab, 2002. [Online] http://www.ee.columbia.edu/ dpwe/resources/matlab/pvoc/.

[6] H. Fujihara, M. Goto, J. Ogata, and H. G. Okuno. Lyricsynchronizer: Automatic synchronization system between musical audio signals and lyrics. *J. Sel. Topics Signal Processing*, 5(6):1252–1261, 2011.

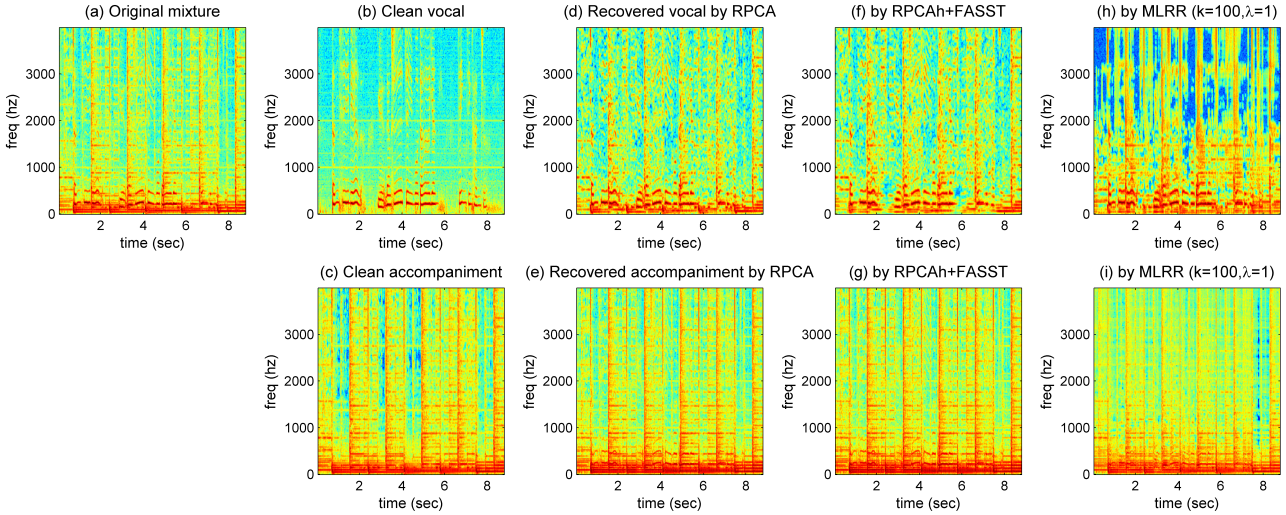


Figure 4. (a) The magnitude spectrogram (in log scale) of the mixture of singing and music accompaniment for the clip 'Ani_1_01' in MIR-1K [7]; (b) (c) The groundtruth spectrograms for the two sources; the separation result for (d) (e) RPCA [8], (f) (g) RPCAh+FASST [23], and (h) (i) the proposed method MLRR (k=100, λ=1) for the two sources, respectively.

[7] 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. Audio, Speech & Language Processing*, 18(2):310–319, 2010.

[8] P.-S. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *Proc. ICASSP*, pages 57–60, 2012.

[9] M. Lagrange, A. Ozerov, and E. Vincent. Robust singer identification in polyphonic music using melody enhancement and uncertainty-based learning. In *Proc. ISMIR*, pages 595–560, 2012.

[10] Z. Lin, M. Chen, L. Wu, and Yi Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. Technical Report UILU-ENG-09-2215, 2009.

[11] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Anal. & Machine Intel.*, 35(1):171–184, 2013.

[12] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proc. Int. Conf. Machine Learning*, pages 689–696, 2009.

[13] M. Müller, D. P. W. Ellis, A. Klapuri, and G. Richard. Signal processing for music analysis. *J. Sel. Topics Signal Processing*, 5(6):1088–1110, 2011.

[14] G. Mysore, P. Smaragdis, and B. Raj. Non-negative hidden Markov modeling of audio with application to source separation. In *Int. Conf. Latent Variable Analysis and Signal Separation*, pages 829–832, 2010.

[15] A. Ozerov, E. Vincent, and F. Bimbot. A general flexible framework for the handling of prior information in audio source separation. *IEEE Trans. Audio, Speech & Language Processing*, 20(4):1118–1133, 2012.

[16] Z. Rafii and B. Pardo. REpeating Pattern Extraction Technique (REPET): A simple method for music/voice separation. *IEEE Trans. Audio, Speech & Language Processing*, 21(2):73–84, 2013.

[17] P. Sprechmann, A. Bronstein, and G. Sapiro. Real-time online singing voice separation from monaural recordings using robust low-rank modeling. In *Proc. ISMIR*, pages 67–72, 2012.

[18] J. Sundberg. *The science of the singing voice*. Northern Illinois University Press, 1987.

[19] I. Tošić and P. Frossard. Dictionary learning. *IEEE Signal Processing Magazine*, 28(2):27–38, 2011.

[20] E. Vincent, R. Gribonval, and C. Fèvotte. Performance measurement in blind audio source separation. *IEEE Trans. Audio, Speech & Language Processing*, 16(4):766–778, 2008.

[21] T. Virtanen. Unsupervised learning methods for source separation in monaural music signals. In A. Klapuri and M. Davy, editors, *Signal Processing Methods for Music Transcription*, pages 267–296. Springer, 2006.

[22] D. Wang and G. J. Brown. *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. Wiley-IEEE Press, 2006.

[23] Y.-H. Yang. On sparse and low-rank matrix decomposition for singing voice separation. In *Proc. ACM Multimedia*, pages 757–760, 2012.

[24] C.-C. M. Yeh and Y.-H. Yang. Supervised dictionary learning for music genre classification. In *Proc. ACM Int. Conf. Multimedia Retrieval*, pages 55:1–55:8, 2012.

INCREMENTAL VISUALIZATION OF GROWING MUSIC COLLECTIONS

Sebastian Stober, Thomas Low, Tatiana Gossen & Andreas Nürnberger
Data & Knowledge Engineering Group, Faculty of Computer Science, University of Magdeburg, DE
{stober, thomas.low, tatiana.gossen, andreas.nuernberger}@ovgu.de

ABSTRACT

Map-based visualizations – sometimes also called projections – are a popular means for exploring music collections. But how useful are they if the collection is not static but grows over time? Ideally, a map that a user is already familiar with should be altered as little as possible and only as much as necessary to reflect the changes of the underlying collection. This paper demonstrates to what extent existing approaches are able to incrementally integrate new songs into existing maps and discusses their technical limitations. To this end, Growing Self-Organizing Maps, (Landmark) Multidimensional Scaling, Stochastic Neighbor Embedding, and the Neighbor Retrieval Visualizer are considered. The different algorithms are experimentally compared based on objective quality measurements as well as in a user study with an interactive user interface. In the experiments, the well-known Beatles corpus comprising the 180 songs from the twelve official albums is used – adding one album at a time to the collection.

1. INTRODUCTION

Within the last decade, two-dimensional maps have become a popular means for visualizing music collections. By providing a collection overview which easily allows to identify regions or neighborhoods of similar songs, they are especially helpful if users want to explore a collection without having to formulate an explicit query. There exists a large variety of approaches to compute maps. The most popular ones in the field of Music Information Retrieval (MIR) are Self-Organizing Maps (SOMs) [14] and Multidimensional Scaling (MDS) [15] as well as related techniques such as Principle Component Analysis (PCA) [10]. Considerable effort has been made to improve the quality and usefulness of the generated maps – e.g., by adapting the underlying similarity measure based on user feedback [11, 27], enriching the visualization with landscape features such as islands [5, 13, 21, 22, 24] and mountain ranges [18, 20], or correcting projection errors (caused by the inherent dimensionality reduction during map generation) with an adaptive distortion technique [29].

However, to the knowledge of the authors, the problem of changing collections has not yet been addressed appropriately. To what extent does a map change when songs are added or removed? Could it even be necessary to recompute the map from scratch? Such questions need to be answered as failing to support collection changes may significantly limit the usefulness of a MIR application in real-world scenarios. Here, being able to add songs to an existing map is more important than removal which is often trivial (at worst leading to blank spaces in the map) and an uncommon use case anyways. New songs that are similar to existing ones should be embedded in the respective neighborhoods. At the same time, the map should also be able to deal with changes in music taste (adding songs from new genres) and an increase of musical diversity. Ideally, a map should be altered as little as possible and only as much as necessary to reflect the changes of the underlying collection. Too abrupt changes in the topology might confuse the user who over time will get used to the location of specific regions in the map.

In this paper, we compare several popular approaches for generating two-dimensional maps of music collections with respect to their ability to deal with growing collections. Section 2 briefly reviews the algorithms covered in the comparison and points out related work. The evaluation has been two-fold: We conducted objective measurements (Section 3) as well as a user study (Section 4). Section 5 summarizes our findings and draws conclusions.

2. LAYOUT ALGORITHMS

2.1 Multidimensional Scaling (MDS)

Given a set of data points, classical MDS [15] finds an embedding in the target space (here $\mathbb{R}^2$) that maintains their distances (or dissimilarities) as far as possible – without having to know their actual values. This way, it is also well suited to compute a layout for spring- or force-based approaches or as a method for vectorization (Section 2.3). MDS is closely related to PCA [33], which projects data points simply onto the (two) axes of highest variance termed principal components. In contrast to SOMs, both are non-parametric approaches that compute an optimal solution (with respect to data variance maximization and distance preservation respectively) in fixed polynomial time. Systems that apply PCA, MDS or similar force-based approaches comprise [1, 6, 31], the *fm4 Soundpark* [5], *MusicBox* [16], and *SoundBite* [17].

MDS does not support incremental collection changes.

Instead, a new map has to be computed every time the collection grows. Even with little change of the collection, the resulting map may look very different because it could be arbitrarily translated, rotated, and reflected without affecting the pairwise distances. In order to remedy this issue, we apply Procrustes analysis [7] to align each newly generated map with the previous one. This method allows to transform a set of data points through translation, rotation, and uniformly scaling such that it resembles another set as closely as possible.

2.2 Landmark Multidimensional Scaling

Landmark MDS as described in [2] is a computationally efficient approximation to classical MDS. The general idea of this approach is as follows: Given a sample set of landmark or pivot objects, an embedding into a low-dimensional space is computed for these objects using classical MDS. Each remaining object can then be located within the output space according to its distances to the landmarks. Obviously, the quality of the projection depends on the choice of the landmarks – especially if the landmark sample set is small compared to the size of the whole dataset. If the landmarks lie close to a low-dimensional subspace (e.g., a line), there is the chance of systematic errors in the projection.

As described in [29], Landmark MDS can be applied to visualize growing music collections by using the initial songs as landmarks. This way, the position of a song once added to the map never changes. However, the landmark set may become less and less representative with increasing collection size and possibly changing music taste. This may have a significant effect on the quality of the projection as, e.g., already observed when applying Landmark MDS for vectorization [28].

2.3 Growing Self-Organizing Map (GSOM)

SOMs are commonly applied for structuring data collections by clustering similar objects into identical or neighboring cells of a two-dimensional grid. In the field of MIR, they have been used in a large number of applications like the *Islands of Music* [22, 24], the *MusicMiner* [20] or *nepTune* [13]. A recent overview on SOM-related MIR publications is given in [30]. Growing Self-Organizing Maps (GSOMs) have the advantage that the structure of the cell grid does not have to be specified prior to training. Instead, they can grow as needed and adapt incrementally to changes in the underlying collection whereas other approaches may always need to generate a new structuring when the grid becomes too small. Growing hierarchical SOMs have been applied in [3, 23, 25]. In contrast, the *BeatlesExplorer* [27] uses a GSOM that grows by adding new cells at the outer boundary. The same approach is used here as a flat structure is desired. For the initialization, a small hexagonal grid of 2×2 cells is chosen. After a regular training phase with a fixed number of iterations, an internal error is computed for each cell as a measure of quality. To this end, the pairwise distance of the objects contained in a cluster is used. Unless the maximum error is below a threshold which is specified as stopping criterion, a new cell is added next to the border cell with the highest error. Here, a very low threshold is used to produce a large map with cells that only contain few songs. In order to reduce visual overlapping of songs in the same cell, song coordinates are computed by taking the distance-weighted mean of the cell's coordinates and those of its direct neighbors. This way, songs are placed slightly off-center.

SOMs generally require the objects they process to be represented as vectors, i.e., elements of a vector space. As the feature representation does not adhere to this condition, some means of vectorization is required. This issue has been discussed in depth in [28] with the recommendation to use MDS for vectorization. For growing collections where only a fraction of the collection may be available for the initial vectorization, the Landmark MDS approach has to be used whereby the songs of the initial collection serve as landmarks. As observed in [28], this may cause a significant drop in the nearest neighbor retrieval precision if many new songs are added – especially if those are very different from the initial ones.

2.4 Stochastic Neighbor Embedding (SNE)

Rather than trying to maintain pairwise distances like MDS, the objective of SNE [9] is to preserve the *probabilities* of points being neighbors. To this end, probability distributions $p_{j|i}$ on how likely it is that the point j is a neighbor of point i are defined based on the input space distances using a Gaussian neighborhood function with width σ_i^2. The algorithm then tries to find suitable coordinates in the output space that lead to (approximately) identical probabilities $q_{j|i}$ for each pair of points. The Kullback-Leibler divergence

$$D_{KL}(p_i, q_i) = \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} \tag{1}$$

between the probability distributions $p_{j|i}$ in the input space and $q_{j|i}$ in the output space serves as cost function. Starting from randomly initialized coordinates (close to the origin), a conjugate gradient method is used for finding a (locally) optimal solution.

SNE does not support changes in the collections. However, it is possible to replace the random initialization and instead use the coordinates from the current visualization for those points already present. This way, the search for a new solution (incorporating the new points) starts from the current one. Whilst this does not give any guarantee that the resulting map resembles the old one, chances are that the difference is small.

2.5 Neighbor Retrieval Visualizer (NeRV)

As shown in [32], SNE focuses on the cost of missing similar objects and – by using continuous probabilities to model neighborhood relationships – optimizes a *smoothed* version of the mean recall, $\sum_i D_{KL}(p_i, q_i)$. [1] NeRV additionally takes the *mean smoothed precision* into account by

[1] The *smoothed mean recall* and the *smoothed mean precision* as introduced in [32] are in fact cost functions to be minimized in contrast to

considering also the cost of retrieving dissimilar objects, $\sum_i D_{KL}(q_i, p_i)$. The resulting combined cost function is

$$E = \lambda \sum_i D_{KL}(p_i, q_i) + (1 - \lambda) \sum_i D_{KL}(q_i, p_i) \quad (2)$$

where the parameter $\lambda \in [0, 1]$ controls the trade-off between precision and recall. For $\lambda = 1$, the cost function of SNE is obtained. Another important parameter is the scale of the neighborhoods, σ_i^2, for the computation of the probability distributions $p_{j|i}$ and $q_{j|i}$. Starting with a large value and reducing it stepwise after each optimization step helps to avoid local minima during the conjugate gradient optimization. Like SNE, NeRV does not support collection changes but can be initialized with given coordinates.

3. VISUALIZATION QUALITY MEASUREMENTS

3.1 Test Collection: The Beatles Corpus

For our experiments, we used the well-studied corpus of the 12 official albums of The Beatles containing 180 songs. The albums were added one-by-one to the collection in the order of their release. Distances between the songs was computed as a simple linear combination over three content-based features: Using the CoMIRVA framework [26], we extracted Gaussian Mixture Models of the Mel Frequency Cepstral Coefficients (MFCCs) [19] and the "fluctuation patterns" described in [24]. Furthermore, we derived chord frequency distributions from the publicly available ground truth annotations [8]. This feature combination was chosen to capture common acoustic properties as well as to reduce the problem of hubs [4].

3.2 Quality Measures

3.2.1 Continuity and Trustworthiness

A visualization can be considered *trustworthy* if the k nearest neighbors of a point on the display are also neighbors in the original space. Following [12], a respective measure of trustworthiness can be computed by:

$$M_{trustworthiness} = 1 - C(k) \sum_{i=1}^{N} \sum_{j \in U_k(i)} (r_{ij} - k) \quad (3)$$

where r_{ij} is the rank of j in the ordering of the distance from i in the original space, $U_k(i)$ is the set of i's false neighbors in the display, and $C(k) = 2/(Nk(2N - 3k - 1))$ is a constant for obtaining values in $[0, 1]$.

Analogously, the measure of *continuity* considers the k nearest neighbors in the original space and captures how well they are preserved in the visualization [12]:

$$M_{continuity} = 1 - C(k) \sum_{i=1}^{N} \sum_{j \in V_k(i)} (\hat{r}_{ij} - k) \quad (4)$$

Here, $\hat{r}_{ij}$ is the rank of j in the ordering of the distance from i in the visualization and $V_k(i)$ is the set of i's true neighbors missing in the visualized neighborhood.

the traditional definition of precision and recall in information retrieval. For simple "binary" neighborhood definitions, the Kullback-Leibler divergences and the precision-recall measures become equivalent [32].

3.2.2 Mean Smoothed Rank-Based Precision and Recall

As a second pair of evaluation measures, it is possible to directly use mean smoothed precision and recall (described in Section 2.5) as error measures. However, these two errors have no upper bound and their scale depends on the dataset. Thus, it would only be possible to compare values for the same sub-collection. As data-independent alternatives, the *mean smoothed rank-based precision and recall* have been proposed in [32]. The idea is to replace the data-dependent distances in the computation of the probability distributions by ranks, i.e., the nearest neighbor gets assigned a distance of 1, the second closest a distance of 2 and so on. The worst case scenario of reversed ranks gives an upper bound that can be used for normalization so that all values lie in $[0, 1]$. [2]

3.2.3 Mean Position Change

Finally, a measure was needed to capture the amount of change in the visualization. To this end, we computed the average Euclidean distance between the initial and updated coordinates of all points contained in two consecutive visualizations. In order to obtain normalized values, the point sets were scaled and translated to fit into the unit square. This resulted in a maximal position change of $\sqrt{2}$ for a single point.

3.3 Results

Figure 1 shows the values of the five quality measures for each step of adding another album to the collection. We chose neighborhoods of size $k = 5$ for measurements. The same parameter value was also used for the NeRV and SNE cost functions. The NeRV parameter λ was set to zero to optimize precision as contrast to SNE. Applying the MDS vectorization method on the initial collection, which only contained the first album ("Please Please Me"), yielded 13-dimensional real vectors to be processed by the GSOM. This vectorization was used in all measurements. In comparison to using the full dataset, which would result in 143 dimensions, a small retrieval performance penalty of 1-4% was observed. This matches with the 5% drop in the 10-nearest-neighbor retrieval precision reported in [28] for a 14:166 ratio of landmarks and new songs. The penalty can be expected to increase for more extreme ratios.

Considering only the overall visualization quality, NeRV clearly and consistently yielded the best results. It was surprisingly only slightly outperformed in continuity and recall by SNE, which is specifically optimizing the latter measure while producing significantly worse results for trustworthiness and precision. In general, all methods did similarly well in continuity and recall, i.e., the number of actual neighbors being misplaced in the visualization (false negatives) was rather low. For trustworthiness and precision, the differences were much more obvious indicating

[2] The mean smoothed rank-based measures are still error measures, i.e., a value of zero indicates optimal performance. To obtain consistency with the traditional precision-recall measures, we additionally transform them by $f(x) = 1 - x$ as proposed in [32].

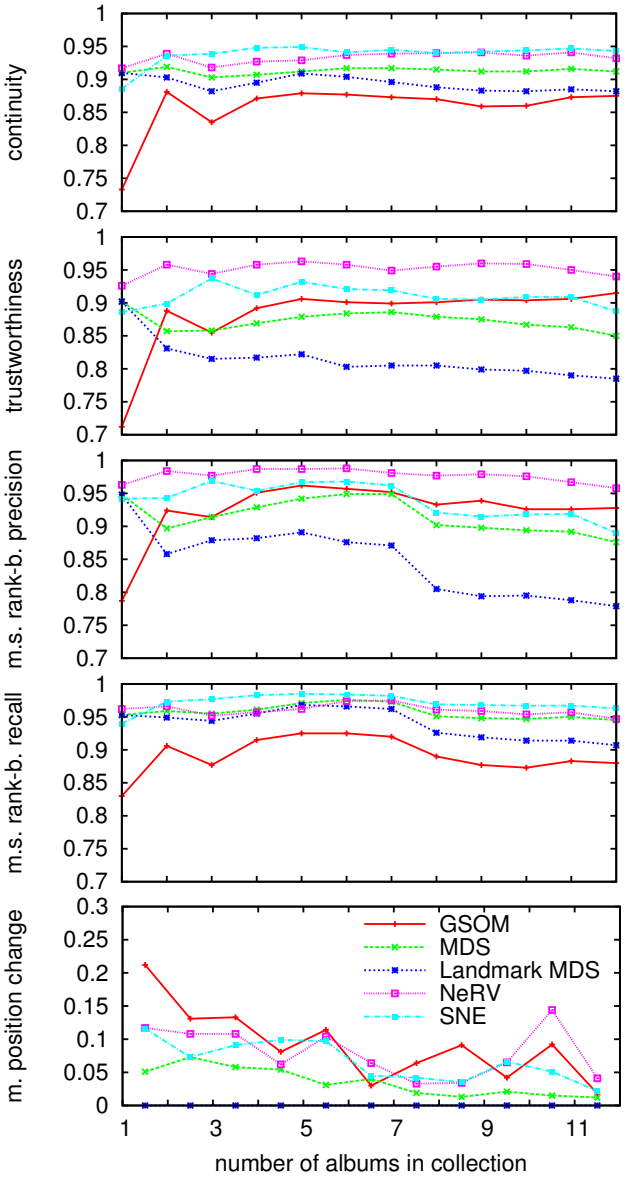


Figure 1. Measurements of the visualization quality and incremental position change. In the top four diagrams, higher values indicate better visualization quality whereas small values of change are desired in the bottom diagram.

issues with neighbors in the visualization that did not belong to the input space neighborhood (false positives). Values were also less constant here. Especially for Landmark MDS, they decreased as more and more songs were added. A significant drop in precision could be observed when the eighth album ("Sgt. Pepper's Lonely Hearts Club Band") was added to the collection.

Whilst NeRV generated an excellent visualization in each step, the difference between them was often much higher than for the other algorithms. Only the GSOM performed worse here. This might at least partly be due to the GSOM's grid-based structure, which leads to a kind of position quantization. Hence, when a song is re-assigned to a neighboring cell (the smallest possible position change), its position already changes roughly by the grid cell width. Landmark MDS was the clear winner here with no position changes at all, but this came at the cost of degrading visualization quality. MDS appears to offer a good compromise: The amount of change was lower than for the other methods (except Landmark MDS) and it visibly decreased as the fraction of new songs became smaller. At the same time, the recall and continuity value were comparable to NeRV. Only trustworthiness and precision were about 10% below NeRV.

4. USER STUDY

In this section, we describe the design and results of a comparative user study as qualitative evaluation. In order to reduce the effort for the participants, we only tested MDS, GSOM, and NeRV. Landmark MDS was not considered because there is no observable change for existing songs that could be evaluated and SNE because of its close relation to the superior NeRV method. The following research questions were addressed: *How well can users follow the changes when adding a new album, what algorithm do users prefer and why?* The study was conducted using the Beatles corpus and the same visualizations evaluated in the quantitative measurements described in Section 3.

4.1 Study Design

The user study was designed as follows: In a pre-interview, we gathered the users' demographic information, computer skills and their knowledge about The Beatles and map-based visualizations. Then, a lab experiment with within-participants design was performed, i.e., each participant viewed all three visualizations. We used a *Latin square design* to reduce the bias caused by the order in which participants were introduced to the different layout algorithms, i.e., a third of the participants tested MDS first, GSOM second and NeRV at last, whereas another third evaluated the algorithms in the order GSOM-NeRV-MDS and another third used the ordering NeRV-MDS-GSOM. After a short trial period of free interactions, we asked all participants to perform a memorization task in the form of a game similar to thimblerig. At each step, starting with the first album of The Beatles, three songs were highlighted randomly for five seconds as shown in Figure 2. Shortly after, participants were asked to track these songs during the transition of the visualization to the next step. The task was to find the previously highlighted songs in the updated layout. There was no time limit. The number of errors made by each user was recorded as a measurement for their performance. Thus, for each algorithm, we gathered errors made at each of the 11 (as we used 12 albums) steps. After the test on each algorithm, users were asked how well they were able to track the changes in the songs' layout. Finally, we asked the participants what algorithm they preferred. The whole procedure took about 30 minutes.

4.2 Study Results

We performed the user study with 19 subjects. They were between 23 and 38 years old (28 on average), eight women and eleven men. 68% of them were professional computer

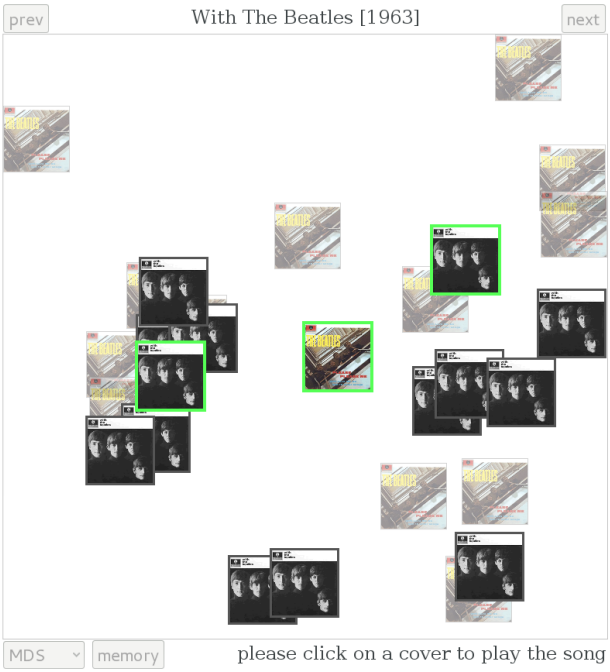


Figure 2. Graphical user interface for the study. [4] Here, a collection containing the first two albums of The Beatles is visualized using the MDS algorithm. The participants were asked to track all three highlighted songs (covers with green border) as they move during a transition.

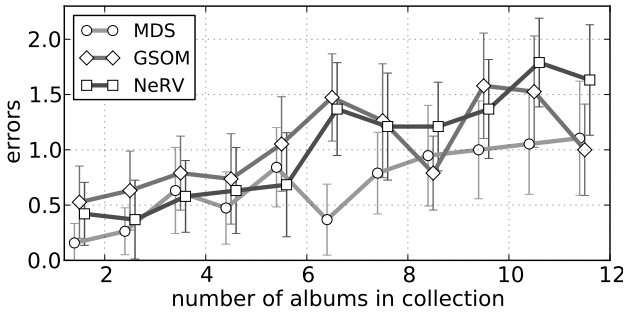


Figure 3. Mean of memorization errors for each transition, and confidence intervals ($\alpha = 0.05$).

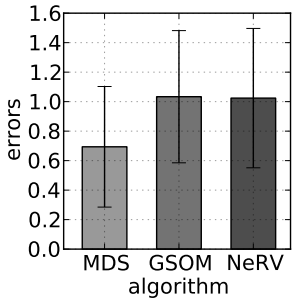


Figure 4. Mean memorization errors over all transitions, and confidence intervals ($\alpha = 0.05$).

users, whereas the rest had intermediate level. 68% of participants were students or PhD students in computer science. Only 21% were well acquainted with the music of The Beatles, 47% were familiar with map visualization.

Thirteen participants chose MDS, six chose NeRV and one chose GSOM as their favorite layout algorithm (one chose both MDS and NeRV). Users described MDS to result in less positional changes, NeRV to better preserve cluster structures and GSOM to have less overlappings. The results of the memorization task are shown in Figure 3. With an increasing number of albums, it became more difficult to find the requested songs. At later steps, participants noted that it was hard to select the desired songs because of many overlappings. Also, at some transitions, e.g., from six to seven albums, there are significant differences between algorithms. In total, the layout generated by the MDS algorithm resulted into the least mistakes as shown in Figure 4. Unfortunately, our results are not statistically significant, presumably due to the low number of participants. However, there was a strong tendency to the MDS algorithm. Most participants (14) achieved their best results using the MDS algorithm.

5. CONCLUSIONS

In this paper, we evaluated five popular methods for producing two-dimensional maps of music collections with respect to their ability to deal with incrementally growing collections. Two of these, GSOMs and Landmark MDS, have this ability by design. The other three, MDS, SNE, and NeRV, had to be slightly modified for this purpose. We conducted objective measurements – tracking the visualization quality and the change of positions over time – and a user study. Considering both the measured results and the feedback from the participants of the user study, MDS can be proclaimed as "winner". This is much to our surprise. We did not expect MDS (in combination with the Procrustes analysis for alignment) would produce such nice incremental visualization updates as observed here. Unlike the second best, NeRV, MDS has furthermore the advantage that it does not incorporate randomness and that it is guaranteed to find a *globally* optimal solution.

It is possible though highly unlikely that the outcome was in part caused by the choice of the collection. Therefore, we will conduct further experiments to verify the results using different datasets and also different media like texts and images. Another highly promising possibility for future work is to modify NeRV to better support incremental collection changes – e.g., by introducing an additional weighted term to its cost function (Equation 2) that penalizes position changes. This way, a new method could be designed that results in less dramatic position changes but maintains NeRV's superior performance for trustworthiness and precision. For reproducibility and to encourage possible testing with further methods, the dataset (in particular the distance matrix used as input for the algorithms) will be made publicly available.

[4] For better readability, the screenshot was taken using a much smaller screen resolution than in the user study. An online demo is available at http://demos.dke-research.de/beatles-history-explorer/

Acknowledgments We would like to thank all participants of the user study and J. Venna et al. for sharing their NeRV implementation code. The work in this paper has been funded in part by the German Federal Ministry of Education and Science (BMBF) through the Forschungscampus STIMULATE under Grant No. FKZ:03FO16103A.

6. REFERENCES

[1] P. Cano, M. Kaltenbrunner, F. Gouyon, and E. Batlle. On the use of fastmap for audio retrieval and browsing. In *ISMIR*, 2002.

[2] V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Adv. in Neural Information Processing Systems 15*, 2002.

[3] M. Dopler, M. Schedl, T. Pohle, and P. Knees. Accessing music collections via representative cluster prototypes in a hierarchical organization scheme. In *ISMIR*, 2008.

[4] A. Flexer, D. Schnitzer, M. Gasser, and T. Pohle. Combining features reduces hubness in audio similarity. In *ISMIR*, 2010.

[5] M. Gasser and A. Flexer. Fm4 soundpark: Audio-based music recommendation in everyday use. In *Proc. of 6th Sound and Music Computing Conf.*, 2009.

[6] D. Gleich, M. Rasmussen, L. Zhukov, and K. Lang. The World of Music: SDP layout of high dimensional data. In *Info Vis*, 2005.

[7] J.C. Gower and G.B. Dijksterhuis. *Procrustes Problems*. Oxford University Press, 2004.

[8] C. Harte, M.B. Sandler, S.A. Abdallah, and E. Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *ISMIR*, 2005.

[9] G.E. Hinton and S.T. Roweis. Stochastic neighbor embedding. In *Adv. in Neural Information Processing Systems 15*, 2002.

[10] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 2002.

[11] C. F. Julia and S. Jorda. SongExplorer: a tabletop application for exploring large collections of songs. In *ISMIR*, 2009.

[12] S. Kaski, J. Nikkila, M. Oja, J. Venna, P. Toronen, and E. Castren. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics*, 4(1):48, 2003.

[13] P. Knees, T. Pohle, M. Schedl, and G. Widmer. Exploring Music Collections in Virtual Landscapes. *IEEE MultiMedia*, 14(3):46–54, 2007.

[14] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

[15] J.B. Kruskal and M. Wish. *Multidimensional Scaling*. Sage, 1986.

[16] A.S. Lillie. Musicbox: Navigating the space of your music. Master's thesis, MIT, 2008.

[17] S. Lloyd. Automatic playlist generation and music library visualisation with timbral similarity measures. Master's thesis, Queen Mary Univ. of London, 2009.

[18] D. Lübbers and M. Jarke. Adaptive multimodal exploration of music collections. In *ISMIR*, 2009.

[19] M. Mandel and D. Ellis. Song-level features and support vector machines for music classification. In *ISMIR*, 2005.

[20] F. Mörchen, A. Ultsch, M. Nöcker, and C. Stamm. Databionic visualization of music collections according to perceptual distance. In *ISMIR*, 2005.

[21] R. Neumayer, M. Dittenbach, and A. Rauber. PlaySOM and PocketSOMPlayer, alternative interfaces to large music collections. In *ISMIR*, 2005.

[22] E. Pampalk, S. Dixon, and G. Widmer. Exploring music collections by browsing different views. In *ISMIR*, 2003.

[23] E. Pampalk, A. Flexer, and G. Widmer. Hierarchical organization and description of music collections at the artist level. In *Proc. of 9th Eu. Conf. on Research and Advanced Technology for Digital Libraries*, 2005.

[24] E. Pampalk, A. Rauber, and D. Merkl. Content-based organization and visualization of music archives. In *Proc. of 10th ACM int. Conf. on Multimedia*, 2002.

[25] A. Rauber, E. Pampalk, and D. Merkl. Using psychoacoustic models and self-organizing maps to create a hierarchical structuring of music by musical styles. In *ISMIR*, 2002.

[26] M. Schedl. The CoMIRVA Toolkit for Visualizing Music-Related Data. Technical0 report, Johannes Kepler University Linz, 2006.

[27] S. Stober and A. Nürnberger. Towards user-adaptive structuring and organization of music collections. In *Proc. of 6th int. workshop on Adaptive Multimedia Retrieval*, 2008.

[28] S. Stober and A. Nürnberger. Analyzing the impact of data vectorization on distance relations. In *2011 IEEE Int. Conf. on Multimedia and Expo*, 2011.

[29] S. Stober and A. Nürnberger. Musicgalaxy: A multi-focus zoomable interface for multi-facet exploration of music collections. In *Exploring Music Contents*, volume 6684 of *LNCS*, pages 273–302, 2011.

[30] S. Stober and A. Nürnberger. Adaptive music retrieval – a state of the art. *Multimedia Tools and Applications*, 65(3):467–494, 2013.

[31] R. van Gulik and F. Vignoli. Visual playlist generation on the artist map. In *ISMIR*, 2005.

[32] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *Journal of Machine Learning Research*, 11:451–490, 2010.

[33] C.K.I. Williams. On a connection between kernel pca and metric multidimensional scaling. *Machine Learning*, 46(1-3):11–19, 2002.

EVALUATING OMR ON THE EARLY MUSIC ONLINE COLLECTION

Laurent Pugin
Swiss RISM Office
laurent.pugin@rism-ch.org

Tim Crawford
Goldsmiths College, University of London
t.crawford@gold.ac.uk

ABSTRACT

The Early Music Online (EMO) collection consists of about 300 printed music books of the sixteenth century held at the British Library. They were recently digitized from microfilms and made available online. In total, about 35,000 pages were digitized. This paper presents an optical music recognition (OMR) evaluation on the EMO collection. Firstly, the content of the collection is reviewed, looking at the type of music notation and the type of printing technique. Secondly, for the books for which it is possible (260 books), an OMR evaluation performed using the Aruspix OMR software application is presented. For each book, one randomly selected page of music was processed and the recognition rate was computed using a corrected transcription of the page. This evaluation shows very promising results for large-scale OMR on the EMO or similar collections. The paper also highlights critical points that should be taken into account in such an enterprise.

1. INTRODUCTION

Over the last decade, libraries around the world have been digitizing their collections extensively, making available online not only books but also music scores. For books, digitization projects most often include an optical character recognition (OCR) step that renders the content of the images searchable in a similar way, for example, to Google Books. Enabling full-text access has radically changed the search paradigm for the user; we no longer search in the same way with the complete content of books as we used to with only the restricted metadata available in a library catalogue. When music scores are digitized, however, they usually remain searchable only through textual metadata, since no transcription of the content is made available during the digitization process. This is mostly due to the fact that optical music recognition (OMR), the equivalent of OCR for music, is a very challenging task [11]. Furthermore, performing OMR on a large scale is particularly complicated because of the heterogeneous nature of music notation and of different types of music scores. It makes it difficult to set up a solution that works acceptably well in all cases. However, preliminary attempts to perform OMR on a large scale on the IMSLP Petrucci Music Library have shown interesting results even if at this stage they are still lacking a clear and systematic evaluation [13].

With historical documents in general, performing OCR remains a challenge. In particular, some steps of the OCR process remain critical, in particular the preprocessing of documents that are often highly degraded, and the layout analysis. More recently, large research projects have focused on OCR for historical documents, such as the IMPACT project, with the aim of developing specific tools for the task [2]. For historical music documents, the leading OMR project is SIMSSA, currently with a focus on the handwritten repertoire in neumes and square notation (before 1500) used for experimenting new methods and testing innovative online tools [7].

This paper focuses on printed music from the 16th century, a period of enormous expansion in the availability of music for domestic use. It was made possible by the development of a typographic technique adapted from Gutenberg's invention of typesetting. This remained by far the most widely-used method for printing music until the beginning of the 18th century when it was superseded by engraving. Typographic music printing was invented by Ottaviano Petrucci in 1501 and simplified by Pierre Attaingnant in 1527. The former introduced a technique using two or three passes through the press, while Attaingnant developed a single-impression technique that made the process commercially sustainable, with the consequence that the number of printed scores increased dramatically from then on. Printed music scores of the time represent a valuable and rich source of material for musicologists, musicians and historians alike.

We present in this paper an evaluation of OMR on the Early Music Online collection, a set of about 300 books held at the British Library and recently digitized from microfilms [12]. The evaluation is twofold. First we evaluated which of the books in such a collection could be processed using the Aruspix OMR software application developed specifically for typeset music prints.[1] In particular, we were interested to see what percentage of a library collection like this can be processed with existing technology. For this, we looked at the type of music notation, the type of format and the type of printing technique. With this in hand, we were able to define a set of books

[1] <http://www.aruspix.net>

to be used for the second part of our evaluation, which was to estimate what recognition rate we can obtain in an OMR workflow without human correction of the data.

In the next section, we introduce the EMO data set. The experiments for evaluating the OMR recognition rates and the results are presented in the following sections. Remarks on future work conclude the paper.

2. THE EMO COLLECTION

In 2011, the British Library together with Royal Holloway, University of London and the UK group of the Repertoire International des Source Musicales (RISM) conducted a digitization project of sources held at the British Library. In total, 324 volumes of 16th-century printed music, mostly individual voice part-books of vocal music, were re-catalogued, digitized and made available online together with very rich metadata that includes information about the title pages, the dedicatees, the printer and the printing place, and more. It comprises approximately 35,000 pages for a total of about 10,000 pieces of music. The books chosen for this digitization project were selected from the RISM B/I and B/II Series covering printed anthologies. Anthologies (collections of music by multiple composers) were selected as a priority because their content is not as well known as that of individual prints (RISM Series A/I). The books were digitized from the microfilm collection of the British Library and are available under the JISC Collections Open Education User Licence version 1.0.

2.1 Notation types

Most of the books in the EMO collection are in mensural music notation on five line staves, the most common notation of the time. The collection also contains twenty-seven books of lute tablature (e.g., K.1.c.11)[1] and about five books of organ tablature (e.g., K.8.h.22). Six books are interesting cases with a mix of mensural notation and lute tablature (e.g., D.250 and K.2.h.12). There is also one book that contains a very rare case of a mixture of five-line mensural notation and four-line square notation, printed by Etienne Gueynard in Lyon in 1528 (K.9.a.23).

2.2 Book formats

The large majority of the books of the EMO collection are part-books, a standard format of the time. In part-books, each voice or instrumental part is printed in a separate gathering. Sometimes, two or more voices are printed in the same book, for example when a book in four voices (i.e., four part-books) also contains pieces with more voices. In these cases, the different voices appear as in a choir book for the singers or musicians to be able to sing or play together by sharing the part-book. The EMO collection also contains proper choir books (e.g., K.9.a.11) where all voices are printed in a single book. Finally, the EMO collection contains ten table books mostly printed by Jacques Moderne in Lyon (e.g., K.10.a.7). The table books are similar to choir books with the main difference that the voice-parts are not all printed in the same direction. They were intended to be laid on a table with musicians sitting on different sides of the book. All the table books in EMO contain parts printed upside-down, but books with parts in three or four directions also exist.

2.3 Printing techniques

About 300 books in the EMO collection were printed using the single impression typographic technique introduced by Attaingnant. His innovation was to use typefaces where the staff lines and the notes were printed at the same time; this became the most widely-used technique for printed music for nearly two centuries. The EMO collection also contains 13 books printed with the multiple impression technique, mostly printed by Petrucci (e.g., K.1.d.12), one of the very few printers who used this technique. There are also seven books printed from engraved plates by Simone Verovio (K.8.b.17) who was one of the first music engravers, and five books in woodcut mostly by Andrea Antico (e.g., K.8.b.7).

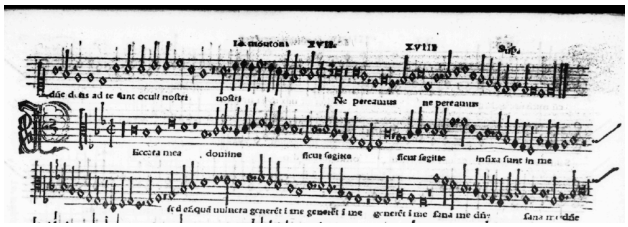


Figure 1. A page printed in multiple impressions by Petrucci left out of the experiments. The processing of pages printed with this technique is problematic when ornate letters are superimposed on the music staves.

3. EXPERIMENTS

3.1 Data set for the OMR evaluation

By looking at the different characteristics in terms of notation type, book format and printing technique, we were able to determine which were the books that could be processed successfully with the OMR tool that was to be used for the evaluation. We selected all the books in mensural notation printed with the single impression technique.

We left out the books in multiple impressions, even though they could also have been processed, albeit certainly yielding less accurate results. As shown on Figure 1, ornate letters are sometimes superimposed on the music staves, which makes small areas of the page badly recognized by the OMR software application used for our evaluation. We also left out lute tablatures and books printed from engraved plates that are not appropriate for the OMR software application. Finally, we also left out

[1] The reference for the books is their signature at the British Library. Images can be accessed from the RISM UK website (www.rism.org.uk).

the table books, the books with a mixture of lute and mensural notation and also the book with square notation.

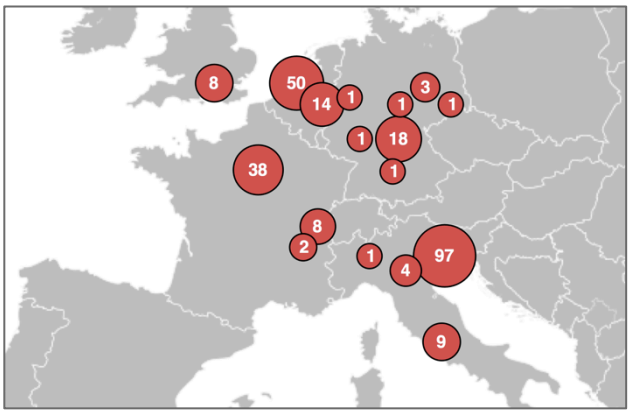


Figure 2. The 260 books selected for the OMR evaluation were printed in 17 different cities, reflecting the importance of the music printing centres of the time.

We ended up with a total of 260 books representing 80% of the EMO collection. These 260 books were printed by no less than 48 different printers coming from 17 cities (plus two from unknown locations). Interestingly, the provenance of the books reflects quite precisely the music printing production of the different cities at that time (see Figure 2); in particular, more than a third of the books were printed in Venice [3].

The diversity of the data set can also be visualized by looking at the different font shapes. In Table 1, we present the 18 different G-clef shapes found in the pages we processed for our evaluation.

Amadino	Antico	Ballard
Donangeli	Ang. Gardano	Gerlach
Hucher	Moderne	Petrejus
Phalèse	Scotto	Short
Susato	Vissenaken	Waelrant
[unknown]	[unknown]	[unknown]

Table 1. The font and woodcut shapes in the books used for the evaluation are highly variable as illustrated by the 18 different G-clef shapes appearing in the data set.

3.2 Procedure

The experiments were performed using the original gray-scale images in uncompressed TIFF format at 400dpi, the resolution used by the British Library during the EMO digitization process. This resolution, however, does not provide accurate information about the original document size because of all the different parameters that were involved in the making of the microfilms and the scanning. As an indication, looking at the staff height in pixels, we noticed in the EMO images that the staff height goes from 60 up to 300 pixels, with an average value of 122 pixels and a median value of 108 pixels. Of course, this is only an indication since the actual staff sizes on the sources themselves vary.

In the EMO images, each image always contains two pages. Since the OMR tool used for our experiments works on single page images, the images had to be split. The images were split automatically by selecting a little more than half the image width for each page (55% of the width). Because the books are always centered in the EMO images and because the tool performs a border removal, this simple method worked well in all cases. For each of the 260 books of our data set, one page of music was randomly selected. For the books where the random selection process did not retrieve a page containing music (9 books), the first subsequent page of music was selected by hand.

The recognition process in the tool used for the experiments includes a binarization step that converts the gray-scale image into a black and white one. This step has been shown to be critical on degraded music documents [8]. The OMR tool offers two options for the binarization, one for documents in good condition (B1), and another based on an algorithm specifically designed for highly degraded documents with marked bleed-through (B2). The B1 algorithm is the Brink & Pendock 1996 algorithm, and B2 its modified three-class version, Pugin 2007, the two best-performing algorithms described in [4].

The choice of the binarization option is left to the user and is not determined by the tool. In our experiments, we allowed the evaluator to choose in advance the algorithm that seemed the most appropriate for each book based on a visual evaluation of its degradation. B1 was applied to 159 books, and B2 to 101. In order to evaluate how appropriate the choice of the evaluator was, we also processed all the pages for which B2 was selected with the B1 options and compared the results.

All the pages were processed without human interaction. That is, without any processing indication or correction other than the binarization method selection. In that sense, the results provide a clear indication of what would be obtained with this technology on a large-scale OMR process involving minimal human intervention for preparing the process and no data correction at all.

4. RESULTS

All but 2 of the 260 pages were successfully recognized. Only two pages produced unusable results because the staff detection step in the OMR process failed. The two pages were taken from books printed by Giorgio Marescotti (C.219.a) and Antonio Gardano (K.3.l.7).

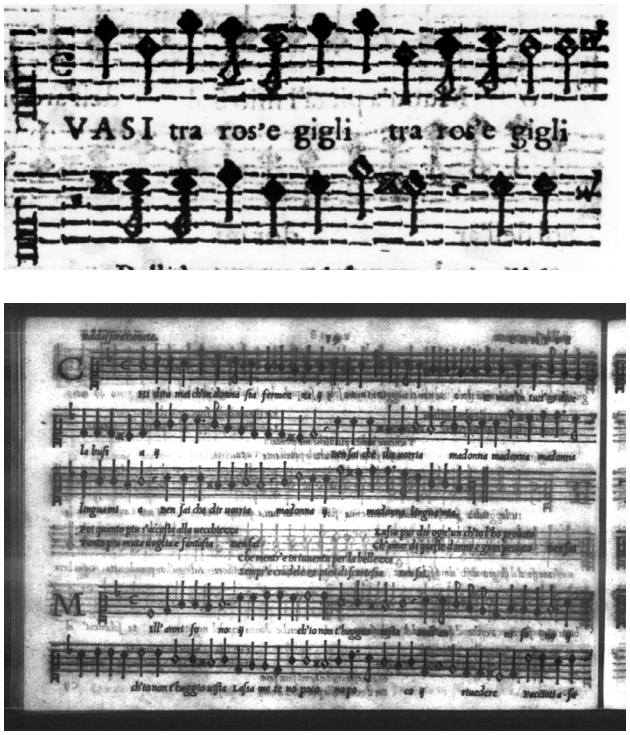


Figure 3. For two pages in the data set the recognition process failed. With the first, both the quality of the typeface used for printing and the quality of the microfilm were poor. With the second, the bleed-through was too strong and the microfilm quality poor.

Both failures can be explained by the poor quality of the documents as shown in Figure 3 for the prints by Marescotti and Gardano. In the first case, the quality of the print itself was very bad because it had been printed with a deteriorated typeface that produced extremely irregular staff lines. The quality of the microfilm and of the image was also problematic. In the second case, the book was highly degraded, with very marked bleed-through. Nothing in the layout of the documents indicates that the recognition would have been problematic, had they been in better condition.

For all the pages that were processed successfully (258), the OMR transcription was fully corrected by hand in order to be able to compute recall and precision following the evaluation procedure previously used for the tool [10]. This method uses a standard best alignment approach for evaluation of the number of substitutions, deletions and insertions. For all symbols, exact matching is required, e.g., for a music note, a match occurs only when both pitch and duration are the same. It is also important to mention that the misrecognition of a clef does not affect the evaluation of the subsequent pitches, even though technically pitches would be incorrect – it is also not rare to see cases where the clef is actually wrong in the source itself.

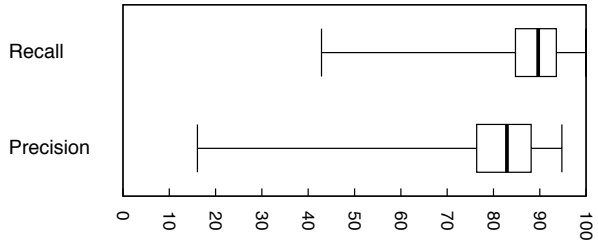


Figure 4. Aside from a few pages that produced poor results, the quartiles show that three quarters of the pages produced a recall rate between 85% and 100% and a precision rate between 76% and 95%.

The median recall rate is 90% and the median precision rate is 83%. Some pages did produce bad results. However, it can be seen that this is the case for only a few of them. Although the lowest recall rate is just 43%, the lower recall rate quartile is 85%, as shown on Figure 4. We noticed that the lowest recall rate was obtained on a fairly atypical page; it contained only two staves with few music symbols on them and also has added handwritten slurs, which explain the poor results.

Looking at the results by printer did not highlight very significant differences. Figure 5 shows the quartiles for the five printers for which we had more than 10 books in the data set. Overall, it seems that document degradation has more impact on the results than the printers, although the difference is slight.

Some of the books in the data set were printed using nested typefaces. With this technique, types with not only five lines but also four or three lines are used together with line elements overlapping several types horizontally. The technique had the advantage of requiring a smaller number of type shapes. It was used mostly in the Low Countries, in France and in Germany, by printers such as Robert Ballard and Tielman Susato. Our experiments show that the nested technique used by some printers do not cause major problems. Though this might be an explanation why the results obtained on the prints by Ballard are more variable, the results obtained on prints by Susato (19 books) are the same as that obtained on the prints by Antonio Gardano (30 books) with an average recall rate of about 92% and an average accuracy of 86%.

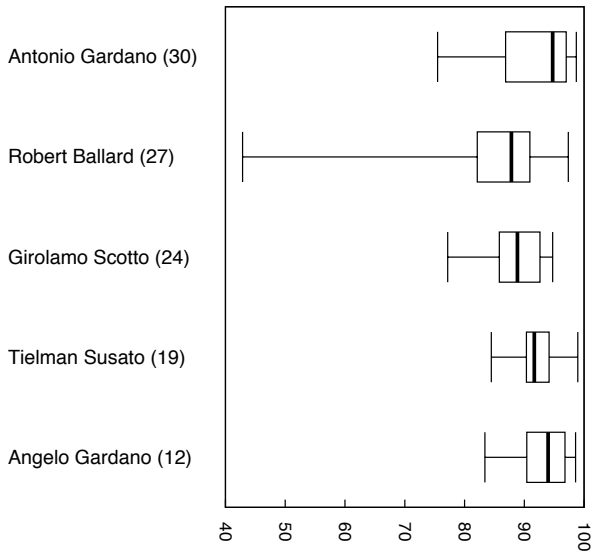


Figure 5. The quartiles for the prints with more than 10 books in the data set shows differences, albeit small ones. The results on the prints of Ballard are more variable and can be lower.

For the binarization algorithm, the results show that the choice made by the human evaluator before the processing of the book was appropriate in most cases. More precisely, the B2 binarization option gave better results than B1 in 80% of the cases where it was chosen by the evaluator. In the 21 cases where this was an inappropriate choice, it produced a recall rate significantly lower (by more than 5%) in one third of the cases. Figure 6 shows a page where B2 did improve the results and a case where it did not. In the first case, the recall rate was 69% with B1 and 91% with B2. In the second case, it was 97% with B1 but only 83% with B2, undoubtedly because the bleed-through is not marked enough.

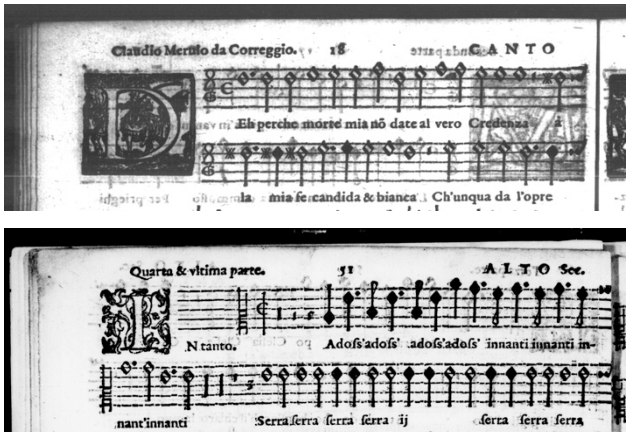


Figure 6. Two pages for which selecting the B2 binarization option did (above) and did not (below) improve the recognition results.

For the case where algorithm B2 improved the recognition results, we can clearly see looking at the binarized images shown on Figure 7 that B1 was not appropriate, in particular where the ornate letter printed on the other side of the page clearly appears through the paper.

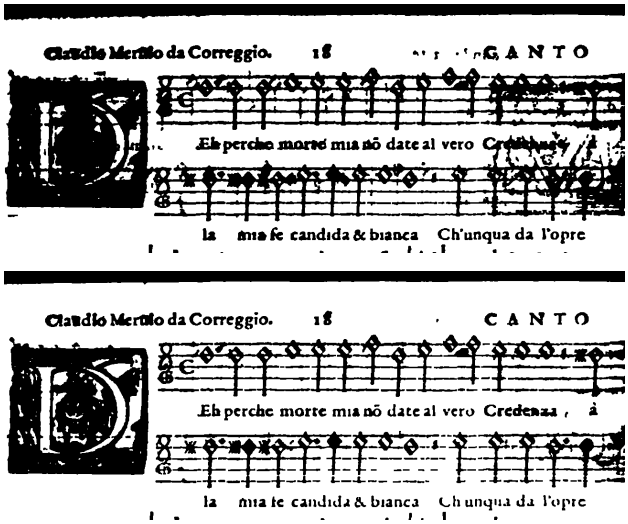


Figure 7. The binarization results with B1 (above) and B2 (below). The B2 option was clearly more appropriate for removing the bleed-through, in particular where the ornate letter is printed on the other side.

Correcting the results for this evaluation highlighted a handful of characteristics that evidently caused recognition errors. Prints with a strophic repertoire, mostly French chansons and lieder, with multi-line lyrics printed under the notes, often caused pre-processing imprecisions that then had an impact on the final recognition process. The problem with multi-line lyrics was that some of them were not properly recognized as text and the software thus attempted to read them as music symbols. Oblique ligatures (appearing on 6 pages) were inconsistently recognized. Other rarities that caused problems were passages with an unusual number of ledger lines (1 page) and the aforementioned handwritten slurs (1 page). In terms of document condition, the typical problems that locally affected the recognition process are: the presence of library stamps (2 pages), stains (4 pages), scratched-out notes (1 page), pages with extremely light inking (2 pages), or curvature on the edge of the page (9 pages). Additionally, in a few cases, because of very high bleed-through (5 pages), the accuracy of the corrected transcription used for the evaluation is questionable due to readability issues.

5. CONCLUSION AND FUTURE WORK

By evaluating OMR on the EMO collection, this paper sets a baseline for large-scale projects in the field. Our evaluation showed that about 80% of the collection can be processed and that we can expect a recall rate of be-

tween 85% and 100% for three quarters of the pages. In comparison, the Bibliothèque Nationale de France retained a minimal recognition rate of 60% for OCR results to be included as full-text [5]. This was based on an evaluation that showed this to be the limit for results to be useful for the user. Although searching text is different from searching music, and the usability of the data needs to be clearly estimated, our evaluation already shows an interesting potential for making the EMO or similar collections searchable. Furthermore, the evaluation does not take into account the improvement that can be obtained with adaptive OMR techniques that have proven to be extremely valuable in such cases [9].

The EMO collection is certainly an excellent example of what we can expect to find in library collections of similar size. Studying it was informative about which types of sources can be problematic. Whilst 20% of the sources had to be left out, there are other state-of-the-art tools that could be used to process some of them, in particular the lute tablature [6]. For others, specific modules would need to be developed, for example for the layout analysis of table books, while books such as the ones printed from engraved plates are certainly more problematic.

We can also notice that some categories that had to be left out are rather highly represented in EMO, quite likely because it contains only anthologies. If we compare this with the RISM A/I Series of individual prints, out of about 4,200 prints inventoried for the 16th century, around 3% are lute tablatures (10% in EMO), 0.6% are prints by Petrucci in single impression (3%) and only three prints by Verovio in engraving are inventoried (2%). Of course, this does not take into account the numerous re-editions to be found in the A/I Series. It does, however, show that the 20% exclusion part we ended up with in EMO is certainly an upper limit if 16th century prints are considered on a larger scale and 5% to 10% is a more likely percentage range on a collection with individual prints.

In terms of workflow, detecting which pages are missing music would be necessary for avoiding preparatory work. Experiments have already been conducted on this [1] and the SISSMA project is currently working on similar issues. Our experiments have also shown that strophic music was not always properly handled in the preprocessing. This confirms it is always worth improving the layout analysis because it is a critical step in the process. For the binarization, having the user select the binarization algorithm for a book is acceptable because it is a quick task. The cases where B2 was a bad choice could certainly have been avoided with better training of the user. However, user-selected binarization is not optimal because it assumes the degradation to be identical throughout the book, which is of course not always the case. Having the most appropriate binarization algorithm selected automatically would be a valuable improvement. Future work will also include reassembling the parts for reconstructing the score, which will open the door to polyphonic processing of the data.

6. REFERENCES

[1] D. Bainbridge, and T. Bell: "Identifying music documents in a collection of images," in *Proceeding of the ISMIR 2006 Conference*, pp. 47–52.

[2] H. Balk, and L. Ploeger: "IMPACT: Working together to address the challenges involving mass digitization of historical printed text," *OCLC Systems & Services*, 25(4), pp. 233–248, 2009.

[3] J. Bernstein: *Print Culture and Music in Sixteenth-Century Venice*, Oxford, Oxford University Press, 2001.

[4] J. A. Burgoyne, L. Pugin, G. Eustace, and I. Fujinaga, "A comparative survey of image binarisation algorithms for optical recognition on degraded musical sources," in *Proceedings of the ISMIR 2007 Conference,* pp. 509–12.

[5] G. Cron: "Corpus." Presentation, Journée IMPACT, BnF, Paris, September 19, 2011.

[6] C. Dalitz, and C. Pranzas: "German Lute Tablature Recognition," in *Proceedings of the ICDAR 09*, pp. 371–375.

[7] A. Hankinson, J. A. Burgoyne, G. Vigliensoni, and I. Fujinaga: "Creating a large-scale searchable digital collection from printed music materials," in *Proceedings of the 21st ACM Conference on World Wide Web,* pp. 903–908.

[8] L. Pugin, J. A. Burgoyne, and I. Fujinaga: "Goal-directed evaluation for the improvement of optical music recognition on Early music prints," in *Proceedings of the ACM-IEEE JCDL 2007*, pp. 303–04.

[9] L. Pugin, J. A. Burgoyne, and I. Fujinaga, "MAP adaptation to improve optical music recognition of early music documents using hidden Markov models," in *Proceedings of the ISMIR 2007 Conference,* pp. 513–16.

[10] L. Pugin, J. Hockman, J. A. Burgoyne, and I. Fujinaga, "Gamera versus Aruspix: Two optical music recognition approaches," in *Proceeding of the ISMIR 2008 Conference,* pp. 419–24.

[11] A. Rebelo, I, Fujinaga, F. Paszkiewicz, A. R. 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), pp. 173–190, 2012.

[12] S. Rose: "Early Music Online," *Early Music Performer*, 30, pp. 22–25, 2012.

[13] V. Viro: "Peachnote: Music score search and analysis platform," in *Proceedings of the ISMIR 2011 Conference,* pp. 359–362.

SPARSE MUSIC DECOMPOSITION ONTO A MIDI DICTIONARY DRIVEN BY STATISTICAL MUSIC KNOWLEDGE

Boyang Gao, Emmanuel Dellandréa and Liming Chen
Université de Lyon, CNRS,
Ecole centrale de Lyon, LIRIS, UMR5205, F-69134, France.
E-mail: {Boyang.Gao, Emmanuel.Dellandrea, Liming.Chen}@ec-lyon.fr

ABSTRACT

The general goal of music signal decomposition is to represent the music structure into a note level to provide valuable semantic features for further music analysis tasks. In this paper, we propose a new method to sparsely decompose the music signal onto a MIDI dictionary made of musical notes. Statistical music knowledge is further integrated into the whole sparse decomposition process. The proposed method is divided into a frame level sparse decomposition stage and a whole music level optimal note path searching. In the first stage note co-occurrence probabilities are embedded to generate a sparse multiple candidate graph while in the second stage note transition probabilities are incorporated into the optimal path searching. Experiments on real-world polyphonic music show that embedding music knowledge within the sparse decomposition achieves notable improvement in terms of note recognition precision and recall.

1. INTRODUCTION

Large amounts of digitalized music available drive the need for the development of automatic music analysis, for example automatic genre classification, mood detection and similarity measurement. Most of the tasks rely on effective features extracted from music signals. Among various features, music notes, denoted by MIDI notes in this paper, provide the most comprehensive information, since music is indeed sound poetry comprised of notes played by instruments. If notes are accurately recovered from music signal, automatic music analysis can be greatly improved. However, mixing different instrument playing is trivial while decomposing is quite challenging due to the intrinsic complexity of polyphonic music.

Recovering notes from a music wave signal is usually referred to multiple F0 estimation. The approaches in literature can be roughly sorted into two categories: parameterized like statistical model based methods and non-parameterized like non-negative matrix factorization (NMF) based methods. Parameterized approaches usually assume that multiple F0 can be described by particular models with a small number of free parameters that can be estimated from the signal. For example, in [1] Kameoka et al. propose a multi-pitch analyzer named the harmonic temporal structured clustering (HTC) method that jointly estimates pitch, intensity, onset and duration. HTC decomposes the power spectrum time series into distinct clusters such that each cluster has originated from a single source modeled by a Gaussian Mixture Model (GMM). The parameters of the source model are computed thanks to maximum a posteriori (MAP) estimation. In [2], Wu et al. extend Kameoka's work to propose a flexible harmonic temporal timbre model to decompose the spectral energy of the signal in the time-frequency domain into individual pitched notes. Each note is modeled with a 2-dimensional Gaussian kernel. Parameters of Gaussian mixtures are then estimated by expectation maximization (EM) algorithm with a global Kullback–Leibler (KL) divergence cost function.

Unlike parameterized approaches, non-parameterized methods like NMF focus on recovering pitch combinations from the signal data itself without presuming any underlying model forms. For example, NMF [3] based methods try to decomposes the multiple pitch spectrum matrix X into two matrices W and H [4]. W contains various harmonic patterns and H consists of activation behaviors so that $X = WH$. In [5], Hoyer extends the original NMF by adding a regulation term to make H sparse. Sparseness property is quite helpful especially for music note estimation, since a short period music can only contain a few notes played together, compared with all possible notes.

NMF is such an extensible framework that it largely dominates non-parameter methods. For example, in [6] Zafeiriou adds a linear discriminant analysis (LDA) stage to the activities extracted by NMF. In [6-8], fisher-like discriminant constraints are embedded inside the decomposition. In [9], Lewandowski proposes a supervised method with two discriminative criteria that maximize inter-class scatter and quantify the predictive potential of a given decomposition. In order to extract features that enforce the separability between pitch labels, pitch information present in time-aligned musical scores is fused in sparse NMF. In [10], Sakaue combines Bayesian inference with NMF to propose a Bayesian non-negative harmonic-temporal factorization (BNHTF). BNHTF models the harmonic and temporal structures separately with Gaussian mixture models. In [11], a music sparse decomposition approach is proposed using high quality MIDI

dictionary. This work is a variant of sparse NMF and uses non-negative matching pursuit to solve sparse NMF. Unlike NMF that processes the entire signal, this work constructs the activity matrix H column by column. It is still worth mentioning the work in [12] where Leveau et al. propose to learn instrument specified note atoms with a modified matching pursuit and a tracking of the played instrumental notes by searching an optimal path with respect to the reconstruction error.

Thus, previous works in the literature have demonstrated the effectiveness of various approaches in multiple-F0 estimation, especially NMF based methods. However, under the NMF framework, the entire music spectrum series X are treated as a whole object to be reconstructed. Most of the algorithms focus on reducing the spectrum reconstruction error so as to overlook the compatibility in concurrent and consecutive notes. This batch processing style makes it hard to fuse note co-occurrence and transition information to guide note detection during the matrix factorization. Indeed, after the signal spectrum matrix is factorized, W and H are new represents of the music, which have lost signal context information for post-processing to correct possible error. Even in [12], the Viterbi algorithm is used to search the optimal path only with respect to a minimum reconstruction error and neglects the underlying note relations. Nevertheless correlation between concurrent and consecutive notes contains significant heuristics that can help to correct the decomposition error introduced by a signal level analysis.

Therefore, to employ note statistical information to help music sparse decomposition, we propose in this paper a two-stage sparse decomposition approach integrated with music knowledge. In frame level decomposition stage, note co-occurrence probabilities are embedded to guide atom selection in modified matching pursuit algorithm with a MIDI dictionary. A sparse multiple candidate graph is then constructed to provide backup choices for later selections. In the global optimal path searching stage, note transition probabilities are incorporated together with a goodness measure of frame decomposition. Its principle is to guide the local sparse music decomposition with co-occurred notes information and decode the global optimal decomposition path with consecutive note knowledge. Due to the Gabor limit, time and frequency resolution cannot be well satisfied at the same time. Thus, we emphasize the frequency resolution aspect rather than the exact time location, since correct note recognition is more important for our following classification task.

The rest of this paper is organized as follows: Section 2 introduces our two-stage approach in detail. Section 3 shows experimental results on real-world music signals. The conclusion is drawn in the final section.

2. SPARSE DECOMPOSITION WITH NOTE STATISTICS

Our proposed method consists of two main steps. In the first step, the entire music signal is framed and a modified orthogonal matching pursuit algorithm is performed on each frame to generate decomposition candidates. In the second step, decomposition candidates are connected to form a directed graph. An optimal path is then constructed to produce the final decomposition result.

2.1 Frame Level Sparse Decomposition

Former study shows that elaborating an appropriate musical dictionary is a key issue since this set of atoms has to be rich enough to characterize the varieties of real word music. Although [12] has developed sophisticated method to learn atoms from instrument recordings, it is still impractical to apply to a large instrument set. Therefore, in order to get adequate instrument note sounds, we propose to make use of a MIDI synthesizer. Logic Pro 9 is employed in our approach to generate the MIDI note dictionary because of its huge instrumental library and the high sound quality. Unlike pre-installed MIDI synthesizer with sound card, Logic Pro 9 uses a large number of real instrument recordings to make synthesized wave signal as natural as possible.

To build the MIDI dictionary, we choose the first 80 realistic instruments as in general MIDI level 1 set and 31 percussion instrument sets including 1860 percussion sounds. For each instrument, we keep 60 notes from note 31 to note 90. The 60 notes span 5 octaves from low to high, covering most instrumental playing range. The duration of each note is set to 186ms, which are 4096 samples under a sample rate of 22050Hz. This duration is long enough to hold one attack-decay-sustain-release (ADSR) envelope and leads to 5.38Hz in terms of frequency resolution, which is sufficient to discriminate adjacent notes in piano roll. Our MIDI note wave is then converted into a single-sided power spectrum obtained by applying the short time Fourier transform (STFT) with a Hamming window. The final MIDI dictionary thus contains 6660 2048-dimensional vectors.

The adoption of the sparse representation is based on the hypothesis that during a 186ms time slot, there will not be many notes played together. Therefore, concurrent notes are sparse within one frame. Sparse representation [13] is originated from finding the solution x^* of an underdetermined linear system $Dx = y$ so that x^* contains as few non-zero components as possible. In most cases $Dx = y$ is hard to satisfy, thus in practice $\|x\|_0$ and $\|Dx - y\|_2$ are minimized simultaneously instead.

Armed with our MIDI dictionary, the classical matching pursuit algorithm like orthogonal matching pursuit (OMP) [17] must be modified because the single-sided power spectrum words in MIDI dictionary impose an inherent positive constraint on sparse solutions. In other words, any negative component of a sparse solution is prohibited, as negative appearance of certain notes is impossible. To solve this problem we adopt a positive constraint matching pursuit (PCMP) algorithm that is mentioned in [11][14]. The difference between OMP and

PCMP is in updating a provisional solution step: for OMP, least mean square (LMS) suffices to solve the minimization resulting in residual signals orthogonal to support set. For PCMP, however, after the positive constraint minimization, orthogonality is not always guaranteed, thus the algorithm is turned to a weak orthogonal matching pursuit.

When scrutinizing the decomposition results of PCMP within one frame, we found a number of irregular note combinations. This is due to PCMP's over-fitting target signals without considering any compatibility of concurrent notes. In fact, atom selection in each iteration of orthogonal matching pursuit algorithm is very important. OMP guarantees that expending support set with any linear independent atoms will decrease the reconstruction error and at the same time keep the residual signal orthogonal to the new expanded support set. Any atom selected in the support set will permanently reside. Therefore previously selected atoms have a great influence on following ones and alter the overall OMP performance. Although PCMP does not always hold orthogonal property, the principle remains the same.

Selecting a new atom in dictionary is thus the very place where concurrent note heuristic information should be embedded. To formulate concurrent note information, Bayes model is employed in our approach to approximate the posterior probability of potential note given observed notes

$$P(X|\boldsymbol{O}) = \frac{\prod_{i=1}^{N} P(O_i|X)P(X)}{\sum_Y \prod_{i=1}^{N} P(O_i|Y)P(Y)} \quad (1)$$

where $\boldsymbol{O} = \{O_1, O_2 \cdots O_N\}$ denotes N observed notes obtained by first N PCMP iterations, X represents a potential co-occurred note with $\boldsymbol{O}$. The note prior probability $P(X)$ and the note co-occurrence posterior probability $P(X|Y)$ are estimated from our classical music MIDI database. To obtain $P(X|Y)$, a joint distribution $P(X,Y)$ is firstly estimated by accounting the frequency with overlap degree of the concurrent note X and Y. Then $P(X|Y)$ is obtained by normalizing $P(X,Y)$ over Y. Although equation (1) provides instructive information to help select appropriate note combinations, it is still risky to only consider the best note decomposition, since the second best one may be more appropriate in adjacent note context. To avoid the one best bias, we propose to preserve multiple candidates to give top-N best decompositions chances to recover in optimal path searching.

Orthogonal matching pursuit is a greedy algorithm. In each iteration only the best atom will be added into support set. This can be risky in some cases, since once a "bad" atom is selected, this error cannot be corrected in the future. In [15], it has been shown that it is possible to select "bad" atom initially so as to trap OMP from reconstructing target signals. Methods like OCMP in [19] are proposed to overcome the problem. However, in music decomposition the same note in different octave or from the same kind of instruments shares the similar harmonic pattern. Therefore it is hazardous to rule out a suboptimal decomposition too early before adjacent note compatibility is checked.

To overcome this drawback of OMP, we propose to keep N best candidates in each iteration instead of only one. To measure the goodness of frame decomposition we define $goodness = \alpha \log(P(\boldsymbol{X})) - \|\boldsymbol{r}\|_2^2$, where $\boldsymbol{X}$ is sparse note decomposition vector, $\boldsymbol{r}$ is decomposition residual signal, $P(\boldsymbol{X}) = \sum_Y \prod_{i=1}^{N} P(X_i|Y)P(Y)$ denotes note concurrent probability, α is a free parameter that balances concurrent probability term and reconstruction error term.·As an example shown in Figure 1 we keep the top 3 decomposition candidates in every iteration. In the first iteration (C), (E), (G) are kept. In the second iteration, (C, D), (E, F) and (E, G) are obtained according to the reconstruction error and concurrent probability. Note that (G) selected in the first iteration is eliminated because its descendant combinations (G,*) are inferior to others'. After 3 iterations, combinations of (C, D, E), (C, D, G) and (E, F, B) survive, as shown in orange.

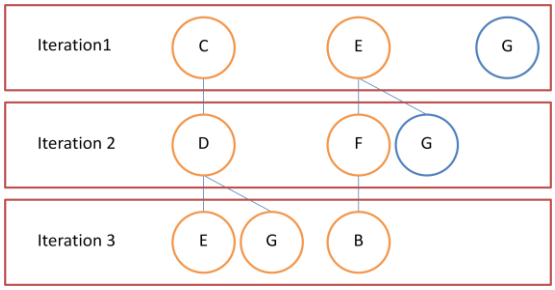


Figure 1. Multiple candidate selection example

When sparse decomposition terminates, the top N note candidates are derived for every signal frame. The best one can be treated as the decomposition result of the current frame. Besides, all candidates are preserved for constructing the optimal decomposition path when we further investigate inter-frame relations. The multiple candidate PCMP algorithm that we propose is summarized in Algorithm 1.

2.2 Global Level Optimal Note Path Searching

All previous steps in section 2.1 focus on improving sparse note decomposition within one signal frame. When further scrutinizing the PCMP decomposition between consecutive frames, we can still find a number of discontinuous note decompositions, in which the note sequence has sudden abnormal jumps in adjacent frames, including octave shift or sharp/flat drift. This is due to a lack of note transition regulation and because the sparse decomposition only minimizes reconstruction error in current frame without considering any neighbor frame contexts.

Besides the co-occurred ones, consecutive notes bear strong correlations which convey various melody, temporal and dynamic information of music. It is reasonable to incorporate such sequential knowledge of notes as to suppress the discontinuous note error.

Task: Approximate the solution of problem: $min_{\boldsymbol{x}}\|\boldsymbol{x}\|_0$, subject to $\boldsymbol{Ax} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}$.
Input: Dictionary $\boldsymbol{A}$, signal $\boldsymbol{b}$, max iteration number I, top N candidates to keep, balance parameter α, note posterior probability $P(X|Y)$ and error threshold ϵ_0.
Output: sparse solution: $\boldsymbol{x}^*$

Initialization:
Initial residual: $\boldsymbol{r}_c^0 = \boldsymbol{b}$.
Initial support: $S_c^0 = \emptyset$.
Initial candidate queue: $Q^0 = \{S_1^0, S_2^0 \cdots S_N^0\}$

Main iteration:
for $k = 1$ to I
 for $c = 1$ to N
 - Compute $P(j|S_c^{k-1})$ according to equation (1)
 - Compute error: $\epsilon(j) = \min_{z_j}\left\|\boldsymbol{a}_j z_j - \boldsymbol{r}_c^{k-1}\right\|_2^2 - \alpha \cdot \log\left(P(j|S_c^{k-1})\right)$ all j using the optimal choice $z_j^* = \boldsymbol{a}_j^T \boldsymbol{r}_c^{k-1} / \left\|\boldsymbol{a}_j\right\|_2^2$.
 - Find top N minimizers of $\epsilon(j)$ to form $J = \{j_1, j_2 \cdots j_N\}$ such that $J \cap S_c^{k-1} = \emptyset$, and push $S^k = \left\{S_c^{k-1} \cup \{j_1\}, S_c^{k-1} \cup \{j_2\} \cdots S_c^{k-1} \cup \{j_N\}\right\}$ into Q^k

 end
 for each $S_i^k \in Q^k$
 - Compute $\boldsymbol{x}$ that minimizes $\|\boldsymbol{Ax} - \boldsymbol{b}\|_2^2$ subject to $support\{\boldsymbol{x}\} = S_i^k, \boldsymbol{x} \geq \boldsymbol{0}$.
 - Compute residual $\boldsymbol{r}_i^k = \boldsymbol{b} - \boldsymbol{Ax}$.

 end
 Ascendingly sort Q^k according to $\left\|\boldsymbol{r}_i^k\right\|_2^2 - \alpha \cdot \log\left(P\left(S_i^k\right)\right)$ and keep the first N items.
 If any $\left\|\boldsymbol{r}_i^k\right\|_2^2 < \epsilon_0$ break.
end
Output result: $Q^* = Q^k$.

Algorithm 1. Positive constraint matching pursuit producing multiple candidates

We thus apply transition probabilities to model relations between two decomposition candidates in adjacent frames. To formulate note transitions, Bayes model is adopted so that conditional probability can be approximated from individual note pairs. Since at most N candidates remain in one frame the posterior probability of candidate j in frame t given candidate i in frame $t-1$ is calculated as

$$P\left(\boldsymbol{X}_t^{(j)} \middle| \boldsymbol{X}_{t-1}^{(i)}\right) \stackrel{\text{def}}{=} \frac{\prod_{k=1}^{D} P\left(X_{k,t}^{(j)} \middle| \boldsymbol{X}_{t-1}^{(i)}\right)}{\sum_l \prod_{k=1}^{D} P\left(X_{k,t}^{(l)} \middle| \boldsymbol{X}_{t-1}^{(i)}\right)} \quad (2)$$

where $\boldsymbol{X}_t^{(j)} = \{X_{1,t}^{(j)}, X_{2,t}^{(j)} \cdots X_{D,t}^{(j)}\}$ denotes the decomposition candidate j in frame t containing D notes. $P\left(X_t | \boldsymbol{X}_{t-1}\right)$ is calculated similarly as in equation (1).

Thanks to the multiple decomposition candidates generated by the modified PCMP previously, an inter-decomposition directed graph is further constructed to help determining the optimal decomposition path through all frames, as illustrated in Figure 2. In this directed graph, each decomposition candidate forms a node and outgoing edge denotes the transition probability computed by equation (2). The nodes are disconnected within the same frame indexed with t.

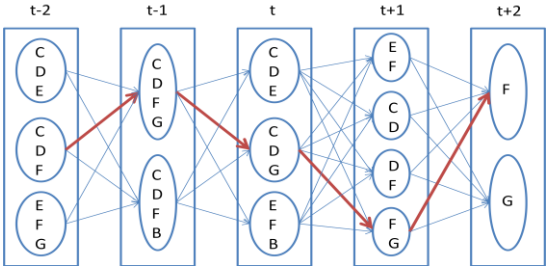


Figure 2. Optimal path decoding example

In order to connect transition probabilities with the sparse decomposition candidates, the decomposition goodness measure is converted into corresponding probabilities as $P(\boldsymbol{X})^{\alpha} \cdot \exp(-\|\boldsymbol{r}\|_2^2)$, since the frame signal $\boldsymbol{b}$ and atoms in $\boldsymbol{A}$ have been normalized to unit vectors. The conversion also reflects a reasonable assumption that the reconstructed signal is approximately Gaussian distributed around original one. Treating the decomposition candidates as hidden states of a Hidden Markov Model (HMM), Viterbi algorithm decodes the optimal decomposition candidate path $\boldsymbol{p}^*$:

$$\boldsymbol{p}^* = \underset{\boldsymbol{p}}{\text{maxarg}} \prod_{t=1}^{F} P(\boldsymbol{X}_{p_t}^{(t)})^{\alpha} \, P\left(\boldsymbol{X}_{p_t}^{(t)} \middle| \boldsymbol{X}_{p_{t-1}}^{(t)}\right)^{\beta} e^{-\left\|\boldsymbol{r}_{p_t}^{(t)}\right\|_2^2} \quad (3)$$

where t is the frame index, F is the total number of frames, β is a balance parameter to adjust emphasis, p_t denotes decomposition candidate index in frame t along path $\boldsymbol{p}$. Initially $P(\boldsymbol{X}_{p_1}^{(1)} | \boldsymbol{X}_{p_0}^{(1)})$ =1.

3. EXPERIMENT AND RESULTS

To evaluate the decomposition quality of the proposed PCMP with note statistics, a multi-timbral music with time domain note reference has been used, which is provided in Mirex2007 multiF0 development data [16]. The music is a recording of the fifth variation from *L. van Beethoven Variations from String Quartet Op.18 N.5*, lasting for 54s. 5 instruments are included into the music. Each instrument was recorded separately and then mixed to a mono 44.1 kHz 16 bits wave file. The whole music is tested by our system (PCMP with multi-candidate and Viterbi) against its ground truth MIDI file.

Another widely used dataset adopted in our experiments is MUS, provided in MAPS [18]. MUS contains 270 pieces of classical and traditional music, recorded in different conditions which vary in piano instruments and surroundings. For each piano music piece, as in [9], first 30 seconds are tested by our system against ground truth MIDI files.

Figure 3 shows precision and recall scatter diagram of the proposed decomposition that improve original PCMP, noted as PCMPMC and PCMPMCV. Table 1 displays the

comparisons in terms of precision, recall and F-measure between our proposed method and the state of the art results. F-measure is defined as the harmonic mean of precision and recall. Statistic of note recognition precision and recall has been made upon consecutive 186ms frames. For ground truth MIDI, if 70% of some note lies in the frame the note is accounted and there is no frequency tolerance. Threshold for drawing the diagram is imposed on sparse solution vector in each frame to filter insignificant note detection according to its sparse solution value. Different thresholds result in scatter points in Figure 3. Two free parameters α and β are set to 0.8 and 1.3 to balance reconstruction error and note statistics.

	Prec. (%)	Rec. (%)	F-meas. (%)
NMF[4]	41.1	46.6	45.3
HTC[1]	57.4	51.3	54.2
JHT[2]	**59.7**	61.4	**60.5**
PCMPMCV	51.8	**72.0**	60.3

Table 1. Average multiple pitch estimation performance on MIREX2007 dataset.

From Figure 3 we can see that when co-occurrence note information is integrated into PCMP, the precision increases about 6% while recall increases by 2%~3%. When the note transition information is fused and the optimal path decoding is applied, the precision and recall are further improved by 5% and 2% approximately. From Table 1 and Figure 3 we can find that if no threshold is imposed on the sparse solution of PCMPMCV, 72% of the notes can be recalled while the precision is 51.8% resulting in an F-measure of 60.3 %. The recall of our best configuration outperforms state of the art result in [2] by more than 10% while the precision is 8% lower, resulting in an F-measure 0.2% lower than that reported in [2].

	Prec. (%)	Rec. (%)	F-meas.(%)
Spectral constraints [20]	71.6	65.5	67.0
Isolated note spectra [20]	68.6	66.7	66.0
DNMF-LV[9]	68.1	65.9	66.9
DNMF-AE[9]	66.8	68.7	67.8
SONIC[21]	**74.5**	57.6	63.6
PCMPMCV	60.7	**77.3**	**68.0**

Table 2. Average multiple pitch estimation performance on MUS dataset.

Table 2 shows the precision, recall and F-measure results on MUS data set. All parameters and setups are the same as used in previous experiment except for the conditional probability estimation. In this experiment the rest data other than first 30 seconds are used to estimate conditional probabilities $P(X|Y)$. From Table 2 we can observe that the proposed approach achieves the highest recall and F-measure of 77.3% and 68%, although obtains the lowest precision of 60.7%.

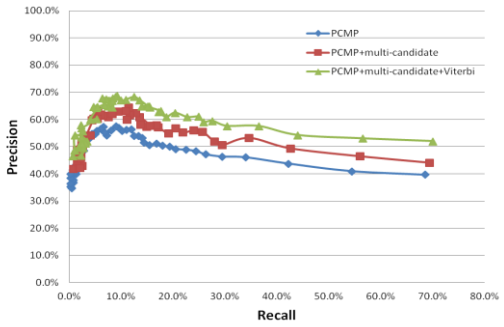


Figure 3. Note precision vs. recall of the two improvements

From the two experiments, we can find that with statistical musical knowledge sparse decomposition is improved in terms of both precision and recall. The proposed approach tends to obtain superior recall and F-measures but lower precisions compared with variant NMF and other methods. Higher recall means the more information is preserved in the decomposition results. Since our final aim of the decomposition is to provide decent features for music classifications, the performance of our system is actually preferred. Our higher recalls and F-measures are attributed to the quality of MIDI dictionary as well as statistical music knowledge fused in sparse decomposition. Longer analysis window is another important factor.

When comparing decomposition with the ground truth, we found numbers of instrument errors even with correct note detections, which is likely caused by mismatches between the MIDI dictionary and the real-world data. In some cases concurrent and transition probability of notes can even make incorrect compensation to original PCMP, which is probably due to the limitation of the naive Bayes model. To overcome these drawbacks, dictionary adaptation techniques and sophisticated graphical models will be proposed and investigated in our future work.

4. CONCLUSION

We have proposed in this paper a novel sparse music decomposition approach driven by music knowledge. It employs note statistical information to improve sparse decomposition with a MIDI dictionary. In the frame level, music signals are decomposed onto a MIDI dictionary with a note co-occurrence heuristic. Transition probabilities are then computed between adjacent decomposition candidates through the whole frame sequence. The final optimal decomposition path is then constructed by the Viterbi algorithm. Experimental results show that embedding concurrent note statistics in PCMP and applying a note sequence heuristic allows improving the note recognition precision and recall.

5. ACKNOWLEDGMENT

This work is partly supported by the French ANR under the project VideoSense ANR-09-CORD-026.

6. REFERENCES

[1] H. Kameoka, T. Nishimoto, and S. Sagayama, "A multi-pitch analyzer based on harmonic temporal structured clustering," *IEEE Transactions on Audio, Speech, and Language Processing* , vol. 15, no. 3, pp. 982–994, 2007.

[2] J. Wu, E. Vincent, S. A. Raczynski, T. Nishimoto, N. Ono, and S. Sagayama, "Multipitch estimation by joint modeling of harmonic and transient sounds," in *Proceedings of 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 25–28, 2011.

[3] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, pp. 556-562, 2001.

[4] S. A. Raczynski, N. Ono, and S. Sagayama, "Multipitch analysis with harmonic nonnegative matrix approximation," in *Proceedings of 8th International Conference on Music Information Retrieval (ISMIR)*, 2007.

[5] P. O. Hoyer, "Non-negative sparse coding," in *Proceedings of 12th IEEE Workshop on Neural Networks for Signal Processing*, pp. 557– 565, 2002.

[6] S. Zafeiriou, A. Tefas, I. Buciu, and I. Pitas, "Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 683–695, 2006.

[7] N. Guan, D. Tao, Z. Luo and B. Yuan, "Manifold regularized discriminative nonnegative matrix factorization with fast gradient descent," *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 2030–2048, 2011.

[8] Y. Wang, Y. Jia, C. Hu and M. Turk, "Fisher non-negative matrix factorization for learning local features," in *Proceedings of Asian Conference on Computer Vision (ACCV)*, 2004.

[9] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Discriminative non-negative matrix factorization for multiple pitch estimation," in *Proceedings of 13th International Conference on Music Information Retrieval (ISMIR)*, 2012

[10] D. Sakaue, T. Otsuka, K. Itoyama, and H.G. Okuno, "Bayesian non-negative harmonic-temporal factorization and its application to multipitch analysis," in *Proceedings of 13th International Conference on Music Information Retrieval (ISMIR)*, 2012.

[11] B. Gao, E. Dellandréa, and L. Chen, "Music sparse decomposition onto a midi dictionary of musical words and its application to music mood classification," in *Proceedings of 10th IEEE International Workshop on Content-Based Multimedia Indexing (CBMI)*, pp. 1‑6, 2012.

[12] P. Leveau, E. Vincent, G. Richard and L. Daudet, "Instrument-specific harmonic atoms for mid-level music representation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 1, pp. 116-128, 2008.

[13] M. Elad, *"Sparse and Redundant Representations from Theory to Applications in Signal and Image Processing,"* Springer-New York, 2010.

[14] A.M. Bruckstein, M. Elad, and M. Zibulevsky, "Sparse non-negative solution of a linear system of equations is unique," in *Proceedings of 3rd IEEE International Symposium on Communications, Control and Signal Processing (ISCCSP)*, pp. 762–767, 2008.

[15] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[16] http://www.music-ir.org/mirex/wiki/2007:Multiple_Fundamental_Frequency_Estimation_%26_Tracking

[17] Y. C. Pati, R. Rezaiifar, P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition." In *Proceedings of IEEE Signals, Systems and Computers*, vol. 1, pp. 40-44, 1993

[18] 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*, vol. 18, no. 6, pp. 1643-1654, 2010.

[19] G. Rath, and C. Christine, "A complementary matching pursuit algorithm for sparse approximation," in *Proceedings of European Signal Process. Conf.*, Lausanne, Switzerland. 2008.

[20] E. Vincent, N. Bertin, and R. Badeau, "Adaptive harmonic spectral decomposition for multiple pitch estimation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3:pp. 528–537, 2010.

[21] M. Marolt, "A connectionist approach to automatic transcription of polyphonic piano music," *IEEE Transactions on Multimedia*, vol. 6, no. 3, pp. 439–449, 2004.

ANNOTATING WORKS FOR MUSIC EDUCATION: PROPOSITIONS FOR A MUSICAL FORMS AND STRUCTURES ONTOLOGY AND A MUSICAL PERFORMANCE ONTOLOGY

Véronique Sébastien, Didier Sébastien, Noël Conruyt
IREMIA - Laboratoire d'Informatique et de Mathématiques, EA2525
University of Reunion Island, Saint-Denis, Réunion (FRANCE)
veronique.sebastien/didier.sebastien/noel.conruyt@univ-reunion.fr

ABSTRACT

Web applications and mobile tablets are changing the way musicians practice their instrument. Now, they can access instantaneously thousands of musical scores online and play them while watching their tablet, put on their music stand. However musicians may have difficulties in getting appropriate tips and advice to play the chosen piece correctly. This is why we conceived a collaborative platform to annotate digital scores on tablets in previous work. However, we noticed that the current Music Ontology (MO) do not allow to tag these annotations appropriately. Thus, we present in this paper a proposition for a Musical Forms and Structures Ontology (MFSO) and a Musical Performance Ontology (MPO) based on music practice. A construction methodology and a model are first detailed. Then, a practical use case is presented. Lastly, inherent theoretical and practical difficulties encountered during the ontology framework's conception are discussed.

1. INTRODUCTION

More and more musicians share their scores and performances on dedicated Web platforms such as free-scores.com or musescore.com. Meanwhile music applications dedicated to scores management are ported to tablet devices (Tonara™, Musescore™, Finale Songbook™). However, musicians still do not dispose of appropriate tools to demonstrate their know-how on these scores: how to play this difficult part? Which fingering should I use? This is why we designed a collaborative score annotation service working on tactile tablets [1]. It allows users to illustrate abstract scores with multimedia content showing tips, exercises or questions directly linked to the concerned notes on the score. We also proposed a matching analyzer to automatically determine a score difficulty level [2]. But in order to suggest relevant annotations to performers, we need to tag the latter appropriately. Indeed, musical know-how is contextual: it relates to a specific piece with its structure and mood. But most techniques can be reused in similar contexts with appropriate adaptations. Thus, correctly contextualized annotations could be reused on different pieces sharing similarities (genre, composer, patterns, etc.). It would also enable complex queries on instrumental issues (for instance: how to play scales? how to produce a soft but expressive sound? what is the best strategy to learn a piece by heart?).

We could rely on social tags created by users (i.e., Folksonomies [3]). However, as noted by Sordo in his experiment on musical genres and moods [4], the emerging vocabulary is not always reliable, especially on very specialized terms. Considering our educational context, reliability and accuracy are essential. These led us to a controlled vocabulary based solution, with extension and adaptation possibilities according to the considered case.

To do so, we propose to extend the existing Music Ontology (MO) [11] with a Musical Forms and Structures Ontology (MFSO) and a Musical Performance Ontology (MPO), through a global Semiotic Annotation framework (SA) (Figure 1). While the MO is dedicated to musical resources and events description for databases, the MFSO focuses on musical works structure analysis, and the MPO on performances and instrumental techniques. However, if the MO and MFSO can be used to relate objective facts about music, the MPO deals with subjective approaches of a given piece. This is why we embed it in a SA. We explain and justify the use of this Sign-based framework in the next section. We then present our ontological propositions in the third and fourth section. A simple use case is detailed in the fifth section. Lastly, we discuss the difficulties we encountered in our attempt to organize musicological and practical instrumental terms.

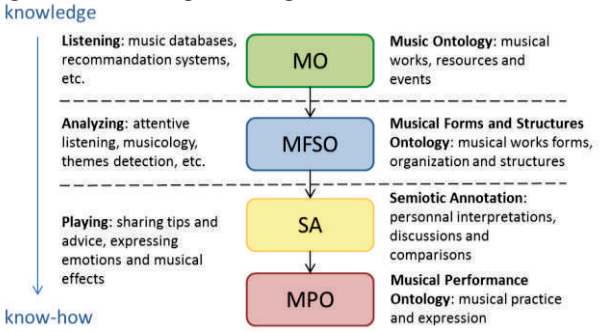


Figure 1: Musical Ontologies global organization and roles.

2. SEMIOTIC ANNOTATIONS ON WORKS EXTRACTS

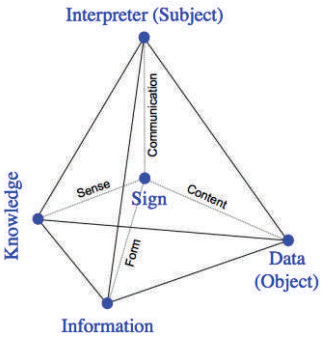


Figure 2: Sign tetrahedron.

Music, as any other artistic field, deals more with interpretations than with formal knowledge. What makes it interesting is that different artists produce different performances, somehow recognizable, even if they play the same piece. But explaining these differences and how to produce them is a delicate task. Common formal representations are not suited to do so for two main reasons. The first one is that textual descriptions cannot really convey emotions and gestures, even if they are important for search engines and knowledge bases. The second one is that they are not adapted to share subjective interpretations. For instance, the comment "I think that the 53421 fingering is more supple for small hands" is difficult to represent accurately into a machine language for the time being.

To overcome these issues, we propose to manage Signs rather than Knowledge. A Sign is a subjective communication object composed of a *content* (images, gestures, writings, sounds), its *form* (structure, organization, context) and its *sense* (interpretation, meaning) from a *subject* point of view at a given time [5]. For instance, a symbol is a particular type of graphical Sign, i.e. a form shared among a group of people. But a Sign can also be a simple gesture (e.g., a nod), with various meanings according to its context (historical, or cultural). In artistic fields, where practical know-how is essential, Signs are essential to communicate different interpretations. To manage Signs in an information system, we link each of their components to a digital element. Contents can be embedded in multimedia *data* (audio, video), form can consist in its contextual *information* (metadata, localization, selection), and sense consists in *knowledge*, represented by a textual comment or a semantic description for the machine (i.e., tags from a structured vocabulary). As shown on Figure 2, the Sign object can be represented as a tetrahedron [5].

We define a Semiotic Annotation (SA) framework to capture these Signs on collaborative annotation platforms. Figure 3 presents the proposed model for Semiotic Annotation. To insure its integration to the current Semantic Web, the SA model is linked to top-level ontologies. The FRBR ontology allows addressing creative works and their different parts. Indeed, to create contextualized annotations, we need to address precise parts of the discussed *Work*. A *WorkPart* can be any entity which is a part of another entity. We note that a *Work* or a *WorkPart* exists independently from its different *Expressions* and *Manifestations*. For instance, the first *Verse* (*WorkPart* resource) in the *Frères Jacques* canon (*Work* resource) can be discussed without reference to a particular performance of the tune (*Expression*), recorded on a particular

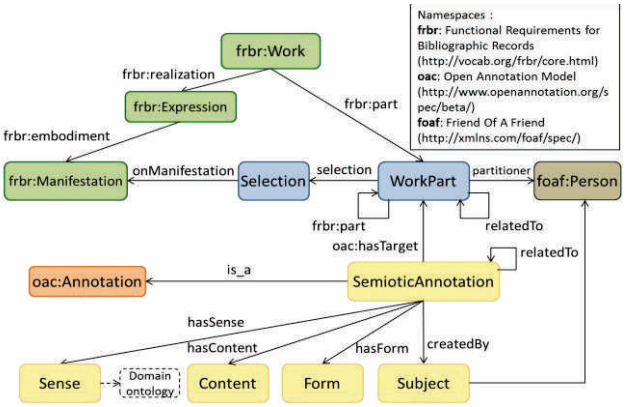


Figure 3: Semiotic Annotation model.

album (*Manifestation*). This is why we introduce a *Selection* concept to isolate the concerned *WorkPart* on any embodiment of a *Work*. For example, in the musical field, it can be:

- an extract of a MusicXML score: $[n_1,n_2,\ldots,n_n]$, where n_i are <*note*> elements defined by their bar number m_i and their order of appearance k_i in this bar (m_i,k_i),
- an area of a PDF score: $[(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)]$, where each (x_i,y_i) defines a summit of a selection polygon on the area of the score,
- an extract of a video performance: $[t_1,t_2]$ where t_i is a timecode of the video and $t_1<t_2$.

Naturally, a *WorkPart* resource can be composed of other *Workpart* resources thanks to the *frbr:part* relation. We also take into account the fact that different users may identify structures differently on a given piece. For instance, different musicologists will not necessarily identify the same structures, according to their respective interests (global structure, harmonic, rhythmic patterns or themes expositions). We thus introduce a *partitioner* relation to specify the person who extracts and names the part. This person is not necessarily the one creating the annotations afterwards, as structuring pieces and annotating them can be distinct activities. Our model also allows to link similar parts. To do so, the *relatedTo* relation can be specialized into more specific relations to indicate how different parts relate to each other (see application to the musical field in the next part). To name the part appropriately, a taxonomy may be helpful according to the considered field (e.g., the Musical Forms and Structures Ontology in the case of music). Once the part has been clearly extracted and identified, it can be annotated. Thus, a *SemioticAnnotation* concept is proposed, including the different components of the Sign (content, form, sense, according to a subject) presented previously. The sense component can be linked to a domain ontology in order to provide a semantic description of the SA for further processing. Of course, the subject is the creator of the SA, which conveys his personal interpretation. Our SA is perceived as a specialization of the *Annotation* concept from the Open Annotation Model already in use in several applications (Utopia[1], YUMA[2] framework). However, the SA is more centered on interpretation comparisons. This

[1] http://getutopia.com/index.php, visited on the 06/05/2013.
[2] YUMA Universal Media Annotator : https://github.com/yuma-annotation/, visited on the 06/05/2013.

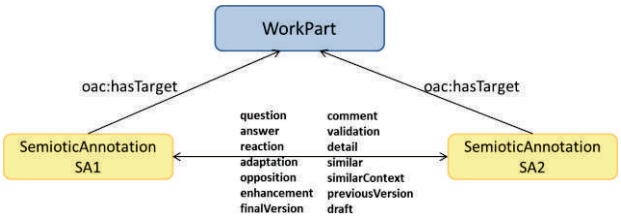


Figure 4: Relations between two Semiotic Annotations.

is why we also define relations to characterize adaptations, oppositions, and enhancements of SA. A SA can also simply be a reaction to another SA, such as an answer to a question, or a comment. Users can also keep a trace of their thought process by using versioning mechanisms on their annotations. Indeed, constructing an interpretation is generally an iterative process, where the subject enhances his performance with training and feedback. These relations are detailed in Figure 4. The main purpose of such annotations is to steer, organize and share helpful know-how emerging from practical cases.

This model can be applied to the musical field by replacing the *frbr:Work* concept by the more specialized *mo:MusicalWork* concept. The *WorkPart* concept will thus become the *MusicalWorkExtract* concept (which is not part of the current MO). But in the musical field, it is important to name the detected parts when possible. For instance, a user could identify a *Chorus*, a *Verse*, a *Fugue Theme*, a *Variation*, a *Leitmotiv*, a *Bass* part, a *Canon Verse*, a *Solo*, a *Musical Bridge*, etc. This is why we propose a conceptual model for a Musical Forms and Structures Ontology.

3. A MUSICAL FORMS AND STRUCTURES ONTOLOGY

Not to be confused with its genre, the form of a musical piece refers to its global organization and instrumentation [8]. For instance, the concerto form has three movements and features a solo instrument. A form sometimes implies the presence of significant structures. For example, a fugue necessarily contains a main theme, which will be exposed at each voice, and then developed, inversed, transposed in successive stretti. Besides, most basic structures have a type but not necessarily an explicit name (e.g., phrases, motifs, scales, arpeggios). While musical genres and moods are regularly studied in the MIR literature (for instance [6] or [4]), musical forms and structures are less discussed. Indeed, the latter are rather addressed in musicological contexts and remain unexploited in music recommendations systems and scores sharing communities. Thus, we design a basic framework for a Musical Forms and Structures Ontology (MFSO) to back up our Musical Performance Ontology with appropriate musicological terms. As noted in an analog work on a Musical Genres Taxonomy [6], achieving objectivity is a difficult task in a musical context. Concerning forms and structures, even specialists do not agree on some terms (see discussion). This is why we only provide high-level concepts for the time being, which are easier to differentiate, are well documented in musicological treaties such as [8] and largely used by musicians.

Our MFSO is thus designed to characterize any musical work and its extracts (Figure 5). A Musical Work can have a Musical Form (e.g., Sonata, Fugue, Song, Canon, etc.). This form can be linked to a characteristic genre (e.g., Fugues and Sonata are generally associated to classical music), but there may be exceptions (e.g., classical structures in symphonic rock). As pointed out previously a form can also be linked to its characteristic structures. These two links (*typicalOf* and *hasStructure*) do not aim at restricting the annotation possibilities, but rather at suggesting appropriate terms to the annotator, from basic metadata. For instance, the title of the piece may indicate its form (example: "*Sonata KV545*" by Mozart) which allows the annotation service to suggest appropriate structures to the user (example: *Exposition*, *Development*). But a musical extract does not necessarily have a name. This is why all named structures inherit from the generic *MusicalWorkExtract* concept. A *MusicalWork* resource can contain several *MusicalWorkExtract* resources. These extracts may be imbricated thanks to the *frbr:part* relation. A musician can detail how an extract is written with the *contains* relation: is it a whole *Phrase*, a *Motif*, a simple *Scale,* or a *Sequence* of *Chords*?

A musical extract can also be labeled. Musicians generally structure a piece by associating alphabetical labels to its different parts. This notation (or codification) allows to clearly distinguish repeated parts (example: ABABC, or ABAB'C meaning B' is almost like B). The *label* relation allows us to link our work to [9], which proposes notations conventions for structure labeling of musical extracts. We insist on the distinction between a notation and an annotation: a notation is a codification (i.e., a representation) of an object, while an annotation is a comment on an object. Thus, they are not related a priori, even if an annotation content can consist in a notation fragment.

The *relatedTo* relation allows to link distinct extracts. It can be specialized in order to express that an extract *introduces*, *concludes*, *imitates, transposes, ornates* or *accompanies* another one. These relations can be automatically associated with specific named structures. For instance, a *Variation* necessarily relates to a *Theme* and should be associated to it via the *variation* relation. Other automatisms can be implemented if a MusicXML representation of the piece is available. Indeed, notes, chords, scales and arpeggios can easily be extracted using the el-

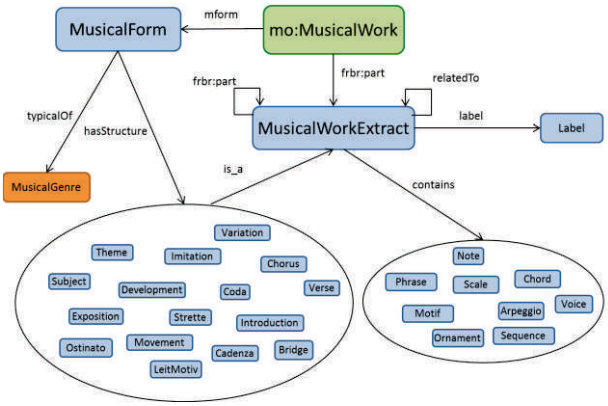


Figure 5: MFSO conceptual model.

ements names (e.g., <chords>) or by detecting regular structures (e.g., ascendant or descendant thirds sequences for arpeggios). Undefined regular patterns are more delicate to automatically identify, even if some works exist on this issue [10]. Some work has also been done on detecting Fugue structures [12]. Now that the part has been extracted and named, it can be annotated.

4. A MUSICAL PERFORMANCE ONTOLOGY

Our Musical Performance Ontology (MPO) aims at tagging Semiotic Annotation appropriately in a musical education context. It intervenes in the sense component of the SA (see Figure 7) to specify what themes are treated by the subject on the annotated extract. This organization has been chosen because the MPO deals with subjective interpretations and advice (emotions, expression, fingerings) rather than verifiable facts (artists, tracks, concerts), as the MO does. Embedding the MPO in a SA resource allows musicians to discuss their interpretations and possibly enhance them. Tagged SA can then be exploited by learning agents to answer complex queries, discover interesting resources, link similar SA, generate exercises or suggest appropriate SA to help a student to learn a new piece. For the time being, our proposition is not meant to be instrument-specific but is highly influenced by our experience at the piano and guitar. However, it is possible to extend it according to other instruments requirements.

The MPO conception methodology is inspired by the Archonte methodology [7]. Recurrent linguistic units were extracted from recorded piano lessons. These units were classified by an identity and differentiation process and each obtained class was labeled (see Figure 6). We then noticed that each label could be associated to one of the main activity of music practice: listening, playing and learning. Thus, we used these three activities to design our MPO concepts tree. This ontology can be used to semantically describe a Musical Expression (listening activity), an Instrumental Technique (playing activity) or an Assimilation Method (learning activity). Naturally, these three concepts are related: an Instrumental Technique produces a Musical Expression but requires an Assimilation Method to be handled by a learner (Figure 7). Besides, the sociologist Megan Winget also highlights Expression and Technique as essential annotation types in her study on musicians' annotation practices [13].

"We will study Espièglerie from Kabalevski. During the learning(6), we should try to keep the playful and light-hearted tone(1) of this little piece. The main work(6) is based on fifths sequences(5).

Concerning text learning(6), there are several parts(5) to consider in this work. There is the first part(5) where the left and right hand(7) play the same notes, namely two fifths(5) (plays(9)). For the right hand(7), we will develop(9) all notes but for the moment, it is better to play(9) only the fifths(5) like this, so that the notes are rapidly assimilated(6) (plays the fifths). One should pay attention to the 2-5 fingering at the left hand(7), which allows to get livelier and lighter detached notes(1). Here is what you should obtain (plays the part), here both hands(7) play at the same time(5).

That's all for the first part(5). This should be rapidly assimilated(6), with the right nuances(4)(1). I played it slowly(3), but it should not be a problem to play it at the right tempo(3) after repeating it 3 or 4 times(6) (plays the fifths at the right tempo). It should be played naturally(6)(1), and by heart(6) as soon as possible because it is rather difficult(6) to move on the keyboard and watch the score(7)(8) at the same time."

1 Sound 2 Harmony 3 Rhythm 4 Dynamic 5 Structure 6 Assimilation 7 Gesture 8 Support 9 Relation

Figure 6: Piano lesson analysis example: extraction and classification of significant linguistic units.

The *MusicalExpression* concept regroups all concepts which deal with musical writing (*Rhythm*, *Harmony*, *Structures*, volume and tempo *Dynamics*) and its resulting *Sound*. Various sound features can be described, such as pitch, duration, timbre, mood and articulation (i.e. legato, staccato). The Structure concept regroups elements from the MFSO previously presented. Indeed, the SA may discuss specific elements inside the selected part. For instance, a musician can extract all occurences of a leitmotiv in a given musical work. This naming task is realized at the *MusicalWorkExtract* resource level, thanks to the *Leitmotiv* concept from the MFSO. An other musician can then annotate one of the occurences and insist on one of its intervals requiring more attention than the others. This time, the structure identification is at the SA level, but still requires the *Interval* concept defined by the MFSO. This is because the second structure identification is more a personal approach of the piece and will not necessarily catch the attention of a musicologist as it does for a performer, who has to "animate" the piece. This approach gives more flexibility to the users, by allowing them to distinguish high level and meaningful structures (theme, phrases) from basic elements (note level). Different relations exist between these musical expression concepts but were not represented in Figure 7 for readability purposes. For example, Duration and Rhythm are strongly related, as extending a note duration (e.g., a fermata) has an impact on the overall rhythmic organization around the note. Identified structures can also be linked to their harmonic (*hasHarmony*) and rhythmic (*hasRhythm*) features. The Harmony concept can be linked to the Chord Ontology[1] to specify the chords at stake in a standardized notation.

The *InstrumentalTechnique* concept allows the musician to tag any instrument-related technical matter. It deals with gestures, movements, fingerings and positions

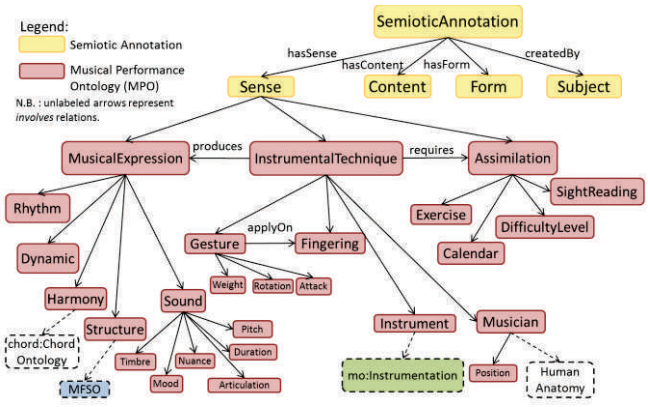


Figure 7: Musical Performance Ontology conceptual model.

[1] http://www.omras2.org/ChordOntology, visited on the 06/05/2013.

which are essential to produce the desired musical expression. As can be seen in Figure 7, an Instrumental Technique involves a musician's limbs or articulations (arms, hands, wrists, etc.) and a part of his instrument (on the piano: the keyboard, the pedals, the white keys, the black keys, etc. in singing: the vocal cords, the lungs, the mouth, etc.). The *produces* relation allows to link a technique to its resulting musical effect. In practice, the *Fingering* concept will certainly be the most used one, as it is one of the most important information for sight-reading musicians. If a MusicXML score of the piece is available, a fingering can be associated to a copy of the annotated extract (MusicXML 3.0 supports <fingering> nodes). Using copies instead of the original MusicXML file allows musicians to propose several fingering solutions for a single part. The idea is to clearly distinguish the basic abstract notation (MusicXML notes, provided by editors) from the concrete annotations demonstrating musical know-how (SA, provided by musicians).

The *Assimilation* concept deals with educational mechanisms, to help a student to handle an instrumental technique. In particular, it aims at providing a framework to create exercises based on rearrangements of the original score. For instance, piano teachers often reduce a part's difficulty by dividing it into smaller parts (i.e. intervals). These elements are repeated three or four times and then grouped by two, by three, etc…gradually increasing the difficulty. We could thus generate the corresponding score for each step of the exercise. Other educational mechanisms can be described in the Assimilation concept: beats counting, sight-reading methods, work calendar, learning by heart, etc.

To have a better understanding of how this ontology can be used and for what purposes, we propose a simple use case in the following section.

5. SA, MFSO AND MPO USE CASE

We propose a fictive use case to study how the proposed SA, MFSO and MPO models can work together. A musician wishes to annotate a score with the following assertions:

- "Here is an occurrence of leitmotiv A. I label it A1."
- "Here is an other occurrence of leitmotiv A. I label it A2."
- "A1 and A2 are almost similar, except A2 is a transposition of A1."
- "Both are based on the following rhythm: eighth – sixteenth triplet – quarter."
- "Beware of the articulation on A1's first note: it is staccato. It must be done with the whole arm to prevent a too sharp attack and prepare for the next position. It is also possible to lightly increase the volume on the ascendant arpeggio. But always keep the solemn mood of the piece."

Figure 8 represents the identified RDF[1] triplets. We highlight the fundamental distinction between the studied objects (the piece and its extracts, described with the MO and MFSO) and their interpretations (semiotic annotations, described with the SA and MPO). The relation between the two leitmotiv occurences (transposition) allows to deduce the type of A2. It also allows to duplicate the explanations given for A1 on A2, as long as tonal dependant concepts are not involved (which is the case here). The "Rhythm URI" resource can be linked to an appropriate rhythm notation resource (e.g., a MusicXML short fragment). Common articulations, dynamics and characters can be suggested in a list to facilitate the semantic description input. The description of the arm gesture remains brief, but will enable to return this annotation for requests such as "how to play staccato?". The content and form components of the SA were not represented in Figure 8 which focuses on semantic description, but can be added by the user. Typically, the staccato gesture should be demonstrated with a video resource (content component). The form component allows the user to select different ways of indexing his annotation. But did the musician use the appropriate terms to describe his interpretation ?

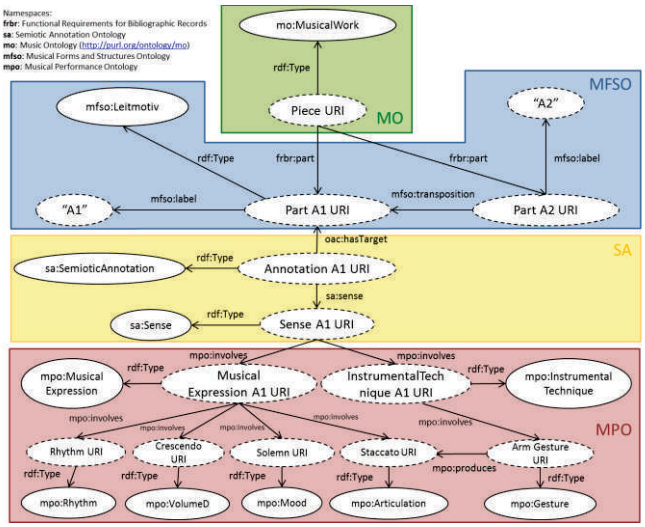


Figure 8: SA, MFSO and MPO use case example.

6. DISCUSSION

As noted earlier, using appropriate words in music is a complex issue, mainly for subjectivity and context reasons. Naturally, musicology also aims at clearly defining terms and analysis methods in this field. But sometimes, musicologists themselves do not agree on appropriate terms to designate structures. For instance, there is a "French school" and an "Anglo-Saxon school" with distinct methods, vocabularies and notations customs.

Indeed, the concept of "rule" in music is complex. There is a framework (mainly the tonal system in occidental music), a history, a culture, an aesthetic which were built over time and various influences, but no definite rule. Thus, the line between form, genre and style can be thin in many circumstances.

Some works exist in order to clarify the manipulated terms meanings and to prevent confusion in musical writ-

[1] Resource Description Framework.

ings, such as the GDRM[1] initiative at Laval University. But the most important factor is the context of use. For example, the "pedal" can either designate a long bass note, or a part of the piano. This example is especially interesting because, even if the two meanings are distinct, they are somehow related and can explain why the same word is used: indeed, the piano pedal do help to produce a bass pedal in some cases. Both meanings thus imply the idea of a "long resonance". For the time being, we propose to overcome this issue by making a distinction between the concept and its common designation. For the pedal example, we define two concepts: *BassPedal* and *PianoPedal*, having both the designation "pedal", corresponding to the way they are commonly called by musicians.

The difficulty also lies in the interrelation of the defined concepts. As seen in the use case, musicians rarely address one theme at a time, as all musical elements interacts with each other, and the musician should have control on each of them while playing. Providing an extensive semantic description in such case is very difficult. This is why we rely on common concepts and try to establish simple relations between them. But more specific relations can only emerge through intensive use of the annotation platform. This is why we rely on an iterative construction of our descriptive model [14].

Thus, our collaborative annotation platform aims at fostering fruitful debates to help musicians confront their ways of analyzing and practicing music.

7. CONCLUSION

In this paper, we proposed a Musical Forms and Structures Ontology (MFSO) and a Musical Performance Ontology (MPO) to annotate performances and scores semantically. These ontologies aim at extending the standard Music Ontology with musicological and musical know-how concepts. To do so, we first introduced a Semiotic Annotation framework to allow users to describe personal interpretations of musical works and then introduced our MFSO. This allows musicians to name precisely the musical extract to annotate and its role in the work. Musical expressions, techniques and assimilation methods can then be described thanks to appropriate concepts and relations from our MPO. This work notably aims at building a music learning agent which can help musicians to answer complex queries, discover interesting knowledge among large digital scores collections, generate exercises or suggest appropriate SA to help a student on a new piece.

Naturally, perspectives for this work include testing it with musicians of all levels, which will allow us to refine and extend our proposition. To do so, a collaborative semiotic annotation platform for mobile tablets is currently being developed.

8. REFERENCES

[1] V. Sébastien, P. Sébastien, N. Conruyt, "@-MUSE: Sharing Musical Know-how Through Mobile Devices Interfaces", *5th Conference on e-Learning Excellence in the Middle East,* Dubaï, 2012.

[2] V. Sébastien, H. Ralambondrainy, O. Sébastien, N. Conruyt, "Score Analyzer: Automaticaly Determining Scores Difficulty Level for Instrumental e-Learning", *Proceedings of the International Conference on Music Information Retrieval*, ISMIR, 2012.

[3] T. Vander Wal, "Folksonomy", *Online posting*, No 7, 2007.

[4] M. Sordo, "Semantic Annotation of Music Collections: A Computational Approach", PhD thesis report, Universitat Pompeu Fabra, 2012.

[5] N. Conruyt, D. Grosser, R. Vignes-Lebbe, "Knowledge Discovery for Biodiversity: from Data Mining to Sign Management", *International Congress on Environmental Modeling and Software*, Leipzig, Germany, 2012.

[6] F. Pachet and D. Cazaly, "A Taxonomy of Musical Genres", in *Content-Based Multimedia Information Access Conference (RIAO)*, Paris, 2000.

[7] B. Bachimont, *Ingénierie des connaissances et des contenus, le numérique entre ontologies et documents*. *(Tr.: Knowledge and contents engineering, digital technologies between ontologies and documents)*, Lavoisier, Paris, 2007.

[8] A. Hodeir, *Les Formes de la musique (Tr.: The Forms of Music)*, Presses Universitaires de France, 2012.

[9] F. Bimbot, E. Deruty, G. Sargent, and E. Vincent, "Semiotic structure labeling of music pieces: concepts, methods and annotation conventions", in *Proceedings of the International Symposium on Music Information Retrieval*, vol. 2, 2012.

[10] O. Lartillot: "Une analyse musicale automatique suivant une heuristique perceptive" *(Tr.: An automatic musical analysis based on a perceptive heuristic)*, *3ème Conférence Internationale Francophone sur l'Extraction et la Gestion des Connaissances*, EGC 03, Lyon, 2003.

[11] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson: "The Music Ontology", *Proceedings of the International Conference on Music Information Retrieval*, ISMIR, 2007.

[12] M. Giraud, R. Groult, F. Levé, "Detecting Episodes with Harmonic Sequences for Fugue Analysis", *Proceedings of the International Conference on Music Information Retrieval*, ISMIR, 2012.

[13] M.A. Winget, "Annotations on musical scores by performing musicians: Collaborative models, interactive methods, and music digital library tool development", *Journal of the American Society for Information Science and Technology*, vol. 59, no. 12, pp. 1878-1897, 2008.

[14] N. Conruyt and D. Grosser, "Knowledge Management in Environmental Sciences with IKBS: Application to Systematics of Corals of the Mascarene Archipelago", *Selected Contributions in Data Analysis and Classification*, pp. 333-343, 2007.

[1] http://www.mus.ulaval.ca/roberge/gdrm/, visited on the 06/05/2013.

CHORD-SEQUENCE-FACTORY: A CHORD ARRANGEMENT SYSTEM MODIFYING FACTORIZED CHORD SEQUENCE PROBABILITIES

Satoru Fukayama **Kazuyoshi Yoshii** **Masataka Goto**
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{s.fukayama, k.yoshii, m.goto}@aist.go.jp

ABSTRACT

This paper presents a system named *ChordSequenceFactory* for automatically generating chord arrangements. A key element of musical composition is the arrangement of chord sequences because good chord arrangements have the potential to enrich the listening experience and create a pleasant feeling of surprise by borrowing elements from different musical styles in unexpected ways. While chord sequences have conventionally been modeled by using N-grams, generative grammars, or music theoretic rules, our system decomposes a matrix consisting of chord transition probabilities by using nonnegative matrix factorization. This enables us to not only generate chord sequences from scratch but also transfer characteristic transition patterns from one chord sequence to another. *ChordSequenceFactory* can assist users to edit chord sequences by modifying factorized chord transition probabilities and then automatically re-arranging them. By leveraging knowledge from chord sequences of over 2000 songs, our system can help users generate a wide range of musically interesting and entertaining chord arrangements.

1. INTRODUCTION

Chord sequences are essential when composing and arranging music. Different songs, composers, arrangers, and musical genres have different tendencies to use chord sequences, which contributes to increased variety in music. Each song could have different natural chord sequences that give different impressions. Although an arranger can change (*i.e.*, arrange) chord sequences of a song to alter its mood, this is very difficult for people who lack knowledge of chord sequences to arrange the chords in an appropriate way. The goal of this research is to assist people to generate variations of chord sequences from an input original sequence by leveraging knowledge from a large number of other existing chord sequences called *references*.

Chord sequence arrangement is a promising approach to create derivative works from existing songs. Although amateur creators could create such derivative works themselves, known as user-generated content (UGC) on video sharing services, the majority of users are generally music listeners who do not create much themselves. We aim to encourage such people to create more derivative works by changing chord sequences. This enables music listeners to personalize songs by customizing chord sequences without any special training [4]. Such content personalization was achieved by *Drumix* [15], a system for arranging the drum track in music audio signals, but content personalization using chord arrangements has not yet been studied.

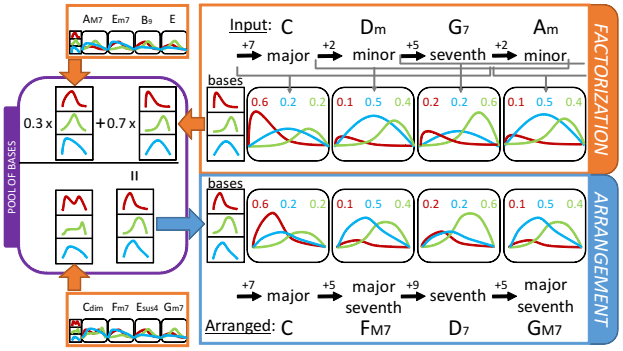


Figure 1. Overview of generating chord arrangements with *ChordSequenceFactory*.

To generate chord arrangements, it is necessary to model and manipulate chord sequences. Conventionally, models for generating chord sequences have been represented by probabilistic models or N-grams [1, 6, 8, 9, 11, 12, 14], genetic algorithms [2], generative grammars [10, 13], and exploiting examples or templates [3, 7]. Although these existing models can easily be used to generate new chord sequences from scratch [2], it is difficult to arrange chord sequences of an existing song while referring to other existing songs.

In this paper we propose a new mathematical formulation of chord sequences using nonnegative matrix factorization (NMF) and use it to build a system, *ChordSequenceFactory,* that enables users to arrange chord sequences of existing songs (Fig. 1). The chord sequence of each song (a sequence of symbols) is first converted into a *chord differential matrix* that represents chord transitions in short regions. The matrix is then decomposed into a set of bases (characteristic chord transition patterns) and a set of corresponding temporal activations using NMF. Chord arrangements can be achieved by interpolating the bases of a song with similar bases obtained from other songs while preserving the activations. An arranged chord sequence is

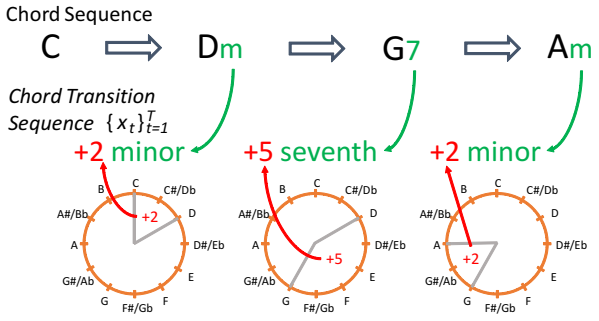


Figure 2. The representation of a sequence of a chord transition used in our method. Each transition is represented as a combination of the interval between the root notes of adjacent chords and the type of the latter chord.

finally generated from the reconstructed *chord differential matrix* as the product of the resulting bases and the original activations.

The remainder of this paper is structured as follows. In Section 2, we present the analysis and synthesis framework of chord symbol sequences based on the *chord differential matrix*. In Section 3, we present the formulation of chord arrangements. In Section 4, we present experimental results to illustrate the performance of the system *ChordSequenceFactory*. Finally in Sections 5 and 6 we include discussion and conclusions, which summarize the main contributions of the paper.

2. ANALYSIS AND SYNTHESIS OF SYMBOLIC CHORD SEQUENCES

2.1 Chord transition sequence

Chord names typically consist of up to three labels: root note, chord type, and bass note specification [5]. Since the bass notes are omitted in many cases, we treat chord information as a combination of the first two labels. In general, chord transitions are considered to be more important than the absolute pitch of the root note of each chord. This is supported by the fact that it is possible to transpose chord sequences into other tonalities without affecting the functions of chord idioms used in these sequences.

In this paper we do not define a vocabulary of chord names but rather, a vocabulary of chord "transitions" for modeling how adjacent chords are arranged in music. Let $\{c_n\}_{n=1}^{N}$ be the vocabulary of chord transitions, where N is the size of the vocabulary and each c_n is defined as follows:

$$c_n \equiv \mathrm{str}\,(\mathrm{INTERVAL}) + \mathrm{str}\,(\mathrm{TYPE})\,, \qquad (1)$$

where INTERVAL indicates the difference in semitones between the root notes of a target chord and the previous chord and TYPE indicates the type of the target chord. An example chord transition sequence is shown in Fig. 2. Since we do not need to know the direction of root changes, the value of INTERVAL is restricted to a nonnegative integer. The operation str(·)+str(·) means the string conjunction (*i.e.*, concatenation). Once a given sequence of chords

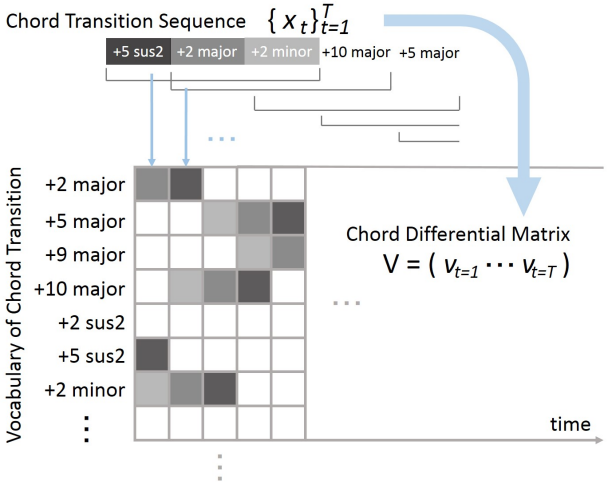


Figure 3. Representation of a chord transition sequence with the *chord differential matrix* representation.

is converted into a sequence of chord transitions according to the defined vocabulary, we can recover the original sequence if the first chord of the sequence is given. This representation has also been used in related work [6] for the same purpose of normalizing the tonality of chord sequences.

An advantage of this representation is that it can reduce the number of parameters to be dealt with. The typical approach to modeling chord sequences is to calculate a transition probability matrix over chord names defined in a vocabulary. This approach, however, requires a large number of parameters, *i.e.*, we need to deal with $M \times M$ parameters if M kinds of chord names are contained in the vocabulary. Since the chord type seems to be less dependent on the type of the previous chord name, we directly focus on the root-note interval and the current chord type.

2.2 Analysis of a chord transition sequence based on a chord differential matrix

We now explain the probabilistic representation of a chord-transition sequence $\{x_t\}_{t=1}^{T}$, where $x_t \in \{c_n\}_{n=1}^{N}$ and T is the length of the sequence. We analyze the sequence on frame-by-frame basis using an exponentially-decaying window as shown in Fig. 3. This window is designed based on our assumption that the characteristics of chord transitions remain the same for some period of time because musical pieces are usually composed so that each section gives a coherent impression. The window is moved one by one from the first chord of the given sequence. In each frame t, we calculate a probability vector $\boldsymbol{v}_t \in \mathbb{R}^N$ such that the elements of the vector sum to unity. More specifically, v_{tn} is a ratio of chord transition c_n to all possible transitions in frame t, which is given by

$$v_{tn} = \frac{\sum_{0\le\tau\le\lambda} \delta_{c_n x_{t+\tau}} e^{-\tau}}{\sum_{0\le\tau\le\lambda} e^{-\tau}}, \qquad (2)$$

where δ_{ij} is the Kronecker delta, λ is the window length, and $e^{-\tau}$ is a temporally-decaying weight. Note that $\boldsymbol{v}_t$ indicates a co-occurrence relationship between chord transi-

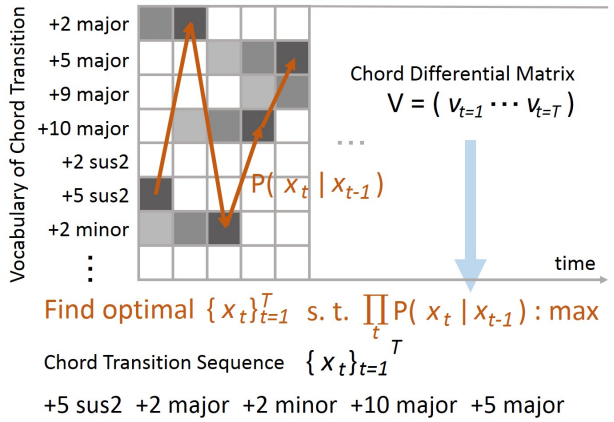


Figure 4. Regeneration of a chord transition sequence from a chord differential matrix representation by means of V and the bi-gram probability obtained *a priori* from the data.

tions in the vicinity of frame t. The process of representing the chord transition sequence as a *chord differential matrix* is shown in Fig. 3.

This frame-based vectorial representation of chord transition probabilities has useful properties for chord arrangement as follows:

- **Mood coherence in short durations**
  Although chords often change at bar boundaries, the mood does not change in such a short time span because we use an exponentially-decaying window of length λ.

- **Reproducibility of chord sequences**
  We can approximately reconstruct the original sequence $\{x_t\}_{t=1}^T$ from a sequence of probability vectors $\{\boldsymbol{v}_t\}_{t=1}^T$ in a principled manner.

2.3 Synthesis of a chord transition sequence

To regenerate a chord transition sequence from a chord differential matrix, we need to consider the transition between successive chord transitions x_t and x_{t-1}. Using the transition probability $P(x_t|x_{t-1})$ trained from the data, we can calculate the probability of observing $\{x_t\}_{t=1}^T$ as follows:

$$P\left(\{x_t\}_{t=1}^T\right) = \prod_{t=1}^{T} \left(\xi v_t(x_t) + (1-\xi) P(x_t|x_{t-1})\right), \tag{3}$$

where ξ is an interpolation coefficient such that $0 \leq \xi \leq 1$.

The chord transition sequence $\{x_t^*\}_{t=1}^T$ is reconstructed by maximizing $P\left(\{x_t\}_{t=1}^T\right)$ as follows:

$$\{x_t^*\}_{t=1}^T = \underset{\{x_t\}_{t=1}^T}{\operatorname{argmax}} P\left(\{x_t\}_{t=1}^T\right). \tag{4}$$

Since there are N possibilities for each x_t, it is computationally infeasible to test all N^T possible sequences with the naive method of exhaustive search. Fortunately, we can obtain the solution $\{x_t^*\}_{t=1}^T$ with $O(N)$ using dynamic programming. The process for generating a sequence is shown in Fig. 4.

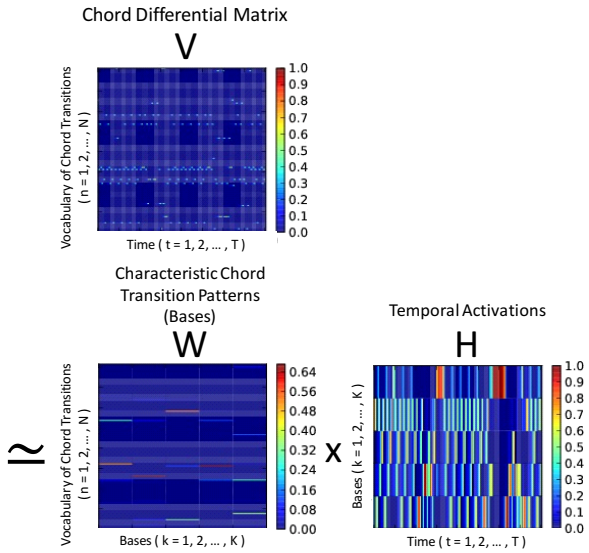


Figure 5. Factorizing the chord deferential matrix: Analysis of the frequent combinations and the temporal occurrences of chord transitions.

3. FORMULATION OF CHORD-SEQUENCE-FACTORY

3.1 Factorization of a chord differential matrix

An overview of *ChordSequenceFactory* is illustrated in Fig. 6. The mood of music varies according to sections of a song. For instance, some sections often use dominant intervals and other sections tend to use chords with more tension notes. Therefore we aim to identify the characteristic patterns of chord transitions that affect the mood of each section. We represent a probability vector $\boldsymbol{v}_t$ at each time t as a convex combination of multiple bases as follows:

$$\boldsymbol{v}_t = \sum_{k=1}^{K} h_{kt} \boldsymbol{w}_k, \tag{5}$$

where $\boldsymbol{w}_k, (k = 1, \cdots, K)$ denotes a characteristic chord transition pattern and h_{kt} is its weight. In order to decompose $\{\boldsymbol{v}_t\}_{t=1}^T$ using shared bases $\{\boldsymbol{w}_k\}_{k=1}^K$, we exploit nonnegative matrix factorization (NMF) (Fig. 5), *i.e.*,

$$V = (\boldsymbol{v}_1 \cdots \boldsymbol{v}_T) \tag{6}$$

is decomposed into matrices $W\,(N \times K)$ and $H\,(K \times T)$ as:

$$V \simeq WH, \tag{7}$$

where W is the matrix of bases:

$$W = (\boldsymbol{w}_1 \cdots \boldsymbol{w}_K) \tag{8}$$

and H is the matrix of activations:

$$H = \begin{pmatrix} h_{11} & \cdots & h_{1T} \\ \vdots & \ddots & \vdots \\ h_{K1} & \cdots & h_{KT} \end{pmatrix}. \tag{9}$$

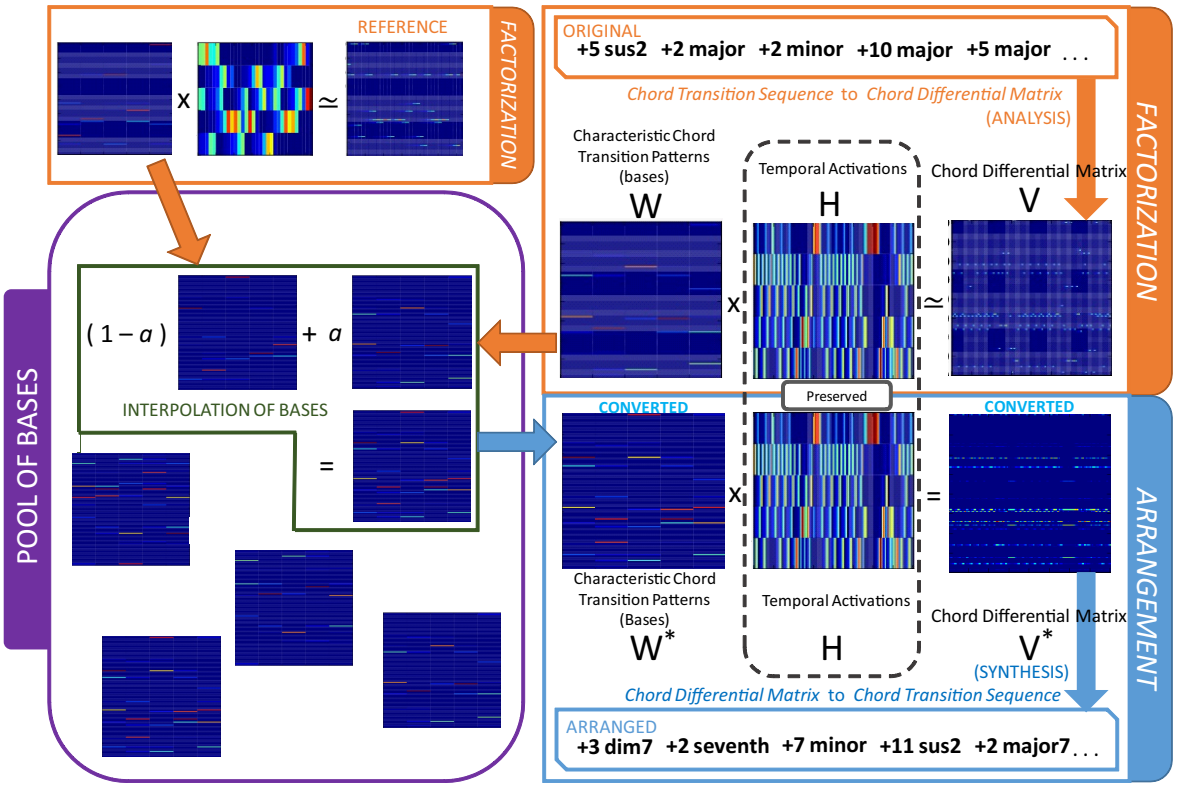


Figure 6. Overview of the process executed by *ChordSequenceFactory* for generating chord arrangements.

3.2 Interpolation

We want to modify the V of a target song by referring to chord transition patterns used in another song. More specifically, to obtain a reconstructed chord differential matrix V^*, we reuse the H of the target song and use a set of modified vectors $W^* = (\boldsymbol{w}_1^* \cdots \boldsymbol{w}_K^*)$ as follows:

$$V^* = (\boldsymbol{w}_1^* \cdots \boldsymbol{w}_K^*)\, H \tag{10}$$

$$= W^* H. \tag{11}$$

To obtain each modified vector $\boldsymbol{w}_i^*$, we interpolate a reference vector $\boldsymbol{w}_i^{\mathrm{ref}}$ of another song with the original vector $\boldsymbol{w}_i$. Since the original bases $\{\boldsymbol{w}_k\}_{k=1}^K$ and reference bases $\{\boldsymbol{w}_k^{\mathrm{ref}}\}_{k=1}^K$ are not guaranteed to have their index aligned, we associate each $\boldsymbol{w}_i$ with a reference $\boldsymbol{w}_{j(i)}^{\mathrm{ref}}$ such that $\boldsymbol{w}_i$ is closest to $\boldsymbol{w}_{j(i)}^{\mathrm{ref}}$, *i.e.*,

$$j\,(1), \cdots, j\,(K) = \underset{j(1),\cdots,j(K)}{\operatorname{argmax}} \sum_{k=1}^{K} D\left(\boldsymbol{w}_i \| \boldsymbol{w}_{j(i)}^{\mathrm{ref}}\right), \tag{12}$$

where D is a distance measure based on the symmetric Kullback Leibler divergence:

$$D\left(\boldsymbol{w}_i \| \boldsymbol{w}_{j(i)}^{\mathrm{ref}}\right) = \sum_k w_{ik} \log \frac{w_{ik}}{w_{j(i)k}^{\mathrm{ref}}} + \sum_k w_{j(i)k}^{\mathrm{ref}} \log \frac{w_{j(i)k}^{\mathrm{ref}}}{w_{ik}}. \tag{13}$$

Using the aligned indicies $j\,(1), \cdots, j\,(K)$, we can calculate the modified vector $\boldsymbol{w}_i^*$ as follows:

$$\boldsymbol{w}_i^* = a\boldsymbol{w}_i + (1-a)\,\boldsymbol{w}_{j(i)}^{\mathrm{ref}}, \tag{14}$$

where a is the interpolation parameter such that $0 \leq a \leq 1$. The value of a represents how the mood of the original song is preserved through the chord arrangement.

3.3 Generation of chord arrangements

As discussed in Section 2.3, we can generate an arranged chord transition sequence from the modified chord differential matrix V^*. We interpolate the transition probability obtained from all songs in the database $P_{\mathrm{all}}\,(x_t|x_{t-1})$ and that obtained from the reference song $P_{\mathrm{ref}}\,(x_t|x_{t-1})$ with that obtained from the original song $P_{\mathrm{org}}\,(x_t|x_{t-1})$ as follows:

$$P^*\,(x_t|x_{t-1}) = \xi_1 P_{\mathrm{all}}\,(x_t|x_{t-1}) + \xi_2 P_{\mathrm{org}}\,(x_t|x_{t-1}) + \xi_3 P_{\mathrm{ref}}\,(x_t|x_{t-1}), \tag{15}$$

where $\{\xi_i\}_{i=1}^3$ are the interpolation coefficients that sum to unity, *i.e.*, $\sum_i \xi_i = 1$. Using $P^*\,(x_t|x_{t-1})$, we can calculate the probability of observing $\{x_t\}_{t=1}^T$ as follows:

$$P\left(\{x_t\}_{t=1}^T\right) = \prod_{t=1}^{T} (\xi v_t^*\,(x_t) + (1-\xi)\,P^*\,(x_t|x_{t-1})). \tag{16}$$

The arranged chord transition sequence is then obtained by maximizing $P\left(\{x_t\}_{t=1}^T\right)$ (see Section 2.3).

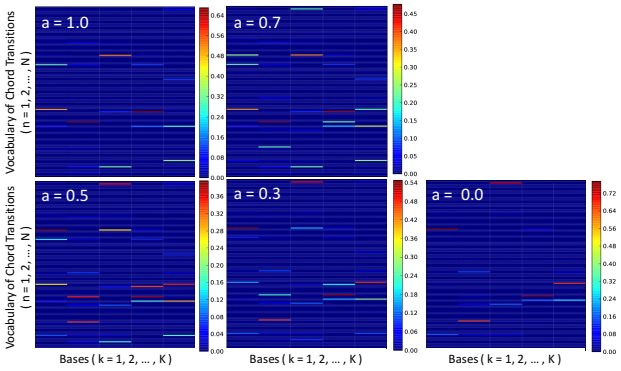


Figure 7. Examples of the bases interpolated with bases of a reference song: we can see that the chord usages originated in two different songs are combined, depending on the value of interpolation factor a.

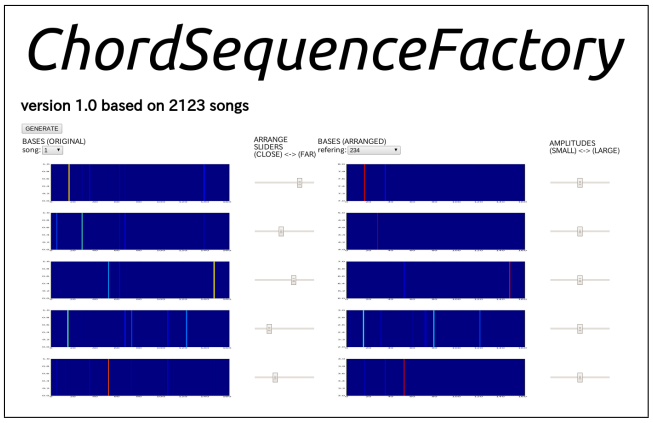


Figure 8. User interface of *ChordSequenceFactory*: users can change the interpolation factor with sliders corresponding to each base.

4. EVALUATION

4.1 Experimental conditions

To evaluate our system, we used 2,123 lead sheets downloaded from the Wikifonia (www.wikifonia.org) website. All lead sheets we used were formatted in the MusicXml format including information of chord sequence. Each chord name in a file included the step (C, D, ...), alternation (#, ♭) of the root note, and the chord type.

To define a vocabulary $\{c_n\}_{n=1}^{N}$, we extracted chord transitions that appeared more than 10 times in the data. A special symbol "Unknown" $\in \{c_n\}_{n=1}^{N}$ was used for representing the rest. The vocabulary size was $N = 163$. Using the vocabulary, we represented each song as a chord transition sequence $\{x_t\}_{t=1}^{T}$ and calculated a chord differential matrix from that sequence by using an exponentially-decaying window of length $\lambda = 6$, corresponding to the number of chord changes. The transition probabilities between $\{c_n\}_{n=1}^{N}$ were calculated in advance for all 2,123 songs and for each song, respectively. The chord differential matrix was decomposed using NMF based on the Euclidean distance with $K = 5$.

We interpolated characteristic chord transition patterns (basis vectors) of a reference song with those of an original song according to an interpolation coefficient $a = 1.0$, 0.7, 0.5, 0.3, or 0.1. A chord transition sequence was synthesized from a modified chord differential matrix. Since the vocabulary of chord transitions only holds information of relative root positions, we set the root note of the initial chord to the original root note.

4.2 Experimental results and discussions

As shown in Fig. 7, we can combine two different characteristic chord transition patterns by controlling the interpolation factor a on the user interface of *ChordSequenceFactory* (Fig. 8). The generated examples of chord arrangements are shown in Table. 1. We confirmed that the mood of an original song can be modified while incorporating the mood of a reference song if these songs have some simillar characteristic chord transition patterns.

Through informal evaluation and listening tests we found our system was, in many cases, able to provide musically coherent chord sequence arrangements (Table. 1). However, our evaluation also revealed some limitations that should be addressed in our future work. Since the vocabulary of chord transitions defined for the system does not include the absolute pitch for the root note, the generated results tended to exhibit transposition frequently. In addition, finding the optimal number of bases when decomposing the probability should be investigated for better performances. Finally, a slider for changing the activation can bring about more effects in the generated sequence.

Although simply adding a seventh note to chord sequences (rather than using our approach) has a similar effect on chord arrangements, dissonance may appear in the connection between the chords. In contrast, with our system, dissonant chords can be avoided by using the constraints given by the transition probabilities. Furthermore, there are no restrictions in terms of combining two songs that have different structures, since the interpolation is done between the decomposed basis.

Conventional methods based on N-grams cannot control the dynamic characteristics of chord transitions and need label sequences to reflect human intention. In our method, we can see that the probability is changed for each time t and that the activations at time t play the same role as the label sequences in the conventional methods.

5. CONCLUSIONS

We have described a system *ChordSequenceFactory* that can assist a user to arrange chord symbol sequences of a song by finding latent frequent patterns of chord transitions and modifying them on the basis of other songs. We proposed a new analysis and synthesis framework for chord symbol sequences, where the temporal changes of chord transition sequences are represented as a chord differential matrix. NMF is used to decompose this matrix into bases corresponding to the frequent patterns of chord transitions and activations corresponding to their temporal occurrences. The matrix can then be updated for a new arrangement by modifying these bases by mixing them with

ORIGINAL			Cmin	B♭maj	E♭sus2	Fmaj	Gmin	Fmaj	B♭maj	Gmaj
	reference	a								
ARRANGED	A	0.7	Cmaj7	Fmaj7	B♭7	E♭7	A♭maj6+9	G♭maj	Bmaj	Gmin6
		0.6	Cmaj	B♭maj	E♭7	A♭7	D♭maj6+9	Bmaj	Emaj	Cmin6
		0.5	Cmaj	B♭maj	Cmaj	Dmaj	Cmaj	B♭maj	E♭maj	A♭min
	B	0.7	Cdim	E♭dim	E♭7	E♭7	F#dim	Adim	D7	Gmin6
		0.6	Cdim	E♭dim	E♭7	E♭7	F#dim	Adim	D7	Gmin6
		0.5	Cmaj	B♭maj	Cmaj	Dmaj	Cmaj	B♭maj	E♭maj	B♭min6
	C	0.7	Cmaj	Fmaj	F7	F7	B♭maj	E♭maj	A♭maj	Gsus4
		0.6	Cmaj	Fmaj	F7	F7	B♭maj	E♭maj	A♭maj	Gsus4
		0.5	Cmaj	Fmaj	F7	F7	B♭maj	E♭maj	A♭maj	Gsus4
	D	0.7	Cmin7	Dmin7	Dmaj7	Dmaj7	Emin7	Amaj7	Dmaj7	Cmaj7
		0.6	Cmin7	Dmin7	Dmaj7	Dmaj7	Emin7	Amaj7	Dmaj7	Cmaj7
		0.5	Cmaj	B♭maj	Cmaj	Dmaj	Cmaj	B♭maj	E♭maj7	A♭min
	E	0.7	Cmaj6	Fmaj6	B♭sus4	E♭sus4	A♭maj6	D♭maj6	F#7	Bmin
		0.6	Cmaj6	Fmaj6	B♭sus4	E♭sus4	A♭maj6	D♭maj6	F#7	Bmin
		0.5	Cmaj6	Fmaj6	B♭sus4	E♭sus4	A♭maj6	D♭maj6	F#major	Bmin

Table 1. Generated results of *ChordSequenceFactory* with five different reference songs (A,B,C,D,E). For each reference song, three values of interpolation factor $a = 0.7, 0.6, 0.5$ were used. The arranged chord sequences were decoded from the chord transition sequence $\{x_t\}_{t=1}^{T}$, by setting the root note of the first chord to C as in the original sequence.

similar bases in the pool of bases obtained from more than 2000 songs. The updated matrix is finally used to generate a new re-arranged chord sequence using dynamic programming. In our experience, *ChordSequenceFactory* generated musically interesting and entertaining chord arrangements. In the future, we plan to extend our framework to consider melody lines as constraints on arranged chord sequences. We will also include audio signal processing so that the chord differential matrix can be used directly on music audio signals.

Acknowledgement
This work was supported in part by OngaCREST, CREST, JST.

6. REFERENCES

[1] C. Ames: "The Markov Process as a Compositional Model: A Survey and Tutorial," *Leonardo Music Journal*, Vol. 22, No. 2, pp. 175–187, 1989.

[2] J. Biles: "GenJam: A Genetic Algorithm for Generating Jazz Solos," *Proceedings of ICMC*, pp. 131–137, 1994.

[3] D. Cope: *Experiments in Musical Intelligence*, A-R Editions, 1996.

[4] M. Goto: "Active Music Listening Interfaces based on Signal Processing," *Proceedings of ICASSP*, pp. 1441–1444, 2007.

[5] C. Harte, M. B. Sandler, S. A. Abdallah, E. Gómez: "Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations," *Proceedings of the ISMIR conference*, pp. 66–71, 2005.

[6] M. Mauch, S. Dixon, C. Harte, M. Casey, B. Fields: "Discovering Chord Idioms through Beatles and Real Book Songs," *Proceedings of the ISMIR conference*, pp. 255–258, 2007.

[7] F. Pachet: "Surprising Harmonies," *International Journal of Computing Anticipatory Systems*, Vol. 4 1999.

[8] J. Paiement, D. Eck, S. Bengio: "A Probabilistic Model for Chord Progressions," *Proceedings of the ISMIR conference*, pp. 312–319, 2005.

[9] H. Papadopoulos, G. Peeters: "Large-scale study of chord estimation algorithms based on chroma representation and hmm," *Proceedings of the ISMIR conference*, pp 225-258, 2007.

[10] M. Rohermeier: "Towards a generative syntax of tonal harmony," *Journal of Mathematics and Music*, Vol. 5, No. 1, pp. 35–53, 2011.

[11] R. Scholz, E. Vincent, F. Bimbot: "Robust modeling of musical chord sequences using probabilistic N-grams," *Proceedings of the ICASSP*, pp. 53–56, 2009.

[12] A. Sheh, D. Ellis: "Chord Segmentation and Recognition using EM-Trained Hidden Markov Models," *Proceedings of the ISMIR conference*, pp. 185–191, 2003.

[13] M. J. Steedman: "A Generative Grammar for Jazz Chord Sequences," *Music Perception*, 2:1 pp. 52–77, 1984.

[14] D. Temperley: *Music and Probability*, The MIT Press, 2007.

[15] K. Yoshii, M. Goto, K. Komatani, T. Ogata and H. G. Okuno: "Drumix: An Audio Player with Real-time Drum-part Rearrangement Functions for Active Music Listening," *IPSJ Digital Courier*, Vol. 3, pp. 137–144, 2007.

MUSIC CUT AND PASTE: A PERSONALIZED MUSICAL MEDLEY GENERATING SYSTEM

I-Ting Liu **Yin-Tzu Lin** **Ja-Ling Wu**
Graduate Institute of Networking and Multimedia, National Taiwan University
{tinaliu,known,wjl}@cmlab.csie.ntu.edu.tw

ABSTRACT

A musical medley is a piece of music that is composed of parts of existing pieces. Manually creating medley is time consuming because it is not easy to find out proper clips to put in succession and seamlessly connect them. In this work, we propose a framework for creating personalized music medleys from users' music collection. Unlike existing similar works in which only low-level features are used to select candidate clips and locate possible transition points among clips, we take song structures and song phrasing into account during medley creation. Inspired by the musical dice game, we treat the medley generation process as an audio version of musical dice game. That is, once the analysis on the songs of user collection has been done, the system is able to generate various medleys with different probabilities. This flexibility brings us the ability to create medleys according to the user-specified conditions, such as the medley structure or some must-use clips. The preliminary subjective evaluations showed that the proposed system is effective in selecting connectable clips that preserved chord progression structure. Besides, connecting the clips at phrase boundaries acquired more user preference than previous works did.

1. INTRODUCTION

A musical medley is a music piece that is composed from parts of existing pieces [22]. It often composed from famous tracks of a specific artist, year or genre. The song excerpts can be played successively with or without cross-fading. In the past, medleys are usually made by professional audio engineers and published by music production companies. Nowadays, more and more music hobbyists create their own medleys from their favourite songs with the help of newly developed audio technologies and publish the results on websites like Youtube. The resulting medley can be used as the background music of personal films and slideshows or non-stopping dance suites. Since each song track just appeared as short clips (usually less than 30 seconds) in a medley, the users can avoid copyright infringement while keeping their personal video with background music of their tastes. Existing editing tools like Goldwave [1] and Audition [2] enable users to cut and connect audio clips at the position they specify directly. However, these tools are not good enough to achieve the pre-described goal. In other words, users still face the following challenging issues: (i) how to choose suitable song excerpts to put together and (ii) how to euphoniously connect them. With the vast amount of digital music, it is not easy for users to find song excerpts that are suitable for playing together. Besides, once the user decides which song excerpts to adjoin, they still need to listen to all of them to decide the position for cutting audio files into clips and connecting them. Furthermore, users may need to manually adjust the tempi and volume levels of the clips if needed to make them smoothly connected. Some previous works provided automatic schemes to create medleys or medley-like mixes. However, they either only focused on the rhythm and beat synchronization issues of adjacent clips or on directly connecting full songs [12, 15]. User preferences of the picked songs, the phrase boundary, and the structure of the medley were rarely considered [14]. We believe that whether music sounds pleasant highly depends on each person's taste. Thus, in this work, a framework for creating personalized musical medleys from users' music collection is proposed, in which the completeness of phrases and chord progression of a song can be preserved. As shown in Figure 1, users specify the structure of the target medley, and optionally select a few song excerpts that should appear at certain positions in the medley, i.e. the darker parts in the figure. Then the system will complete the medley with song excerpts automatically selected from the song collection user provided. The excerpts, automatically segmented by our system, are about four-bars long (∼10 seconds on average.) We treat the medley generation process as an audio version of musical dice game. Various

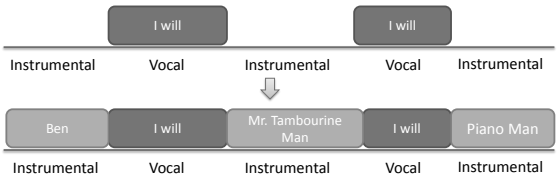


Figure 1. The scenario of the proposed system. Upper part: the user-specified structure and must-used clips. Bottom part: the generated medley by our system.

[1] http://www.goldwave.com/
[2] http://www.adobe.com/products/audition.html

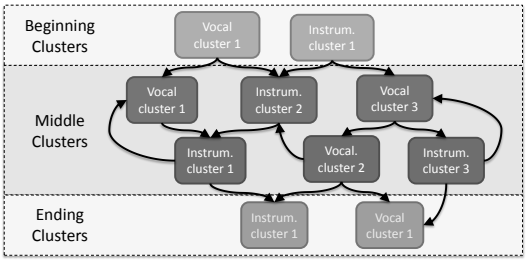


Figure 2. An example of a musical dice graph.

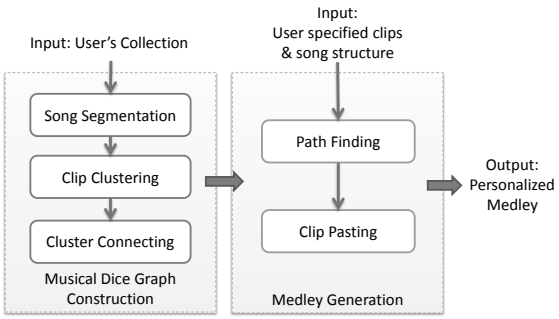


Figure 3. The proposed system framework.

medleys can be generated once we've finished analyzing user's song collection. The flexibility allows us to create medleys based on user needs. We also provide an interactive scheme for users to change selected clips, adjust connecting positions and the overlap ratios between clips if users are not satisfied with the result.

2. RELATED WORK

The problem of medley creation can be divided into two parts: (i) finding proper song excerpts and their orders (including the order of the clips), and (ii) concatenating the selected clips (including finding connecting positions). There are several previous works addressed these issues separately, but few considered them both.

The clip selection issue is similar to the song selection problem in the study of playlist generation. The general approach to these tasks is to select similar songs based on some specific criteria. Commonly used criteria usually fall into one of the following types: (i) meta-data based (e.g. same artist, album, or genre), (ii) content-based (e.g. audio feature similarity [7, 17], key, tempo [5]), and (iii) collaborative filtering based (e.g. occurrence of the songs the the user's friends' past playlists [1]). In our case, we dealt with song excerpts rather than the whole song, i.e, songs are usually interrupted instead of being played to their end. As a result, the required properties used to find adjacent clips are stricter than that of finding adjacent songs. In addition to the global similarities between songs, local audio similarities between song excerpts should also be considered to meet the listeners' expectations for the musical flow.

The works focused on clip concatenation are often found in the studies of DJ tools [3, 9]. In those works, the clips are designated by users, and the authors deal with the concatenation issue only. Some studies of DJ tools also considered both the issues of clip selection and clip concatenation [12, 15]. However, in the aforementioned works, "rhythm similarity" and "beat alignment" are emphasized most because the results are often used for dance. Since a medley is still a kind of music composition, we focus more on the chord euphoniousness of the adjacent clips in this work.

There are also some works similar to us except that they targeted at single song only [16, 25]. In these works, self-similarity matrix of audio feature sequences are used to segment songs [16] or to find cut points [25], and thus the methods cannot be directly applied to our case. The corresponding computational complexity grows dramatically as the number of songs increases. Besides, one can only find near-identical clips with self-similarity matrix. In our case, clips are selected from different songs. If we group the clips based on the same criteria as these works, the resulting medley would be composed only with excerpts taken from the same song, violating the definition of a medley. Therefore, a higher level feature – the chord sequence similarity is adopted in this work. Another similar work was Bernardes et al.'s EarGram [4], but the unit they used to combine music is much shorter than us, i.e. as short as a musical note (usually less than 1 second). Consequently, the music piece produced by EarGram will not keep the phrases of the original songs. It becomes a totally new song rather than a combination of existing songs, and the main interesting feature of medley songs – it is a combination of different songs – is lost.

The work most related to ours is Lin et al.'s "Music Paste" [14]. They dealt with both the clip selection and the clip concatenation issues in their work. However, they did not consider phrase structures while finding connecting positions among clips. Consequently, clips might be cut in the middle of a phrase, resulting in unsmooth transitions. In addition, in Lin et al.' s work, the order of the clips is determined only by matching similar short-time chroma features. It is, therefore, difficult to incorporate user's preferences into the process of clip ordering. In our work, it is easy to put users' preference into consideration, and the chord progression can be preserved better.

3. PROPOSED FRAMEWORK

The proposed framework is inspired by the musical dice game in the symbolic domain [18]. In a musical dice game, players throw dices to randomly "generate" new music from the pre-composed interchangeable musical figures [3] at each bar. Similarly, in our system, we would like to generate medleys by choosing the clip at a given position from a set of interchangeable clips. In order to achieve that, we first analyze the songs in the user-provided collection and cut the songs into self-coherent clips. Then, we group similar clips into clusters. The clips in the same cluster are musically similar to each other, and thus are assumed to be interchangeable. We then connect clusters according to the transition probability calculated from clips' connectivities in the songs they are extracted from. We term it

[3] A short musical phrase [22].

the "musical dice graph" in the rest of this paper. A path on the graph is a version of a medley. Figure 2 shows an example of a musical dice graph. With this graph, we can generate numerous medleys based on user's preferences and the transition probability. Then, the aforementioned two issues in medley creation, "clip selection" and "clip concatenation", can now be turned into "musical dice graph construction" and "medley generation from the walk on the graph", respectively. Figure 3 illustrates the proposed framework. We will discuss the details in the following sections.

4. MUSICAL DICE GRAPH CONSTRUCTION

We divide the construction of the musical dice graph into three steps: song segmentation, clip clustering, and transition probability calculation.

4.1 Song Segmentation

The goal of song segmentation is to divide songs into self-coherent segments. According to [24], phrasing is one of the most important factors to be considered when concatenating different tracks. Interruption happened in the middle of a musical phrase is just as unpleasant and unexpected as that of an oral sentence in a conversation. Therefore, the transition between clips should occur at the boundary of musical phrases. The length of a musical phrase is ambiguous. Here, we define a musical phrase as a self-coherent clip that sounds like ending, usually four or eight-bars long because pop songs usually take a thirty-two-bar form [4] . In order to identify the appropriate timing of transition, i.e., the boundary of musical phrases, we perform singing voice detection on each song track. The reason is that most medleys are composed of pop songs and that the boundary of each singing voice section often corresponds to that of a musical phrase. As a result, we cut the songs into clips based on the boundary of detected vocal segments, and use them as the basic unit for creating medleys.

4.1.1 Singing voice detection

The goal of singing voice detection is to recognize a given audio segment as vocal or instrumental. An instrumental segment is defined as a segment consisting of purely instrumental sounds, e.g. the bridges and the intro. A vocal segment, on the other hand, is defined as a mixture of singing voice with or without background music, as was defined in [21]. Many studies have been done in the past focusing on the task of singing voice detection. The typical procedure performed to solve this problem is to extract short-time audio features, and train a two-class classifier to classify each frame of the audio into instrumental or vocal class. Common classifiers used are Gaussian Mixture Models (GMM), Hidden Markov Models (HMM) and their variants, and Support Vector Machines (SVM). Frequently used features include Mel Frequency Cepstral Coefficients (MFCC), Linear Prediction Coefficients (LPC), and Zero-Crossing Rate (ZCR) [19]. Often, temporal smoothing techniques are also performed to constrain the length of each vocal and instrumental segment afterwards to prevent over-segmentation [23]. Here, we employ beat-synchronized MFCC as audio features and HMM as the classifier. The idea of using beat-synchronized features comes from the phenomenon that voices are more likely to join the accompaniment at beat times [13]. MFCCs are first extracted during the training phase. The MFCCs within a musical beat act as the observed sequence for an HMM classifier. The beats are detected by using BeatRoot [6], a state-of-the-art beat-detection tool. In the testing phase, each beat in a test song can then be classified as vocal or instrumental. Consecutive vocal/instrumental beats can then be group into vocal/instrument clips, i.e. the unit we used to create medleys. In addition, in order to avoid over-segmentation, we apply a median filter of 8 beats long (i.e. about 4 seconds) to the singing voice detection result.

[4] also known as AABA form [22]

4.2 Clip Clustering

After dividing songs into clips, we group these clips according to the similarities among them. There are numerous metrics for measuring similarity among clips, e.g. volume, rhythm, timbre, tonality, genre, etc.. Here we choose the chord sequence similarity because in the musical dice game, the candidates of musical figures are often chord-similar or dominant-pitch-similar. The clip clustering process can be divided into three steps: chord detection, chord sequence similarity computation and clustering.

4.2.1 Chord detection

First, we detect the chords of all songs with Harmony Progression Analyzer (HPA) [20], a state-of-the-art chord estimation system. The number of possible chords estimated is limited to 25, which are 12 major chords, 12 minor chords, and no-chord for the periods of silence. Then, beat-synchronized chord sequences are extracted for each clip by aligning the chord estimation results with the detected beats of the clips.

4.2.2 Chord Sequence Similarity Computation

We then measure the chord sequence similarity between each clip pairs. A dynamic-time-warping (DTW)-like approach proposed by Hanna et al. [11] is used in our system to measure the edit distance between two given chord sequences. In order to better capture the harmonic relationship between the sequences, the substitution score used to calculate the edit distance varies with the consonance of the interval between the two given chord roots, as Hanna et al. [10] proposed. Consonant intervals are the intervals that sound stable [22], and chords whose roots form a consonance are given higher substitution scores.

4.2.3 Clustering

Given the similarity score between each pair of the chord sequences, we could then cluster the clips by hierarchical clustering. The vocal clips and the instrumental clips are clustered separately. Also, the clusters are categorized into three types according to the positions of the clips in the

song. In other words, we would have six types of clusters in total: beginning, ending and middle clusters, each of which can be either vocal or instrumental. Note that each cluster type also contains several clusters.

4.3 Cluster Connecting

Then, we connect the clusters according to the transition probability defined as in Equation (1). For two arbitrary clusters A and B, the transition probability $P(B|A)$ is defined as the proportion of clips in cluster A that originally concatenate with clips in cluster B, that is:

$$P(B|A) = \frac{|S|}{|A|},\ S = \{(a,b)|a \in A, b \in [B \cap N(a)]\} \tag{1}$$

where a and b stands for an arbitrary clip, and $N(a)$ is the set of clips that appeared next to clip a in the original song.

5. MEDLEY GENERATION

After the musical dice graph is constructed, we can then compose medleys by finding a path on the graph and concatenating them according to the path.

5.1 Path Finding

Now we find a path on the musical dice graph with the maximum transition probability along the path. First, we picked candidate clusters according to user-specified medley structure. For example, the user may designate the structure as I→V→I→V→I, where "I" and "V" stand for instrumental and vocal clips, respectively. Then, we choose clusters conforming to the types the user specified in the structure as candidate clusters. That is, instrumental-beginning clusters for the first clip slot, and vocal-middle clusters for the second, and so forth, for the previous example. If the user has specified the third slot to be clip a, then the cluster that clip a belongs to is chosen. Then, we use Viterbi algorithm [8] to find a path of candidate clusters with maximal transition probability, where the candidate clusters at each slot are regarded as the states in the algorithm. After determining the clusters, we randomly select one clip per cluster since clips in the same cluster are interchangeable. The selected clips then compose the final medley.

5.2 Clip Pasting

Once the clips are selected along the chosen path on the graph, we then concatenate them smoothly. For each selected clip, we adjust the tempi with Lin et al.'s method [14]. The algorithm takes just noticeable difference (JND) between the tempi of two clips into account, adjusting the tempi gradually to lessen listeners' discomfort. After adjusting the tempi, the volumes of the clips are then normalized, and the clips are finally concatenated by using a short-length logarithmic cross-fade in between.

6. EXPERIMENT

In this section, we conduct three experiments, two subjective and one objective, to evaluate the three components of the system separately: the performance of the song segmentation method and the clip selection mechanism, and the quality of the transition. The experimental data set contains 100 best-selling English songs from 1950s to 1990s collected from Youtube [5] , and the genres include folk, pop, jazz, musical and movie soundtrack, the lengths of which range from one and a half minutes to five and a half minutes. Two sets of annotation are built manually for each track: the singing voice annotation and the musical phrase annotation. The former was used to perform singing voice detection, while the latter is used as the ground truth of boundary detection task. All songs in this dataset contain singing and instrumental parts, i.e. none of them is a pure instrumental nor A cappella music.

Method	G-to-T	T-to-G	Pre.	Rec.	F-meas.
GT	0.00	0.00	0.73	0.78	0.75
GMM	2.85	10.29	0.20	0.17	0.18
HMM	2.55	**2.66**	**0.25**	**0.30**	**0.27**
DTM	**2.38**	5.13	0.23	0.18	0.20

Table 1. The song segmentation result.

6.1 Effectiveness of Song Segmentation

This experiment measures the effectiveness of using singing voice detection techniques to detect phase boundaries. We compare three kinds of segmentation scheme: singing voice detection with GMM and HMM classifier (which will be referred to as GMM and HMM later in this paper), and Dynamic Texture Model (DTM), a state-of-the-art song segmentation method proposed by Barrington et al. [2]. We evaluate the boundary detection performance by precision, recall and F-measure, where a detected boundary is regarded as a "hit" if its time interval between the nearest true boundary lies within a certain threshold. We also calculate two median time values: guess-to-true (G-to-T) and true-to-guess (T-to-G), which are the median time interval between each detected boundary and the closest true boundary and vice versa, as defined in [2]. For singing voice detection, all audio files are 22050 Hz-sampled, and 26 MFCCs are extracted every 256 samples, overlapping by half. Singing voice detection by GMM and HMM were performed respectively with 5-fold cross validation, and the number of mixtures of GMM and the number of hidden states of HMM are both empirically set to 5. The hit time interval was set to 0.5 second as in [2].

The results are presented in Table 1. The song segmentation performances using ground-truth singing voice annotation are also listed as a reference, denoted as GT. From Table 1, HMM and DTM outperform GMM obviously, while HMM performs approximately as good as DTM. In this work, we chose HMM over DTM for its simplicity and speed during the singing voice detection testing phase. Different parameters, such as the length of the median filter used to prevent over-segmentation, may influence the performance, and therefore introduce a trade-off

[5] Song names and URLs are listed at: `http://goo.gl/khjtB`.

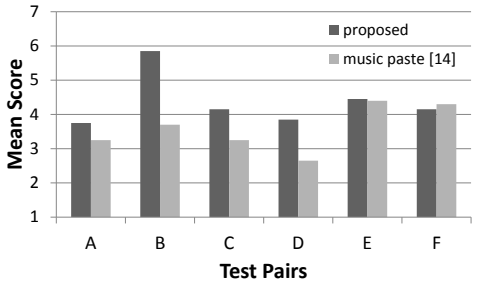


Figure 4. The user evaluation results of clip concatenation.

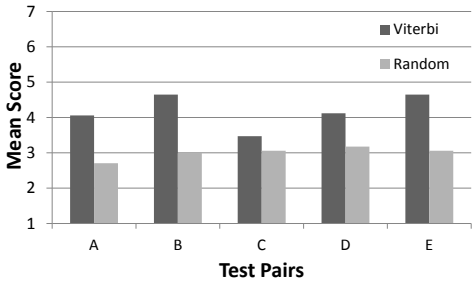


Figure 5. The results of user evaluation on clip selection.

between precision and recall. In this work, the goal of segmentation is to avoid interruption in the middle of a phrase when pasting songs. Thus, when choosing between different parameters, we would favour those results in higher precision.

6.2 Effectiveness of Clip Concatenation

To see whether segmentation really helped on song concatenation, we compare 6 pairs of medleys. Each pair contains two medleys, and each medley is composed of two song clips. The two song clips used in the medleys of one test pair are the same. For one medley in each pair, the transition of clips happens immediately and directly at the end of a phrase of the first clip (the proposed method). For the other medley, the transition point and cross-fade length are decided with Lin et al. [14]'s method, which is the location where the short-term chroma features of two clips matched the best and are in the middle of a phrase rather than the boundary. In this and the following experiments, we used human-labeled annotation for musical phrase information because we wish the results wouldn't be biased by inaccurate segmentations. The generated medleys used in the experiments can be found in `http://goo.gl/khjtB`.

We ask 20 users to listen to and compare these test pairs. Each pair of medleys are played twice. Users are asked to report how these two medleys sound according to transition smoothness between two adjacent clips. All questions are designed on a 7-point Likert scale. Figure 4 shows the score of each test pair. In 5 out of 6 pairs, the mean score of the medleys generated by the proposed method are higher than the one created by Lin et al.'s method.

6.3 Effectiveness of Clip Selection

This experiment is designed to measure the effectiveness of selecting clips based on their chord sequence transition probability. The test set for this experiment contains 5 pairs of medley. The number of transitions and the structure of the two medleys in a pair are the same, and the beginning and the ending clips of the medleys are specified to be the same. For one medley in each pair, the chosen clips and their order are decided by the Viterbi algorithm along with the calculated transition probability. For the other medley, clips are selected randomly.

We asked 17 participants to listen to and evaluate generated medleys. Each pair of medleys are played twice. Figure 5 illustrates the score of each test pair. We performed

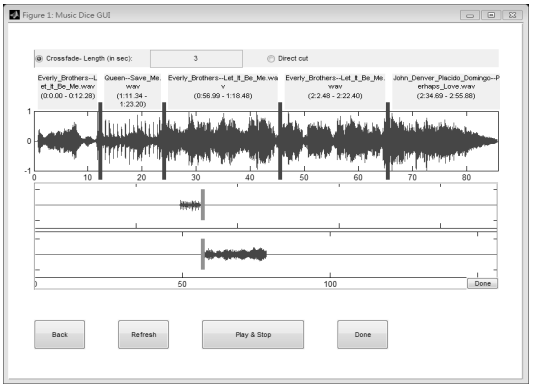


Figure 6. A screenshot of the proposed system.

pairwise t-test to analyze the results. Overall speaking, the mean score of medleys composed with Viterbi algorithm is significantly higher than the one composed with random clip selection ($p < 0.05$). If the 5 test sets are evaluated separately, the mean score of medleys composed with Viterbi is significantly higher than randomly generated one in 4 out of 5 test pairs (Pair A, B, D, and E, $p < 0.05$). The result shows that the proposed clip selecting scheme is effective in selecting clips that sound pleasant when they are concatenated and played in sequence.

7. USER INTERFACE

The quality of the generated medley is highly subjective and depends on users' tastes. For example, the actual boundary of a vocal phrase that fades out gradually is hard to determine and there is no correct answer. The cross-fade length between two clips is also subjective given that some may prefer longer overlaps because of its smoothness, while others may prefer shorter or no cross-fade to avoid blur of sounds. In order to better satisfy different user needs, we developed a simple GUI, as shown in Figure 6. With the interface, users could specify parameters, modify the segmentation result and change the selected clips.

8. CONCLUSION AND FUTURE WORK

In this work, we propose a framework for creating personalized music medleys from users' music collection. Borrowing the idea of musical dice, we construct a music dice graph with self-coherent clips that are segmented by singing voice detection. After graph construction, the system is able to generate numerous medleys, automatically. The completeness of phrases in a song is retained, and harmonic contents

are considered to improve the quality of the generated medley. Since the quality of a medley is highly dependent on listeners' tastes, we also provide a simple feedback mechanism that enables users to specify song clips, modify the transition positions and the length of cross-fades to better fit their preference. The objective experimental results demonstrated that song segmentation performed with singing voice detection is comparable with state-of-the-art methods. In addition, the preliminary subjective evaluations show that connecting clips at phrase boundary acquired more user preference than previous works did. We also demonstrated the proposed Viterbi-based algorithm is effective in selecting harmonious clips.

Many aspects of our system can be improved. First, more clip similarity measures can be introduced during clip clustering, e.g. rhythm, volume, genre, mood, etc.. Second, the number of cluster types can be extend into , for example, "first half verse", "second half chorus", "bridge", etc.. Third, song segmentation with singing voice detection restricted this work to vocal songs. In the future, we will delve into other music segmentation methods that are able to recognize phrases in instrumental music. Finally, automatic separation of background music from foreground singing voice may enable us to create and add intermediate bridges, allowing more flexibility in medley generation.

9. ACKNOWLEDGMENTS

Special thank to Professor Jyh-Shing Roger Jang for constructive discussion with us, and the course members of MSAR for helping annotate the songs.

10. REFERENCES

[1] C. Baccigalupo and E. Plaza: "Case-based Sequential Ordering of Songs for Playlist Recommendation," *LNCS*, vol. 4106, pp. 286–300, 2006.

[2] L. Barrington, *et al.*: "Modeling Music as a Dynamic Texture," *IEEE Trans. ASLP*, vol. 18, no. 3, pp. 602–612, 2010.

[3] S. Basu: "Mixing with Mozart," *Proc. ICMC*, 2004.

[4] G. Bernardes, *et al.*: "EarGram : an Application for Interactive Exploration of Large Databases of Audio Snippets for Creative Purposes," *Proc. CMMR*, June, pp. 19–22, 2012.

[5] L. Chiarandini, *et al.*: "A System for Dynamic Playlist Generation Driven by Multimodal Control Signals and Descriptors," *Proc. MMSP*, 2011.

[6] S. Dixon: "Evaluation of the Audio Beat Tracking System BeatRoot," *J. New Music Res.*, vol. 36, no. 1, pp. 39–50, 2007.

[7] A. Flexer, *et al.*: "Playlist Generation Using Start and End Songs," *Proc. ISMIR*, pp. 2–7, 2008.

[8] J. G. D. Forney: "The Viterbi Algorithm," *Proc. of the IEEE*, vol. 61, no. 3, pp. 302–309, 1973.

[9] G. Griffin, *et al.*: "Beat-Sync-Mash-Coder: a Web Application for Real-Time Creation of Beat-Synchronous Music Mashups," *Proc. ICASSP*, pp. 2–5, 2010.

[10] P. Hanna, *et al.*: "On Optimizing the Editing Algorithms for Evaluating Similarity Between Monophonic Musical Sequences," *J. New Music Res.*, vol. 36, no. 4, pp. 267–279, 2007.

[11] P. Hanna, *et al.*: "An Alignment Based System for Chord Sequence Retrieval," *Proc. JCDL*, p. 101, 2009.

[12] H. Ishizaki, *et al.*: "Full-Automatic DJ Mixing System with Optimal Tempo Adjustment based on Measurement Function of User Discomfort," *Proc. of ISMIR*, pp. 135–140, 2009.

[13] Y. Li and D. Wang: "Separation of Singing Voice From Music Accompaniment for Monaural Recordings," *IEEE Trans. ASLP*, vol. 15, no. 4, pp. 1475–1487, 2007.

[14] H.-Y. Lin, *et al.*: "Music Paste: Concatenating Music Clips Based on Chroma and Rhythm Features," *Proc. ISMIR*, Kobe, 2009.

[15] Q. Lin, *et al.*: "Music Rhythm Characterization with Application to Workout-Mix Generation," *Proc. ICASSP*, pp. 69–72, 2010.

[16] Z. Liu, *et al.*: "Adaptive Music Resizing with Stretching, Cropping and Insertion," *Multimedia Syst.*, 2012.

[17] B. Logan: "Content-based Playlist Generation: Exploratory Experiments," *Proc. ISMIR*, pp. 2–3, 2002.

[18] G. Loy: *Musimathics*, vol. 1, pp. 295–296,347–350, The MIT Press, USA, 2006.

[19] N. C. Maddage, *et al.*: "Content-based Music Structure Analysis with Applications to Music Semantics Understanding," *Proc. ACM MM*, p. 112, 2004.

[20] Y. Ni, *et al.*: "An End-to-End Machine Learning System for Harmonic Analysis of Music," *IEEE Trans. ASLP*, vol. 20, no. 6, pp. 1771–1783, 2012.

[21] T. L. Nwe, *et al.*: "Singing Voice Detection in Popular Music," *Proc. ACM MM*, p. 324, 2004.

[22] D. M. Randel: *The Harvard Dictionary of Music*, Belknap Press, 2003.

[23] L. Regnier and G. Peeters: "Singing Voice Detection in Music Tracks Using Direct Voice Vibrato Detection," *Proc. ICASSP*, pp. 1685–1688, 2009.

[24] S. Webber: *DJ Skills: The Essential Guide to Mixing and Scratching*, Focal Press, 2007.

[25] S. Wenger and M. Magnor: "Constrained Example-based Audio Synthesis," *Proc. ICME*, 2011.

A META-ANALYSIS OF THE MIREX STRUCTURE SEGMENTATION TASK

Jordan B. L. Smith
Queen Mary, University of London
jblsmith@eecs.qmul.ac.uk

Elaine Chew
Queen Mary, University of London
elaine.chew@eecs.qmul.ac.uk

ABSTRACT

The Music Information Retrieval Evaluation eXchange (MIREX) serves an essential function in the MIR community, but researchers have noted that the anonymity of its datasets, while useful, has made it difficult to interpret the successes and failures of the algorithms. We use the results of the 2012 MIREX Structural Segmentation task, which was accompanied by anonymous ground truth, to conduct a meta-evaluation of the algorithms. We hope this demonstrates the benefits, to both the participants and evaluators of MIREX, of releasing more data in evaluation tasks.

Our aim is to learn more about the performance of the algorithms by studying how their success relates to properties of the annotations and recordings. We find that some evaluation metrics are redundant, and that several algorithms do not adequately model the true number of segments in typical annotations We also use publicly available ground truth to identify many of the recordings in the MIREX test sets, allowing us to identify specific pieces on which algorithms generally performed poorly and to discover where the most improvement is needed.

1. INTRODUCTION

MIREX is a highly valued event in the MIR community. Modeled in large part on the evaluations conducted by the Text Retrieval Conference, its role is to establish benchmarks of performance and to allow the community to compare the efficacy of different approaches [5]. MIREX also stimulates competition and helps to drive innovation in areas that the community feels are valuable.

At previous ISMIR conferences, the problems facing MIREX have been a frequent topic of discussion. Some of these are challenges for any evaluation: e.g., the high cost of generating ground truth, and the legality of sharing the music in most test collections [5]. However, [14] points out some issues specific to MIREX, including the problem of hidden data: namely, the results published by MIREX do not identify the songs used in the evaluation or provide metadata other than a general description of the corpus. For example, in the Audio Key Detection task, participants can see how often their algorithm makes different kinds of mistakes—e.g., being off by a major fifth or by a relative key—but cannot see on which pieces their algorithm made the mistakes, or see other information related to the piece, such as the composer, instrumentation, or key.

In his call for more meta-evaluation, Urbano points out that the hidden nature of the MIREX testing data impedes the learning phase of algorithm development [20]. That is, although MIREX evaluations allow the community to compare the performance of state-of-the-art algorithms, it is difficult to learn from the mistakes of the algorithms without any information about the ground truth. For example, although a researcher could learn that their key-detection algorithm makes many parallel key errors, they would have no idea in what situations their algorithm is more likely to make these kinds of errors.

While improving MIREX by creating new ground truth or copyright-free datasets would be expensive, it would only require a change in publishing policies for MIREX to release more detailed evaluation data. In fact, such a change in policy was recently made for the Audio Structural Segmentation (SS) task: in 2012, MIREX posted not only the performance of each algorithm on each piece in the datasets, but the output of the algorithms and the matching annotation. This allows the community for the first time to look more deeply into the large-scale MIREX evaluation, and potentially identify patterns in the performance of the algorithms in order to improve them..

Urbano also recognized the need for more meta-evaluation in the MIR community [20]. Meta evaluation, or the analysis of evaluation systems, is a popular subject in the text retrieval community (e.g., [4]) but has received less attention in MIR. A meta-evaluation investigates whether an evaluation experiment accomplishes its purpose: do the metrics measure the desired quantities or qualities? Is the experiment fairly and effectively estimating the relative quality of different algorithms? With the release of ground truth data in the 2012 SS task, we can attempt to answer some of these questions.

The SS task was introduced to MIREX in 2009, and by then had already been the subject of a meta-evaluation of performance metrics: the merits and shortcomings of over a dozen previously used metrics was discussed in [10], which proposed an additional two. In [6], an extended evaluation of the algorithms submitted to the 2011 MIREX SS task, the authors accessed the algorithms submitted to MIREX in 2010 and tested them on the newly available SALAMI corpus [19]. They compared the algorithms' performance on the large- and small-scale segmentation data, on the different genres within SALAMI and on the different MIREX datasets. With the new MIREX task came a greater interest in generating test collections and designing appropriate methodologies: [13] made recommendations that were adapted by [19] in the creation of the SALAMI dataset, while [3] conceived an alternative methodology, leading to the INRIA collections of annotations.

In this article, we study the results of the 2012 MIREX SS task for these two purposes: first, to learn about the failure modes of the current state of the art algorithms, and second, to appraise the success of the task so far. In Section 2, we review the test sets, algorithms, and evaluation metrics involved in the SS task. In Section 3, we conduct a correlation analysis of the SS evaluation results to find which metrics and basic features could be measuring the same quantities. In Section 4, we show how the ground truth released in 2012 allows us to identify many of the anonymous pieces in the published results, and we survey some of the easiest and hardest songs to analyze.

2. THE STRUCTURE SEGMENTATION TASK

In this section we summarize the materials of the 2012 SS task: the algorithms submitted, the test data used, and the evaluation metrics calculated.

2.1 Algorithms

Five algorithms were evaluated in the 2012 MIREX SS task: KSP [8], MHRAF [11], OYZS, SBV [16] and SMGA [17]. KSP was submitted with four different parameter settings, and two versions of SMGA were submitted, resulting in nine algorithm runs. The algorithms are outlined in abstracts published with the MIREX results, although no abstract is posted for OYZS, and the difference between the two versions of SMGA is not specified in [17]. The four algorithms with abstracts are briefly compared below.

Among the most important differences between the algorithms is each one's hypothesis about what musical structure is. Using the terminology of [12], the KSP algorithm uses the states hypothesis, which holds that sections are musically homogenous and distinct from one another. MHRAF uses the sequences hypothesis, which holds that sections are defined by distinct sequences. SMGA uses a combined approach, employing a novel feature representation that captures information about long-term repetitions and short-term homogeneity. Finally, SBV is based on the novel "system and contrast" theory of musical structure described in [2]. The algorithm expects that sections will consist of 4 groups of 4 measures, with the fourth group either conforming to or contrasting with the system of musical relationships established by the previous three groups. Other important differences between the algorithms are:

- MHRAF, SBV and SMGA all use harmonic features, while KSP uses a combination of harmonic and timbral features.
- All the algorithms except for MHRAF estimate boundaries first and then estimate segment labels; the MHRAF algorithm detects repetitions first and uses these to define the segmentation.
- SBV is the only algorithm to employ a beat-detection step to align the analysis frames.
- The parameters of the SBV algorithm were set from a test on the INRIA annotations of the RWC database, and a version of SMGA was previously tested on the Beatles and RWC datasets [18].

2.2 MIREX data

The 2012 SS task was evaluated using three datasets:

- MIREX09: A set of 297 pieces introduced in 2009, with annotations taken from the EP [22], Isophonics [24] and TUT collections [26]. According to the analysis in Section 4, of the 343 distinct annotations in these collections, the MIREX test set includes pieces by The Beatles, Carole King, Michael Jackson, and Queen.
- MIREX10: The RWC popular music database, which consists of 100 Japanese pop tunes, with annotations provided by AIST [7, 21] and by INRIA [3, 23]. The INRIA annotations only give boundaries and were first tested in 2010; the AIST annotations, introduced to MIREX in 2011, also provide section labels. INRIA annotations with segment labels were recently introduced [2].
- MIREX12: The SALAMI data [25], introduced to MIREX 2012, and consisting of approximately 859 pieces, over 500 of them with annotations by two listeners [19]. The pieces include popular, classical, world and jazz recordings, publicly available live recordings taken from the Internet Archive, and some pieces borrowed from the Isophonics and RWC collections.

Each of the collections were produced by a small number of listeners (fewer than 10 each) annotating the structure they perceived, but differed in their methodology. The Beatles annotations were adapted from musicologist Allan Pollack's descriptions [15]. Many of the RWC annotations benefitted from a beat-tracked grid computed from a click track, and only the "obvious" sections were annotated, meaning there are some unannotated gaps in the descriptions [7]. Most SALAMI pieces were annotated by two listeners, and its annotations have separate layers for musical function, similarity at two timescales, and leading instrumentation. Finally, the INRIA annotations indicate boundaries according to a more concrete definition of segments [3].

2.3 Evaluation metrics

For the SS task, MIREX published 14 of the most common metrics reported in the literature: pairwise retrieval (precision pw_p, recall pw_r and f-measure pw_f), proposed by [9]; over- and under-segmentation scores (S_O and S_U, respectively), proposed in [10]; Rand index (R), a metric for comparing partitions of data first used for structural segmentation in the 2009 MIREX task; boundary retrieval with a specified tolerance of 3 seconds (precision b_{p3}, recall b_{r3} and f-measure b_{f3}) or 0.5 seconds ($b_{p.5}$, $b_{r.5}$, $b_{f.5}$); and the median distance from each true to the nearest claimed boundary (*mt2c*) and vice versa (*mc2t*). Since the output of each algorithm is also available [27], it is possible to evaluate the algorithms with metrics not published by MIREX. We did so for 5 other metrics: average cluster purity (*acp*) and speaker purity (*asp*), and their summary metric called the K-measure (K), mentioned by [10] as a potential metric in MIR; and the fragmentation and missed-boundary scores (f and m, respectively) used by [1].

Some of these metrics evaluate boundary estimation, and the others evaluate the grouping of sections. Boundary estimation measures either penalize over-segmentation, under-segmentation, or both. Similarly, grouping metrics either penalize the estimation of spurious similarity relationships, the omission of true similarity relationships, or both. Table 1 summarizes the general purpose of the different metrics. Although each metric is

Purpose of the metric	Boundary metrics	Label metrics
Summary metric	b_{f3}, $b_{f.5}$	pw_f, *R*, *K*
Penalize over-segmentation (spurious boundaries and omitted similarity relationships)	b_{p3}, $b_{p.5}$, *1–f*, *mc2t*	pw_r, S_O, *asp*
Penalize under-segmentation (omitted boundaries and spurious similarity relationships)	b_{r3}, $b_{r.5}$, *1–m*, *mt2c*	pw_p, S_U, *acp*

Table 1. Summary of the metrics by evaluation purpose.

distinct, we expect the metrics in a single group will agree with each other.

3. CORRELATION ANALYSIS

With so many metrics we would like to know whether the metrics in fact measure different things. This problem can be posed in two ways: first, do the metrics differ in how they rank the algorithms? And second, do they differ in how they rank the difficulty of analyzing each recording?

Since our data (the evaluation metrics) are not normally distributed, we compute Kendall's τ rather than the Pearson correlation. Consider all pairs of items in two ranked lists; if p is the probability that the lists agree on how to rank a pair, then $\tau = p - (1 - p)$ and ranges from $\tau = 1$ for identical rankings to $\tau = -1$ for reversed rankings. With independent lists of rankings, τ is a random variable with mean 0, and we can estimate the precision with which τ has been measured. In all the correlation plots that follow, we use the simple, conservative Bonferonni correction to determine which values for τ are significantly non-zero. Saying whether a given value of τ indicates a "strong" or "weak" correlation remains a subjective decision; we arbitrarily deem $|\tau| \geq 0.8$ as a strong correlation (for positive values, this means that two lists rank at least 9 in 10 pairs of items the same way), $|\tau| \geq 0.33$ as a weak correlation (2 in 3 pairs are ranked the same), and $|\tau| < 0.33$ as no correlation.

3.1 Correlation among metrics

The agreement between the metrics when the algorithms are ranked according to the median grade achieved is shown in Figure 1a. The trio of evaluation measures not used by MIREX (*K, asp, acp*) rank the algorithms very similarly to the pairwise retrieval metrics (pw_f, pw_r, and pw_p, respectively), supporting their exclusion from MIREX evaluations for being redundant. We expected each metric to be most similar to other metrics measuring the same type of error (under-segmentation, over-segmentation, and both together), but instead we find the over-segmentation metric S_O is more similar to the summary metrics pw_f and *K*; and the intended summary metric *R* is more similar to the under-segmentation metrics S_U, pw_p and *acp*.

We can also see whether the ranking of the recordings according to difficulty depends on the metric. Here the results (see Figure 1b) conform more to our expectations: *K, asp* and *acp* are again found to be redundant; S_U and S_O are grouped appropriately with pw_r and pw_p, but *R* now resembles an over-segmentation metric.

Performing the same analysis with the boundary evaluation metrics, we again find that related metrics are somewhat redundant: bp_3, *1-f*, and *mc2t* are highly intercorrelated, as are $br_{.5}$, br_3, *1-m* and *mt2c* (see Figure 2a). Interestingly, $bp_{.5}$ does not correlate with bp_3 or the other over-segmentation metrics; locating boundaries to within 3 seconds and to within 0.5 seconds are perhaps two distinct skills. This discrepancy is not true for $br_{.5}$ and br_3, and hence is probably the cause of the surprising finding that the boundary *f*-measure summary metrics (bf_3 and $bf_{.5}$) also do not intercorrelate significantly.

When ranking the recordings (Figure 2b), the groups of metrics (summary, over- and under-segmentation) are each consistent, but the summary metrics are also similar to the over-segmentation ones. Does this indicate that the algorithms are better at boundary precision than recall? In fact, the opposite is the case: mean bp_3 and $bp_{.5}$ were simply consistently worse for all algorithms.

Lastly, while there is insufficient space to demonstrate it, the findings of this section were consistent across the datasets, albeit with some variation in significance levels.

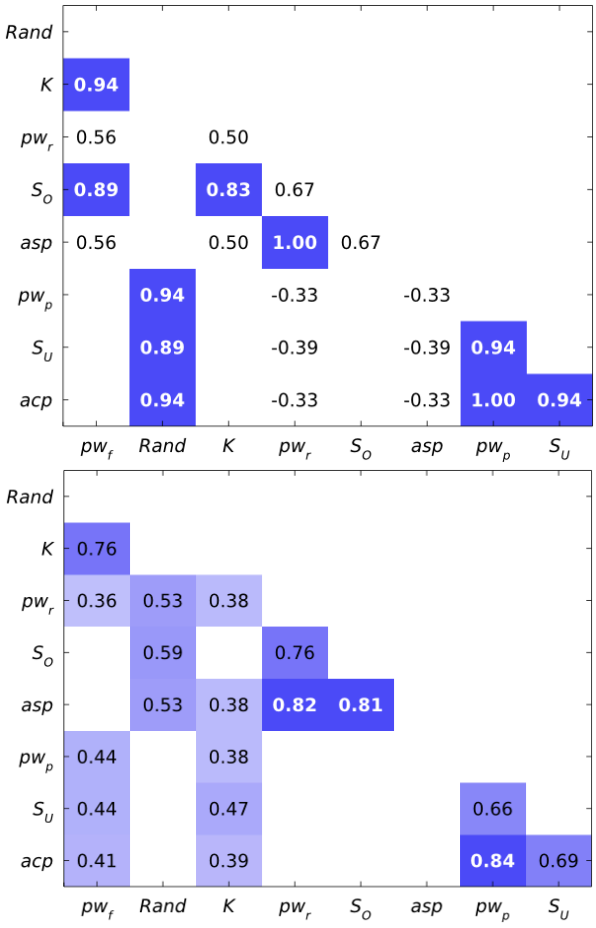


Figure 1a (top). Agreement (Kendall's τ) between rankings of algorithms (according to median across all recordings) by different labelling metrics. All values of $\tau \geq 0.33$ are plotted. Shaded backgrounds indicate significance; bold values indicate strong agreements.
Figure 1b (above). Agreement in the ranking of recordings by different labelling metrics.

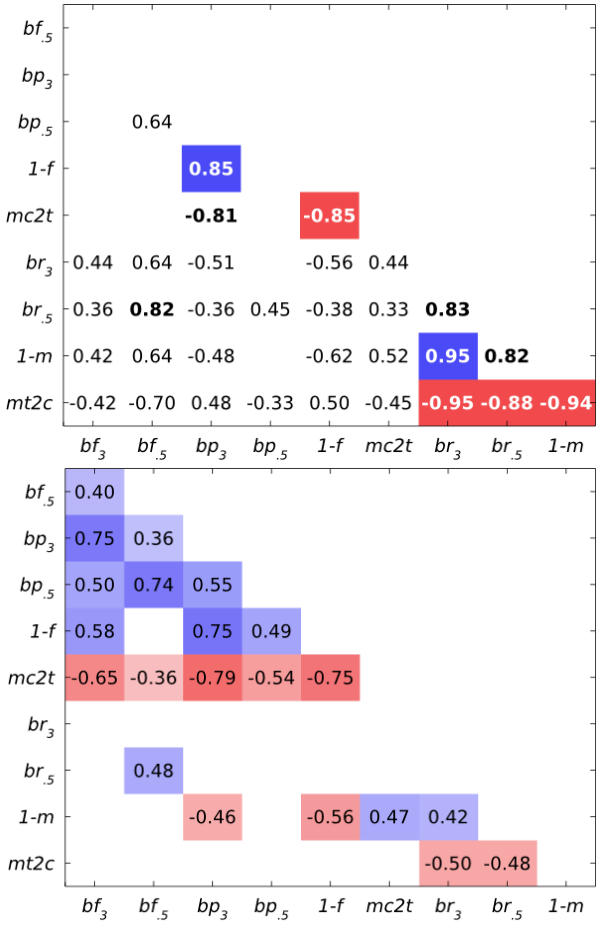


Figure 2a (top). Agreement in the ranking of algorithms by different boundary metrics.
Figure 2b (above). Agreement in the ranking of recordings by different boundary metrics.

3.2 Correlation with ground truth properties

The preceding analysis uses only the evaluation data, not the additional ground truth information made available for the 2012 SS task. With the ground truth we can push the correlation one step further and check whether there are simple properties of the annotated and estimated descriptions that strongly influence the evaluation metrics. For example, it could be that the algorithms simply find longer songs to be more difficult to analyze.

We tested the correlation between all the preceding metrics and the length of the recording (*len*), as well as ten other properties. Four are properties of the annotation: number of segments (ns_a), number of unique labels (nl_a), mean segment length (msl_a) and the number of segments per label ($nspl_a$). The next four are the same properties for the estimated description (ns_e, nl_e, msl_e, $nspl_e$). We also take the number of extra segments estimated (ns_e–ns_a) as a "direct" over-segmentation measure for boundaries (*ob*), and likewise the number of extra labels (nl_e–nl_a) as the over-segmentation measure for labels (*ol*).

The correlation between these properties reveals an interesting mismatch between the algorithms and the annotations with regard to the mean segment length. In the annotations, song length correlates significantly with mean segment length ($\tau = 0.37$), but hardly at all with the number of segments (0.22). The pattern is reversed among the algorithm outputs, where song length barely correlates with mean segment length (0.25) but strongly with the number of segments (0.72). It appears that the algorithms do not model how listeners tend to identify a number of sections that is stable across most pieces: while the middle half of the values of ns_a ranges from 7 and 13 segments, the middle values for ns_e for most algorithms range from 11 to 20 segments. The two exceptions are MHRAF and OYZS, for which both msl_e and ns_e match the distributions seen in the annotations.

This shortcoming of the algorithms is reflected in the strong dependence of boundary estimation metrics on the mean segment length in the annotation (msl_a): worse values of b_p, *1-f* and *mc2t,* all of which punish spurious boundary estimation, are correlated with longer annotated segments.

Figure 3 also shows that the label evaluation metrics seem more sensitive to the number of unique labels in the annotation (nl_a). When nl_a is high, so are *asp* and S_O (these reflect over-segmentation errors, which are more difficult to make with more fine-grained annotations), while pw_p and *acp* are reduced, indicating a susceptibility to under-segmentation errors. This dependence suggests that algorithms have difficulty estimating the number of unique segment types heard by a listener.

4. IDENTIFYING MIREX RECORDINGS

In the 2012 SS task, ground truth and the estimated analysis of each algorithm were provided for each piece.

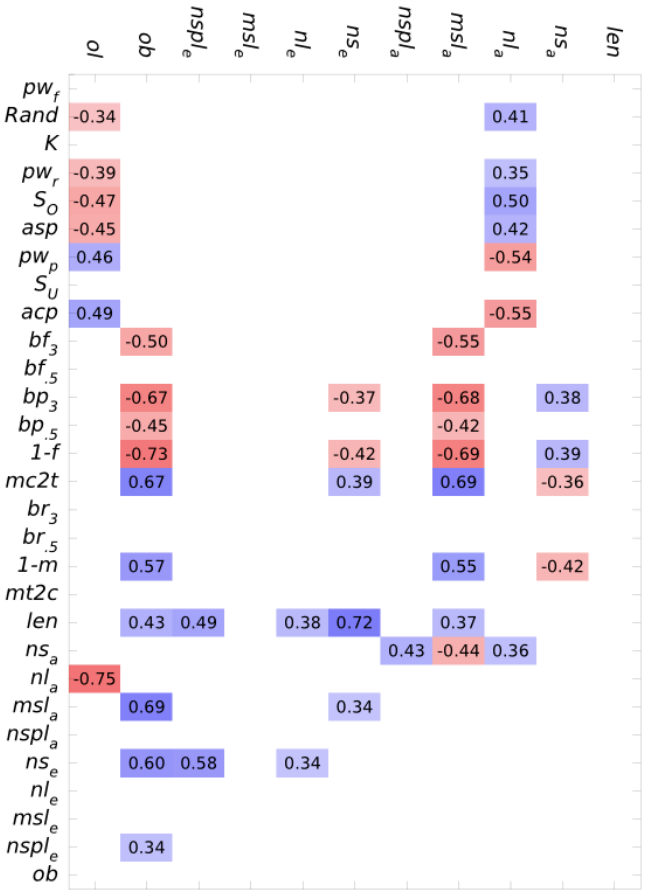


Figure 3. Agreement in the ranking of recordings between metrics and properties of the annotated and estimated descriptions.

Since the ground truth collections used by MIREX are publically available, we can try to identify the recordings in the evaluation by matching the anonymized ground truth published by MIREX with public ground truth.

Before we identify the recordings, we acknowledge that there are advantages to keeping test data private: it is difficult for the designers of algorithms to overfit to hidden data, which means the same data can be reused in successive years; this is useful since ground truth is expensive to create. However, to learn from an evaluation with private data would require the task moderators to conduct meta-evaluation, which is also costly. Moreover, for the SS task, most of the annotation data is already public, and the Beatles and RWC datasets are already widely distributed—indeed, RWC was designed in order to be distributable at cost, without regard for copyright issues. Hence the main advantages of private data do not apply to this task.

The collections used in the SS task are summarized in Table 2. We downloaded all the publicly available annotations for these sources [21–26] to compile a grand public corpus, as well as all MIREX output and annotation for the SS task [27]. For every anonymous MIREX annotation, we searched for public annotations where the lengths of the pieces differed by less than 15 seconds, and computed the boundary *f*-measure between them. If the boundary *f*-measure exceeded 0.99, we assumed a match was found. Checking many of the matches informally, it was clear the match was correct.

The number of annotations in the four test collections is provided in Table 1, as well as the number of pieces that were positively matched with an existing annotation. The greatest number of pieces missed were in the SALAMI collection. This is to be expected since half of the SALAMI data remains private.

Associating the MIREX results with actual recordings allows us to search for possible commonalities between the recordings that were "easiest" and "hardest" to annotate. The piece with the highest median pw_f is The Beatles' "Her Majesty," a 30-second song with just one section. When a song has just one section, any algorithm is guaranteed to get pairwise precision of 1, and the only boundaries in the song are within 3 seconds of its beginning and end, ensuring boundary recall of 1 as well. The next-best Beatles song, "I Will", is an instance where both the states and sequences hypotheses apply well: the repeating sections are relatively homogeneous, but contain distinct harmonic sequences. Also, like "Her Majesty," the song is short and contains few sections, reducing the chance of under-segmentation.

MIREX dataset	Dataset contents	Number of pieces	Number of pieces identified
mrx09	Beatles, Queen, Michael Jackson	297	274
mrx10_1	RWC Popular	100	100
mrx10_2	RWC Popular	100	100
sal	SALAMI	1000	674

Table 2. Summary of the annotations identified in each corpus

On the opposite end, the worst overall performance in the MIREX09 dataset is on songs by Queen and Michael Jackson. At the bottom is Jackson's "They Don't Care About Us". The nine algorithms' output for this song, along with the ground truth and pw_f scores, are shown in Figure 4. This song is highly repetitive, especially harmonically, although the sung portions are very distinct from the instrumental sections: intro (before 0:42), outro (after 3:50), and interlude (2:40 to 3:10). Most of the estimated descriptions discover this overall structure, but fail to differentiate between the similar verses and choruses. Perhaps algorithms will need to employ more than just harmonic features to improve performance on a case like this.

On the other hand, the annotation also labels the end of the song differently from the chorus, even though they sound similar. That is, the annotation conflates musical similarity with musical function, the situation discussed by [13]. To improve performance, MIREX may wish to update its ground truth to reflect primarily musical similarity; or, algorithms should aim to characterize the semantic labels usually ignored in an analysis.

Conspicuously, 17 of the easiest 20 songs (again, with respect to pw_f) are Beatles tunes, while only 2 of the most difficult 20 songs are—the rest being Michael Jackson, Queen and Carole King songs. Taking the median pw_f across the algorithms and comparing this value for the 274 annotations identified as one of these four artists, a Kruskal-Wallis test confirms that the groups differ. A multiple comparison test reveals that pw_f is significantly greater for the Beatles group than the three others. The simplest explanation is that the songs by the other artists are simply more challenging to analyze than the bulk of the Beatles catalogue. However, this may be evidence that the community is overlearning on the Beatles dataset, which has been widely distributed and used as a test collection for at least 6 years.

5. CONCLUSION

We revisited the 2012 MIREX Structure Segmentation task to better understand the performance of the algo-

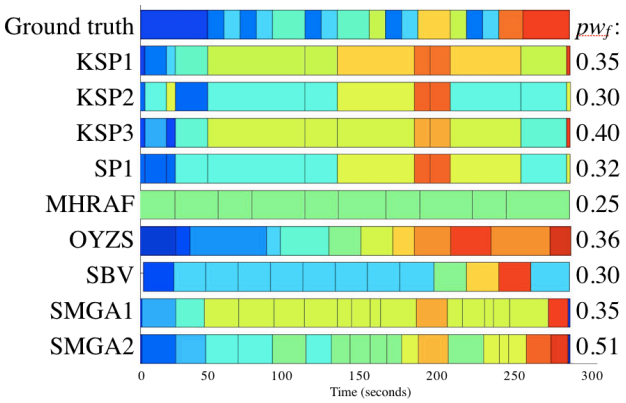


Figure 4. Annotated ground truth and algorithm output for "They Don't Care About Us" by Michael Jackson. The median pw_f achieved by these algorithms was among the lowest in all of MIREX 2012.

rithms and the behaviour of the evaluation metrics. Using a correlation analysis, we identified the same metrics excluded from MIREX as redundant (*K, asp, acp*) and one as unstable and biased (*R*). Thanks to the release of the ground truth and algorithm output with the 2012 MIREX SS task, we were able to investigate the relationship between evaluation metrics and simple properties of the annotated and estimated descriptions, identifying the lack of regularity in the number of segments per song as a hindrance to many submissions.

We hope that this investigation serves as a positive example of the kind of learning that can be accomplished through meta-analysis. Other MIREX tasks could benefit from the release of algorithm output data and information about the ground truth. While the MIR community must weigh the value of open evaluations with the cost of new datasets, note that it is not necessary to release all the ground truth to benefit a meta-analysis: indeed, this analysis focused mainly on non-identifying parameters of the annotations (Section 3.2) and only a few high- and low-performing songs.

6. ACKNOWLEDGEMENTS

This research was supported in part by the Social Sciences and Humanities Research Council of Canada and a QMUL EPSRC Doctoral Training Account studentship.

7. REFERENCES

[1] Abdallah, S., Noland, K., Sandler, M., Casey, M., & Rhodes, C. 2005. Theory and evaluation of a Bayesian music structure extractor. In *Proc. ISMIR*. London, UK. 420–5.

[2] Bimbot, F., Deruty, E., Sargent, G., & Vincent, E. 2012. Semiotic structure labeling of music pieces: Concepts, methods and annotation conventions. In *Proc. ISMIR*. Porto, Portugal. 235–40.

[3] Bimbot, F., Le Blouch, O., Sargent, G., & Vincent, E. 2010. Decomposition into autonomous and comparable blocks: A structural description of music pieces. In *Proc. ISMIR*. 189–94).

[4] Buckley, C. and E. M. Voorhees. 2005. Retrieval system evaluation. In *TREC: Experiment and Evaluation in Information Retrieval*. E. M. Voorhees and D. K. Harman, eds. MIT Press, Cambridge, MA.

[5] Downie, J. S. 2008. The music information retrieval evaluation exchange (2005-2007): A window into music information retrieval research. *Acoustical Science and Technology* 29, 247–55.

[6] Ehmann, A. F., M. Bay, J. S. Downie, I. Fujinaga, and D. De Roure. 2011. Music structure segmentation algorithm evaluation: Expanding on MIREX 2010 analyses and datasets. In *Proc. ISMIR*. Miami, FL. 561–6.

[7] Goto, M. 2006. AIST annotation for the RWC music database. In *Proc. ISMIR*. Victoria, Canada. 359–60.

[8] Kaiser, F., Sikora, T., & Peeters, G. 2012. MIREX 2012 - Music Structural Segmentation Task: IrcamStructure submission. In *Late-breaking and demo session, ISMIR*.

[9] Levy, M. & Sandler, M. 2008. Structural segmentation of musical audio by constrained clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, 16 (2). 318–26.

[10] Lukashevich, H. 2008. Towards quantitative measures of evaluating song segmentation. In *Proc. ISMIR*. Philadelphia, PA. 375–80.

[11] Martin, B., Hanna, P., Robine, M., & Ferraro, P. 2012. Structural analysis of harmonic features using string matching techniques. In *Late-breaking and demo session, ISMIR*.

[12] Peeters, G. 2004. Deriving musical structures from signal analysis for music audio summary generation: "sequence" and "state" approach. In G. Goos, J. Hartmanis, and J. van Leeuwen (Eds.), *Computer Music Modeling and Retrieval*, 2771: 169–85. Springer Berlin / Heidelberg.

[13] Peeters, G. & Deruty, E. 2009. Is music structure annotation multi-dimensional? A proposal for robust local music annotation. In *Proceedings of the International Workshop on Learning the Semantics of Audio Signals*. Graz, Austria. 75–90.

[14] Peeters, G., Urbano, J., & Jones, G. J. F. 2012. Notes from the ISMIR 2012 late-breaking session on evaluation in music information retrieval. In *Late-breaking and demo session*, ISMIR.

[15] Pollack, A. 2001. "Notes on ... Series." http://www.icce.rug.nl/~soundscapes/DATABASES/AWP/awp-notes_on.shtml

[16] Sargent, G., Bimbot, F., & Vincent, E. 2012. A music structure inference algorithm based on morphological analysis. In *Late-breaking and demo session*, ISMIR.

[17] Serrà, J., Müller, M., Grosche, P., & Arcos, J. L. 2012a. The importance of detecting boundaries in music structure annotation. In *Late-breaking and demo session*, ISMIR.

[18] Serrà, J., Müller, M., Grosche, P., & Arcos, J. L. 2012b. Unsupervised detection of music boundaries by time series structure features. In *Proceedings of the AAAI International Conference on Artificial Intelligence*, Toronto, Ontario, Canada. 1613–9.

[19] Smith, J. B. L., Burgoyne, J. A., Fujinaga, I., De Roure, D., & Downie, J. S. 2011. Design and creation of a large-scale database of structural annotations. In *Proc. ISMIR*. Miami, FL. 555–60.

[20] Urbano, J. 2011. Information retrieval meta-evaluation: Challenges and opportunities in the music domain. In *Proc. ISMIR*. Miami, FL. 609–14.

[21] AIST: http://staff.aist.go.jp/m.goto/RWC-MDB/AIST-Annotation/

[22] EP: http://www.ifs.tuwien.ac.at/mir/audiosegmentation.html#anchor_corpus

[23] INRIA: http://musicdata.gforge.inria.fr/structureAnnotation.html

[24] Isophonics: http://www.isophonics.net/content/reference-annotations

[25] SALAMI: http://ddmal.music.mcgill.ca/research/salami/annotations

[26] TUT: http://www.cs.tut.fi/sgn/arg/paulus/structure.html

[27] MIREX data: http://nema.lis.illinois.edu/nema_out/mirex2012/results/struct/

A DETERMINISTIC ANNEALING EM ALGORITHM FOR AUTOMATIC MUSIC TRANSCRIPTION

Tian Cheng, Simon Dixon and Matthias Mauch
Centre for Digital Music, Queen Mary University of London
{tian.cheng, simon.dixon, matthias.mauch} @eecs.qmul.ac.uk

ABSTRACT

In the past decade, non-negative matrix factorisation (NMF) and probabilistic latent component analysis (PLCA) have been used widely in automatic music transcription. Despite their successes, these methods only guarantee that the decomposition converges to a local minimum in the cost function. In order to find better local minima, we propose to extend an existing PLCA-based transcription method with the deterministic annealing EM (DAEM) algorithm. The PLCA update rules are modified by introducing a "temperature" parameter. At higher temperatures, general areas of the search space containing good solutions are found. As the temperature is gradually decreased, distinctions in the data are sharpened, resulting in a more fine-grained optimisation at each successive temperature. This process reduces the dependence on the initialisation, which is otherwise a limitation of NMF and PLCA approaches. The method was tested on two standard multi-instrument transcription data sets (MIREX and Bach10). Experimental results show that the proposed method significantly outperforms a state-of-the-art reference method, according to both frame-based and note-based metrics. An additional analysis of instrument assignment results shows that instrument spectra are typically modelled as mixtures of templates from several instruments.

1. INTRODUCTION

Automatic music transcription is the process of transcribing audio into a symbolic music representation. To date, non-negative matrix factorisation (NMF) [15] and its probabilistic counterpart, probabilistic latent component analysis (PLCA) [17], have been used extensively for this task. These methods treat the spectrogram as a matrix, and decompose it into spectral bases, gain functions, and instrument distributions (when considering different instruments). Although not yet providing the best transcription results, they provide a powerful mathematical model which can lead to a meaningful decomposition, using the constraints of non-negativity and sparsity. Another advantage of these methods is that they are easy to extend, by formulating a more complex model, adding variables or combining them with other models.

One obvious problem of non-negative matrix decomposition methods (such as NMF and PLCA) is that they are initialisation-sensitive and tend to converge to a local minimum. Training instrument templates is an effective way to initialise the spectral bases. By fixing the templates during the updating, we obtain a stable output for the gain function, independent of its initialisation. But when the model becomes more complicated, as by introducing an instrument variable into the model, which is used widely nowadays, it is not possible for us to find good initialisations for all variables.

In this paper, we tackle the local minimum problem by introducing an optimisation method. When using non-negative matrix decomposition methods, the transcription result is related to the cost function, the update rules and also the constraints. Here, we particularly focus on PLCA, which utilises the Kullback-Leibler (KL) divergence as the cost function and derives the update rules based on the EM algorithm [16]. To address the local minimum problem of the EM algorithm, we make use of the deterministic annealing EM algorithm [18] by introducing a temperature parameter into an existing PLCA-based model [2]. The proposed method is tested on the Bach10 dataset [5] and the MIREX multi-F0 development dataset [1]. In comparison to the original PLCA-based model, the proposed method improves the results of multi-F0 estimation and note tracking, while the instrument assignment results vary for each individual instrument.

Although not much attention has been paid to the local minimum problem of automatic music transcription methods, there is still some related work. Bertin et al. [3] used a tempering scheme to favour the convergence of Itakura-Saito (IS) divergence to global minima. Experiments on music transcription show that IS-NMF can provide a good result by choosing a suitable temperature parameter. Hofmann [9] proposed a model based on the tempered EM algorithm to avoid overfitting in probabilistic latent semantic analysis. Kameoka et al. [11] introduced the DAEM algorithm into the harmonic-temporal-structured clustering (HTC) model for audio feature extraction. The HTC model is represented by a Gaussian kernel, and the DAEM algorithm is used to optimise the parameter convergence. Itaya et al. [10] used the DAEM algorithm to estimate the parameters of Gaussian mixture models (GMMs) and hidden

Markov models (HMMs). Experiments on speaker recognition and speech recognition show that DAEM is an effective method for GMM- and HMM-based acoustic modeling. Finally, Smaragdis et al. [16] stated that it is more likely to get "meaningful" decompositions and quick convergence by using "annealing" in PLCA.

The rest of this paper is organised as follows. In Section 2, we describe the PLCA model and the local minimum problem of this model. In Section 3, the update rules of a PLCA-based model are modified according to the DAEM algorithm. The results for three transcription subtasks are presented in Section 4. Finally discussion and conclusions are indicated in Section 5 and 6, respectively.

2. PLCA AND SHIFT-INVARIANT PLCA

Two basic PLCA models, PLCA and Shift-invariant PLCA, are presented in [17]. For automatic music transcription, the spectrogram is formulated by PLCA as:

$$V(\omega,t) \approx P(\omega,t) = P(t)\sum_{p} P(\omega|p)P(p|t) \quad (1)$$

where $V(\omega,t)$ is the input spectrogram, $P(\omega,t)$ the approximated spectrogram, ω is the frequency bin, t the frame number. $P(t)$ is the energy of each time frame, $P(\omega|p)$ is the spectral bases corresponding to pitch p, and $P(p|t)$ the gain function.

To build a shift-invariant PLCA model, the spectrogram needs to be presented on a logarithmic frequency scale, such as the constant-Q transform. Assuming that the energy distributions of adjacent pitches are similar for any given instrument, the spectral basis can be shifted in frequency very easily, as the pattern of partial spacings is the same for all pitches, due to the logarithmic frequency axis. The spectrogram is formulated as:

$$\begin{aligned} V(\omega,t) \approx P(\omega,t) &= \sum_{z} P(z)P(\omega|z) *_{\omega} P(f,t|z) \\ &= \sum_{z} P(z)\sum_{f} P(\omega-f|z)P(f,t|z) \end{aligned} \quad (2)$$

where $P(\omega|z)$ and $P(f,t|z)$ are the spectral templates and time-dependent shifted variant f of component z, and $P(z)$ is the prior distribution of the components.

In many recent systems the PLCA model is extended by introducing an instrument distribution, with templates trained per pitch per instrument. The spectrogram is formulated as:

$$V(\omega,t) \approx P(\omega,t) = P(t)\sum_{p,s} P(\omega|s,p)P(s|p,t)P(p|t) \quad (3)$$

where $P(\omega|s,p)$ represents the spectral templates corresponding to each instrument s and pitch p, $P(s|p,t)$ the instrument contribution to each pitch in the t^{th} frame, and $P(p|t)$ the pitch probability distribution for each frame.

The parameters of the PLCA models are estimated by iteratively decreasing the KL divergence of the input spectrogram $V(\omega,t)$ and the synthetic spectrogram $P(\omega,t)$ using the EM algorithm. The KL divergence is convex in one variable, but not convex in multiple variables [12]. In this case, the EM algorithm can only guarantee to find a local minimum for these parameters, so the results depend on the initialisation. The use of instrument templates is an effective way to deal with the initialisation sensitivity of the algorithm. Taking the model described in Eqn. (1) for example, if the templates are fixed as a constant, the gain function will be convex. This means that when we formulate the model as the product of the spectral bases and a gain function, we obtain a unique gain function corresponding to a fixed set of templates. On the one hand, the templates lead to a stable decomposition for automatic music transcription; on the other hand, the templates also limit the performance of the transcription. However, when encountering the extended model as described in Eqn. (3), the instrument contribution and the pitch contribution still face the risk of converging to local minima, even with fixed templates.

3. PROPOSED METHOD

To deal with the local minimum problem of PLCA models, we derive the update rules according to the deterministic annealing EM algorithm [18], which introduces a temperature parameter into the EM algorithm. The temperature parameter is employed on the posterior probability density in the E-step. Then by gradually reducing the temperature, the EM steps are iteratively executed until convergence at each temperature, leading the result to a global or better local minimum. We apply this method to a baseline PLCA-based model proposed in [2]. Since the templates are kept fixed, the temperature parameter is applied to the posterior probability density of the instrument distribution. In this way, we can enjoy the benefits of the DAEM algorithm and the templates.

3.1 The Baseline PLCA Model

Benetos and Dixon [2] proposed a model that adds an instrument distribution variable to shift-invariant PLCA. The time-frequency representation of the input signal was computed with the Constant-Q Transform [14] using 120 bins per octave. Templates were trained for 10 instruments allowing shifts within a semitone range, in order to deal with arbitrary tuning and frequency modulation. The model is formulated as:

$$P(\omega,t) = P(t)\sum_{p,s} P(\omega|s,p) *_{\omega} P(f|p,t)P(s|p,t)P(p|t) \quad (4)$$

where $P(\omega,t)$ is the approximated spectrogram, $P(t)$ is the energy distribution of spectrogram. $P(\omega|s,p)$ are the templates of instrument s and pitch p, $P(f|p,t)$ is the shifted variant for each p, $P(s|p,t)$ is the instrument contribution for each pitch, and $P(p|t)$ is the pitch probability distribution for each time frame. The templates $P(\omega|s,p)$ are trained using the MAPS dataset [6] and RWC dataset [7].

The update rules are derived from the EM algorithm.

	instrument	lowest note	highest note
1	Bassoon	34	72
2	Cello	26	81
3	Clarinet	50	89
4	Flute	60	96
5	Guitar	40	76
6	Horn	41	77
7	Oboe	58	91
8	Piano	21	108
9	Tenor Sax	44	75
10	Violin	55	100

Table 1: Instrument ranges, adapted from [1]

For the E-step, the posterior probability density is:

$$P(p,f,s|\omega,t) = \frac{P(\omega-f|s,p)P(f|p,t)P(s|p,t)P(p|t)}{\sum_{p,f,s}P(\omega-f|s,p)P(f|p,t)P(s|p,t)P(p|t)} \quad (5)$$

For the M-step, each parameter is estimated.

$$P(f|p,t) = \frac{\sum_{\omega,s}P(p,f,s|\omega,t)P(\omega,t)}{\sum_{f,\omega,s}P(p,f,s|\omega,t)P(\omega,t)} \quad (6)$$

$$P(s|p,t) = \frac{(\sum_{\omega,f}P(p,f,s|\omega,t)P(\omega,t))^{\alpha_1}}{\sum_s(\sum_{\omega,f}P(p,f,s|\omega,t)P(\omega,t))^{\alpha_1}} \quad (7)$$

$$P(p|t) = \frac{(\sum_{\omega,f,s}P(p,f,s|\omega,t)P(\omega,t))^{\alpha_2}}{\sum_p(\sum_{\omega,f,s}P(p,f,s|\omega,t)P(\omega,t))^{\alpha_2}} \quad (8)$$

The templates $P(\omega|s,p)$ are not updated as they are previously trained and kept fixed. The parameters α_1 and α_2 used in Eqn. (7) and (8) are used to enforce sparsity, where $\alpha_1, \alpha_2 > 1$. We set $\alpha_1 = 1.3$ and $\alpha_2 = 1.1$. The final piano-roll matrix $P(p,t)$ and the pitches assigned to each instrument $P(p,t,s)$ are given by:

$$P(p,t) = P(p|t)P(t) \quad (9)$$

$$P(p,t,s) = P(s|p,t)P(p|t)P(t) \quad (10)$$

For post-processing, instead of using an HMM, the note events are extracted by performing thresholding on $P(p,t)$ and using minimum-length pruning (deleting notes shorter than $50ms$). The instrument-wise note events are detected in the same way using $P(p,t,s)$.

3.2 The DAEM-based Model

To modify the update rules according to the DAEM algorithm, in the E-step, the posterior probability density in Eqn. (5) is modified by introducing a temperature parameter τ [1]:

$$P_\tau(p,f,s|\omega,t) = \frac{(P(\omega-f|s,p)P(f|p,t)P(s|p,t)P(p|t))^{1/\tau}}{\sum_{p,f,s}(P(\omega-f|s,p)P(f|p,t)P(s|p,t)P(p|t))^{1/\tau}} \quad (11)$$

And the update rules are extended by adding a τ-loop:

- Set $\tau \leftarrow \tau_{max}(\tau_{max} > 1)$.
- Iterate the following EM-steps until convergence:
  E-step: calculate $P_\tau(p,f,s|\omega,t)$.
  M-step: estimated $P(f|p,t)$, $P(s|p,t)$ and $P(p|t)$ by replacing $P(p,f,s|\omega,t)$ with $P_\tau(p,f,s|\omega,t)$.
- Decrease τ.
- If $\tau \geq 1$, repeat from step 2; otherwise stop.

By gradually decreasing τ, the temperature is cooling down. At higher temperatures, the distributions are smoothed and general areas of the search space containing good solutions are found. As the temperature is gradually decreased, distinctions in the data are sharpened, resulting in a more fine-grained optimisation at each successive temperature.

Considering the properties of this particular model, we simplify the posterior probability density to:

$$P_\tau(p,f,s|\omega,t) = \frac{P(\omega-f|s,p)P(f|p,t)P(s|p,t)^{1/\tau}P(p|t)}{\sum_{p,f,s}P(\omega-f|s,p)P(f|p,t)P(s|p,t)^{1/\tau}P(p|t)} \quad (12)$$

The convolution of the templates and the pitch impulse distribution, giving the terms $P(\omega-f|s,p)P(f|p,t)$, works as the shift-invariant templates here. These are not modified by the temperature parameter, as the templates are fixed during the iterative process [2]. In addition, having observed that the pitch distribution $P(p|t)$ is dependent on the instrument distribution $P(s|p,t)$ in this model, we only need to modify $P(s|p,t)$ in the posterior probability density.

In the experiment, the parameter τ took the values $10/i,\ i \in \{8,9,10\}$. When τ finally decreases to 1, the update rules agree with the original ones.

4. EVALUATION

4.1 Datasets

We used the Bach10 Dataset [5] and the MIREX Multi-F0 Development Dataset (MIREX dataset) [1] to test the performance of the proposed method. The Bach10 dataset consists of 10 quartet recordings performed on violin, clarinet, saxophone and bassoon. The MIREX dataset is an excerpt from a woodwind quintet recording, played on bassoon, clarinet, flute, horn, oboe.

4.2 Evaluation Metrics

The performance of the proposed system is evaluated on three subtasks of automatic music transcription. The first two, multiple F0 estimation and note tracking, are very commonly used. The third subtask, instrument assignment, evaluates the algorithms' ability to assign the notes to corresponding instruments.

[1] The parameter used in [18] is β, and the temperature is indicated by $1/\beta$. The reason of using τ here is because we want to indicate the temperature directly by τ and distinguish the proposed method from the β-divergence.

[2] This was also confirmed by test experiments where the power $1/\tau$ was also applied to the pitch impulse distribution $P(f|p,t)$, giving similar transcription results.

Dataset	Methods	P	R	F	Acc	$\boldsymbol{E}_{tot}$	E_{subs}	E_{miss}	E_{fa}
Bach10	BD(2012)	0.784	0.791	0.787	0.650	0.311	0.116	0.093	0.102
	Proposed	0.819	0.796	0.807	**0.677**	**0.282**	0.098	0.106	0.078
MIREX	BD(2012)	0.748	0.537	0.625	0.455	0.486	0.158	0.305	0.023
	Proposed	0.769	0.561	0.649	**0.480**	**0.461**	0.146	0.292	0.023
Both	BD(2012)	0.781	0.768	0.772	0.632	0.327	0.120	0.112	0.094
	Proposed	0.814	0.775	0.793	**0.659**	**0.299**	0.102	0.123	0.074

Table 2: Multiple F0 estimation results (see text for explanation of symbols).

In the multiple F0 estimation subtask, performance is evaluated frame by frame with an interval of $10ms$. The accuracy metrics are precision (P), recall (R), F-measure (F) [19] and the overall accuracy (Acc) [4], defined as follows:

$$P = \frac{N_{tp}}{N_{sys}},\ R = \frac{N_{tp}}{N_{ref}},\ F = \frac{2 \cdot R \cdot P}{R + P} \tag{13}$$

$$Acc = \frac{N_{tp}}{N_{tp} + N_{fp} + N_{fn}} \tag{14}$$

where N_{tp} is the number of true positives, N_{sys} and N_{ref} denote the number of the detected pitches and the ground-truth pitches, N_{fp} and N_{fn} are the number of false positives and false negatives respectively. The error metrics are the rates of total error (E_{tot}), substitution error (E_{subs}), missed detections (E_{miss}) and false alarms (E_{fa}). See the definitions in [13].

For the note tracking task, a note is considered correctly detected if the note is within the following ranges of ground truth.

pitch range $\pm 3\%$
onset range $\pm 50ms$
offset range $\pm$ max $\{20\%$ of the duration, $50ms\}$

The algorithms are evaluated in terms of onset-only and onset-offset accuracies, which are denoted by P_{on}, R_{on}, F_{on}, Acc_{on} and P_{off}, R_{off}, F_{off}, Acc_{off} respectively.

The instrument assignment task assesses whether the transcription not only identifies the correct pitch, but also the correct instrument. First, pitches are detected for each individual instrument. Then instruments actually occurring in the piece are evaluated according to the frame-based F-measure (13), whereas for the other instruments we calculate the false positive rate.

4.3 Results

We compare the performance of the proposed method to that of the baseline PLCA model introduced in Section 3.1 (mentioned as BD(2012) below). Here, we provide results for three subtasks on two different datasets.

4.3.1 Multiple F0 Estimation

The results for multiple F0 estimation using the Bach10 and MIREX datasets for two methods are shown in Table 2. It can be seen that the proposed method outperforms the BD(2012) method in terms of accuracy (Acc) on both individual datasets by at least 2.5 percentage points, leading to an increased overall accuracy of 0.659 (up 2.7 percentage points). The total error decreases by 2.8 percentage points. On the Bach10 dataset improvements are mainly due to a reduced false alarm rate (E_{fa}), which decreases from 10.2% to 7.8%. This is also reflected by increased precision (P) and stable recall (R). The improvement for the MIREX dataset mainly comes from reduction in both substitution error (E_{subs}) and missed detection error (E_{miss}) rates, leading to higher precision and recall.

In order to determine if the increase in accuracy (Acc) is significant we ran a Friedman test for this subtask. The resulting p-value of $0.0009 < 0.01$ indicates that the difference is highly significant. The distribution of Acc of the ten files in the Bach10 dataset is shown in Figure 1a.

Dataset	Methods	P_{on}	R_{on}	$\boldsymbol{F}_{on}$	Acc_{on}
Bach10	BD(2012)	0.319	0.339	0.328	0.197
	Proposed	0.399	0.354	**0.374**	0.231
MIREX	BD(2012)	0.628	0.420	0.503	0.336
	Proposed	0.690	0.459	**0.551**	0.380
Both	BD(2012)	0.347	0.346	0.344	0.209
	Proposed	0.427	0.364	**0.391**	0.245

(a) onset-only accuracy

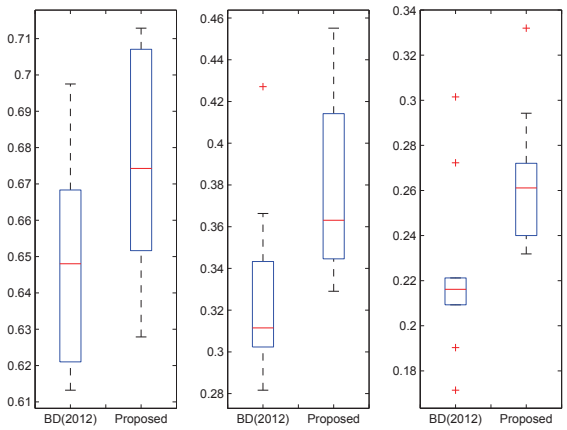
Dataset	Methods	P_{off}	R_{off}	$\boldsymbol{F}_{off}$	Acc_{off}
Bach10	BD(2012)	0.217	0.230	0.223	0.126
	Proposed	0.281	0.249	**0.263**	0.152
MIREX	BD(2012)	0.487	0.326	0.391	0.243
	Proposed	0.537	0.357	**0.429**	0.273
Both	BD(2012)	0.242	0.239	0.238	0.137
	Proposed	0.305	0.259	**0.279**	0.163

(b) onset and offset

Table 3: Note-tracking results

4.3.2 Note Tracking

For the note tracking subtask, we found that the F-measure was improved by almost 5 percentage points for onset-only evaluation and around 4 percentage points for onset-offset evaluation for both datasets, as shown in Table 3. We ran a Friedman test with regard to the F-measures (F_{on} and F_{off}) for this subtask. For both onset-only and onset-offset metrics, the p-values are less than 0.01, showing that—here,



(a) *Acc* (b) F_{on} (c) F_{off}

Figure 1: Box-and-whisker plots of (a) accuracy; (b) onset-only F-measure; and (c) onset-offset F-measure; for the Bach10 dataset.

too—the differences are significant. The distributions of F_{on} and F_{off} for the Bach10 dataset are shown in Figures 1b and 1c.

The note tracking evaluation shows that both methods under consideration perform better on the MIREX dataset, whereas according to the frame-based evaluation (see Section 4.3.1) they perform better on the Bach10 dataset. This result is in line with the results from other methods on the same data, [3] and is likely to stem from the unusual co-occurrence of trills and legato notes that dominates the MIREX piece.

4.3.3 Instrument Assignment

The results for instrument assignment for the two datasets are shown in Table 4. In this subtask, we cannot identify a systematic advantage of either method, with the F-measure means over all instruments being very close (20.7% and 20.9% on the Bach10 dataset, and 35.1% and 34.3% on the MIREX dataset). Slight differences between the methods for particular instruments do not show a consistent advantage of one method either; we will therefore focus on the proposed method in the rest of the discussion. The most obvious differences in F-measure occur between instruments. For example, the results for the Bach10 dataset show that instrument assignment works better for the clarinet and bassoon than for the violin and saxophone. Also, since the note templates include instruments not present in the pieces, false positives occur for these instruments, with the largest ratio of false positives occurring for horn (18.6%) and piano (16.4%). The problem instrument in the MIREX dataset is the oboe, to which few notes are assigned, leading to a low F-measure of around 12-13%. Notes are detected in three instruments that do not feature in the music, with the largest ratio of false positives found in the piano (47.9%) and guitar (34.5%). No false positives were detected for saxophone or violin.

The discrepancy between the satisfactory multiple F0 estimation results and the comparatively low results for instrument assignment is due to the fact that often the correct pitch is detected, but assigned to a wrong instrument or combination of instruments. That is, note templates from different instruments are combined to approximate the observed spectra. The proposed method provides a better reconstruction of the observed data using combinations of templates at the correct pitches, resulting in better performance for frame level and note tracking tasks.

[3] as published on the MIREX website [1].

F-measure	Violin	Clarinet	Saxophone	Bassoon	Mean
BD(2012)	0.175	0.313	0.092	0.246	0.207
Proposed	0.190	0.275	0.127	0.243	0.209

(a) Bach10

F-measure	Bassoon	Clarinet	Flute	Horn	Oboe	Mean
BD(2012)	0.292	0.444	0.485	0.409	0.125	0.351
Proposed	0.294	0.420	0.489	0.385	0.129	0.343

(b) MIREX

Table 4: Instrument assignment results

5. DISCUSSION

The use of the temperature parameter τ that is central to the DAEM algorithm in Eqn. (11) is similar to the use of the sparsity parameters in Eqn. (7) and Eqn. (8). In fact, the sparsity method used here is related to the Tempered EM algorithm [8]. Both the DAEM and sparsity equations 'put an exponent on a distribution'. When the exponent is larger than one, the distribution becomes sharper and sparser; when the exponent is smaller than one, the distribution is smoothed, as in the case of high-temperature stages of DAEM.

So far we have used DAEM with only one configuration of three temperature steps. In the future, we would like to explore different configurations to see whether we can further improve the results of multiple F0 estimation and note tracking.

We have shown that DAEM can improve the performance of an EM-based model, but further investigations are needed to show how well this result generalises. For example, preliminary tests have shown that applying DAEM directly in the standard PLCA model in Eqn. (1) without templates, fails to provide better results.

We observe that the previously-trained templates are very important and work as an excellent initialisation for the spectral bases in the PLCA models. On the other hand, they also influence the result of the gain functions, which means that the transcription result will be poor if we use poor or inappropriate templates. The risk of updating the templates during the iteration is that an updated template might no longer accord with its labels (pitch, instrument). Due to the different ways a note can be played and differences in sound transmission, templates will never match observations precisely. Spectral decomposition algorithms compensate for this mismatch by finding mixtures of templates which provide a better approximation of the data

(see Section 4.3.3). In order to capture the variations of instrument sounds in a single model, we intend to explore physical modelling for time-varying templates in future work.

6. CONCLUSIONS

In this paper, we modified a baseline PLCA model for automatic music transcription. The model's update rules were changed according to the DAEM algorithm to tackle the local minimum problem. The DAEM algorithm introduces a temperature parameter to the update rules and leads the decomposition to converge to a global or better local minimum by gradually lowering the temperature. The proposed method was tested using two standard transcription datasets, the Bach10 dataset and the MIREX dataset. The results show that the proposed method significantly outperforms the baseline method in multiple F0 estimation (accuracy increases by 2.7 percentage points) and note tracking (F-measure increases by 4 percentage points). Although results on an additional instrument assignment task show no significant difference between the methods, they reveal that both methods use mixtures of instrument templates to approximate observed spectra in the test data. We noted several aspects that call for further study: DAEM temperature configurations, the extension of DAEM to more general PLCA models, and the use of physical modelling to generate more flexible instrument templates.

7. ACKNOWLEDGEMENTS

Tian Cheng is supported by China Scholarship Council (CSC)/Queen Mary Joint PhD scholarships. We would like to thank Emmanouil Benetos and Roland Badeau for their comments on this paper.

8. REFERENCES

[1] Music Information Retrieval Evaluation eXchange (MIREX). `http://www.music-ir.org/mirex/wiki/MIREX_HOME`.

[2] E. Benetos and S. Dixon. A shift-invariant latent variable model for automatic music transcription. *Computer Music Journal*, 36(4):81–94, 2012.

[3] N. Bertin, C. Févotte, and R. Badeau. A tempering approach for itakura-saito non-negative matrix factorization. with application to music transcription. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP09)*, pages 1545–1548, Apr. 2009.

[4] S. Dixon. On the computer recognition of solo piano music. In *Proceedings of Australasian Computer Music Conference*, pages 31–37, 2000.

[5] 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, 2010.

[6] 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, 2010.

[7] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC Music Database: Music Genre Database and Musical Instrument Sound Database. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR03)*, pages 229–230, 2003.

[8] G. Grindlay and D. Ellis. A probabilistic subspace model for multi-instrument polyphonic transcription. In *Proceedings of the 11th International Society for Music Information Retrieval Conference*, Utrecht, Netherlands, August 9-13, 2010.

[9] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence, UAI99*, pages 289–296, 1999.

[10] Y. Itaya, H. Zen, Y. Nankaku, C. Miyajima, K. Tokuda, and T. Kitamura. Deterministic annealing EM algorithm in acoustic modeling for speaker and speech recognition. *IEICE Transactions*, 88-D(3):425–431, 2005.

[11] H. Kameoka, T. Nishimoto, and S. Sagayama. Harmonic-temporal structured clustering via deterministic annealing EM algorithm for audio feature extraction. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR05)*, pages 115–122, Sep. 2005.

[12] D. Lee and H. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562. MIT Press, 2001.

[13] G. Poliner and D. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, pages 154–162, 2007.

[14] C. Schoerkhuber and A. Klapuri. Constant-q transform toolbox for music processing. In *the 7th Sound and Music Computing Conference*, 2010.

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

[16] P. Smaragdis and B. Raj. Shift-invariant probabilistic latent component analysis. Technical report, 2007.

[17] P. Smaragdis, B. Raj, and M. Shashanka. Sparse and shift-invariant feature extraction from non-negative data. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP08)*, pages 2069–2072, Apr. 2008.

[18] N. Ueda and R. Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11(2):271 – 282, 1998.

[19] 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, 2010.

MODELLING THE SPEED OF MUSIC USING FEATURES FROM HARMONIC/PERCUSSIVE SEPARATED AUDIO

Anders Elowsson **Anders Friberg**
KTH Royal Institute of Technology,
CSC, Dept. of Speech, Music and Hearing
elov@kth.se afriberg@kth.se

Guy Madison **Johan Paulin**
Department of Psychology, Umeå University
{Guy.Madison,Johan.Paulin}@psy.umu.se

ABSTRACT

One of the major parameters in music is the overall speed of a musical performance. In this study, a computational model of speed in music audio has been developed using a custom set of rhythmic features. Speed is often associated with tempo, but as shown in this study, factors such as note density (onsets per second) and spectral flux are important as well. The original audio was first separated into a harmonic part and a percussive part and the features were extracted separately from the different layers. In previous studies, listeners had rated the speed of 136 songs, and the ratings were used in a regression to evaluate the validity of the model as well as to find appropriate features. The final models, consisting of 5 or 8 features, were able to explain about 90% of the variation in the training set, with little or no degradation for the test set.

1. INTRODUCTION

This study is focused on one of the major parameters in music, the overall speed of a musical performance. From a music theoretic background we are used to associate speed with the tempo of the music. However, as suggested earlier, the perceived speed is related to the tempo but may also depend on other aspects like the note density (number of onsets per second) [9]. An indirect indication of this was provided in [2] where it was found that the note density (and not the tempo) was constant for a certain emotional expression across different music examples. Madison & Paulin [12] asked listeners to rate the speed for 50 music examples spanning a variety of musical styles and rhythms. They found that speed correlated with tempo but that there must also be other aspects involved in the perceptual judgment of speed. In earlier works [11, 15, 16] it has been shown that a classification of songs as fast or slow has helped to improve the accuracy of tempo estimation algorithms.

The current work is part of an ongoing study about perceptually determined features in music information retrieval. In a previous study it was shown that speed could be modeled by a combination of tempo and different note densities of the instruments using symbolic data [7]. The explained variation was about 90 % using linear regression. This indicates that a similar result could in theory be obtained using audio data provided that the appropriate low-level audio features could be extracted. Unfortunately, audio features extracted with the MIRToolbox [14] as well as the VAMP plugins available in the Sonic Annotator[1] did not map well to the perception of speed, highlighting the need for new features to be developed [7].

The purpose of the current study was to develop a computational model of speed in music audio restricted to examples containing percussive elements (e.g. drums). A set of rhythmic features were computed, mainly from detected onsets of the music. An important idea was that a relevant model should exploit the characteristics of each onset to better understand the music. As indicated in [7], good results can be achieved by tracking both percussive and harmonic onsets. Therefore, these parts were analyzed separately in the current model. As a first step, source separation was used to separate harmonic content and percussive content in the audio. Onsets and features were computed from both the percussive and the harmonic part as well as from the original audio. A flowchart of the processes used is shown in Figure 1.

To find appropriate features as well to evaluate the validity of the model, regression was used, in which the audio features were mapped against ground truth data consisting of listener ratings of speed.

2. SOURCE SEPARATION AND ONSET DETECTION

2.1 HP-Separation

Source separation has been used in the past in computational models related to rhythm [1]. For this study, the source separation method proposed by FitzGerald [6] was used to separate harmonic and percussive content. The basic idea of the method is that percussive sounds are broadband noise signals with short duration and that harmonic sounds are narrow band signals with longer duration. The audio is first transformed to the spectral domain by using a short-time Fourier transform (STFT). By applying a median filter across each frame in the frequency direction, harmonic sounds are suppressed. By applying a median filter across each frequency bin in the time direction percussive sounds are suppressed. After median filtering, the signal is transformed back to the time domain again using the inverse STFT.

With the STFT it is possible to accurately detect percussive content in the music. The frequency resolution in

[1] http://www.omras2.org/SonicAnnotator

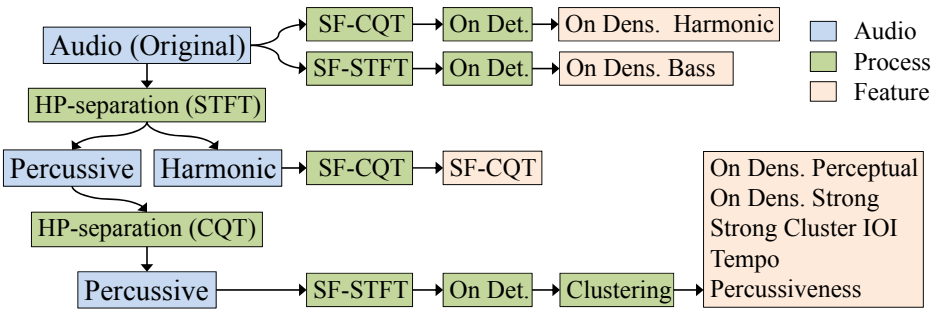


Figure 1. Flowchart of the processes used to compute audio features for the speed in music. The audio is filtered to separate harmonic and percussive content, onsets are detected from a spectral flux, and audio features are computed.

the lower frequencies is however not sufficient to detect harmonic content there. Thus, to further suppress harmonic content in the percussive waveform a second separation stage incorporates a constant-Q transform (CQT) [17]. The CQT can be understood as an STFT with logarithmically spaced frequency bins, accomplished by varying the length of the analysis window. With the CQT, an appropriate frequency resolution can be achieved at all frequencies, at the expense of a poor time resolution in the low frequencies. The frequency resolution of the CQT was set to 60 bins per octave and each frame was median filtered across the frequency direction. After filtering, the percussive signal was transformed back to the time domain using an inverse CQT. Notice that the phase information is retained in the transformation back to the time domain. It can be regarded as a mapping that connects a frequency bin to a certain point in time. The percussive and harmonic waveforms are shown in Figure 2.

2.2 Onset Detection

Audio features were computed from all three waveforms (original, harmonic and percussive) by the scheme shown in Figure 1. The first step, independent of feature and waveform, was to compute a spectral flux (SF) [3], where spectral fluctuations along the time-domain are detected. The SF was computed several times in numerous different ways. Some shared steps will be described here, with unique steps described in Sections 3.1-3.8. The power spectrum was computed with a CQT or a STFT and converted to sound level. A range of 30 dB was used. Thus, the maximum sound level of each band was set to 0 dB and sound levels below -30 dB were set to -30 dB. Let *L(n, i)* represent the sound level at the *i*th frequency bin/band of the *n*th frame. The SF is given by

$$SF(n) = \sum_{i=1}^{b} H\left(L(n,i) - (L(n-s,i)\right) \qquad (1)$$

where *b* is the number of bins/bands. The variable *s* is the step size and *H* is a half-wave rectifier function, or for the percussive SF:

$$H(x) = \begin{cases} x & \text{if } x > 0 \\ 0.2x & \text{if } x \le 0 \end{cases} \qquad (2)$$

The implication of Eq. 2 is that negative spectral fluctuations have a slight influence on the onset detection function. Onsets were detected by peak picking on a low-pass filtered curve of the spectral flux (see Figure 2).

2.3 Clustering

Onsets were clustered based on sound level in 8 frequency bands, spaced approximately an octave apart. An additional band was based on the RMS sound level. As the appropriate number of clusters was unknown beforehand, three K-means clusterings were carried out, with the number of clusters *k*, set to 2, 3 and 4. The fit of each clustering attempt was defined by the smallest Euclidian distance between any two clusters, where a large smallest distance gave a higher fit. When choosing *k*, a higher number of clusters were premiered over a lower if their fit was similar. The result of the clustering is a separation of onsets into different groups as shown in Figure 2.

3. FEATURE EXTRACTION

A total of 8 audio features were computed, 2 from the original waveform, 5 from the percussive waveform and 1 from the harmonic waveform. These features are shown in the flowchart in Figure 1 and described in Sections 3.1-3.8, with one subsection for each feature. An in-depth visualization of the processes involved to compute the features is shown in Figure 2. For conversion to *onset density*, the length of each song was set as the distance between the first and last onset.

3.1 Onset Density – Harmonic

Onsets were tracked from the original waveform, using the SF of a CQT. To avoid false onset detections at pitch glides, deviations in a peak by 20 cents (one bin), without an increase in sound level, were restricted from affecting the SF. This was accomplished by subtracting the sound level of each bin of the new frame, by the maximum sound level of the adjacent bins in the old frame.

3.2 Onset Density – Bass

To comply with the bass feature in [7], onsets in the low register (40 Hz - 210 Hz) were tracked using the SF of the lower bins of a STFT. The frequency bins were summed to a single band before the SF.

3.3 Onset Density – Perceptual weighting

Percussive onsets were tracked using the SF of a STFT on the percussive waveform. The bins of the frequency

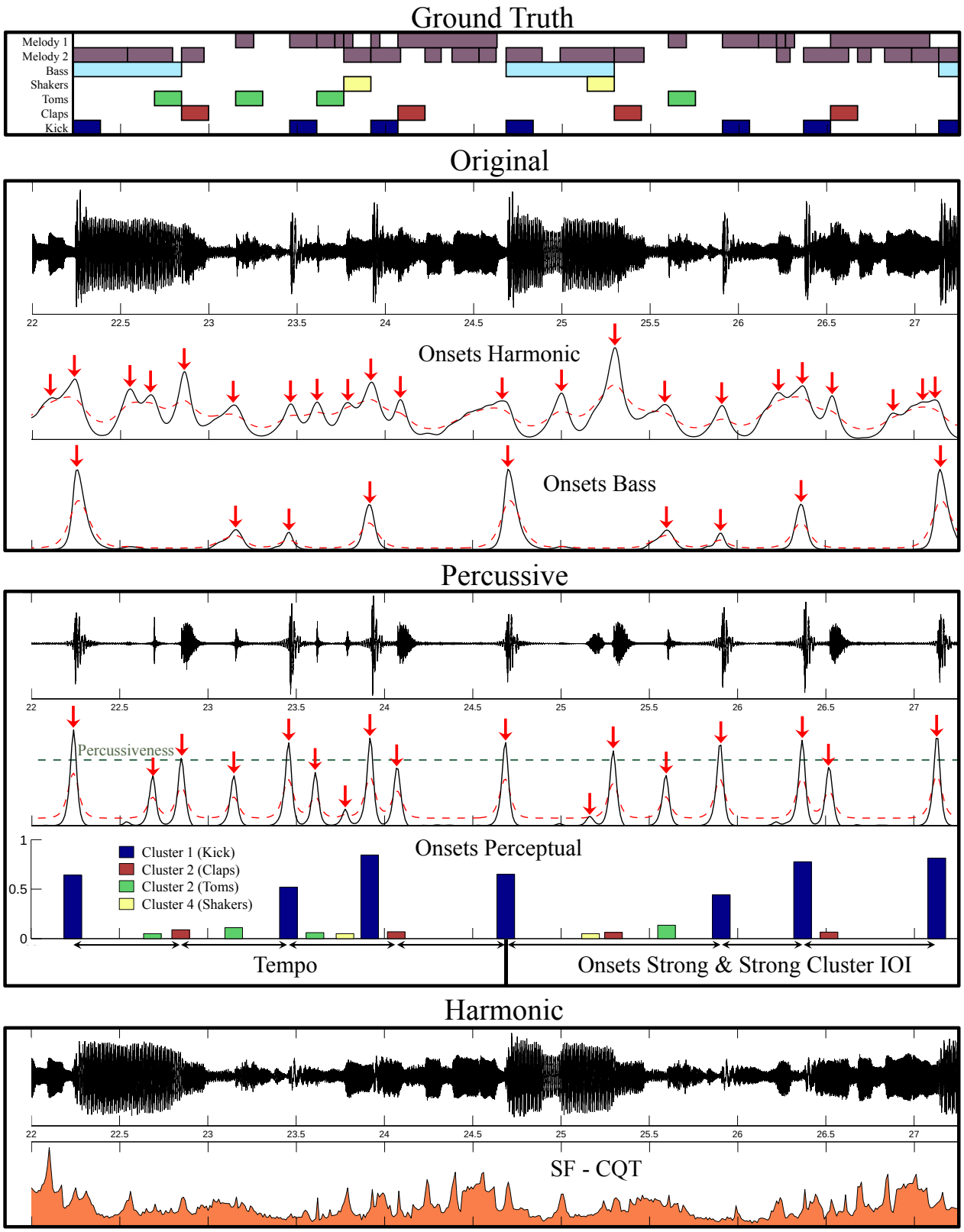


Figure 2. The process of estimating the perceived speed of a piece of music. The example is a 5-second section of the song *Candy Shop*, by *50 cent*. In the top pane we see the ground truth of the audio file. The melodic lines have been consolidated into a single row to convey only onset times. In the next pane we see the processes involved in extracting audio features from the original waveform. In the third pane the percussive audio is used. Notice that the clustering of the audio matches the ground truth. *Tempo* is detected as the IOI between kick and handclaps. Finally the integral of the spectral flux is used from the harmonic waveform in the fourth pane.

domain representation were divided into 13 non-overlapping frequency bands (half-octave spacing). Sub-band processing for onset detection has been described in [13], and can be motivated by its similarity to human hearing [4]. The strength of each detected onset was calculated based on the average sound level of the first

50 ms from the onset position, where lower frequencies were given a higher impact.

To further determine the perceived strength of the onsets, each onset was compared to the surrounding onsets within 1.5 seconds. This time span was defined as the perceptual present of the particular onset. By comparing it with the strongest onset within the perceptual present its strength could be altered to represent its perceptual impact. The onset was given a higher strength if there were no significantly stronger onsets within the perceptual present. If there were onsets that were significantly stronger, its strength was lowered. The height of the cluster-bars in Figure 2 represents the perceptual strength. To derive at a measure of onsets density, the sum of the perceptual strength of the onsets was used.

3.4 Onsets Density – Strong

The strongest clusters of the clustering contributed to two features. The first feature was simply the number of onsets, belonging to a strong cluster, per second. The idea behind this feature is that prominent percussive elements such as the kick drum and the snare drum likely influence the perception of speed in a different way than the less prominent elements such as the hi-hat.

3.5 Strong Cluster IOI

The second feature derived from the strong clusters was developed to catch the assumed perception of a *slow* speed, when the interonset intervals (IOIs) of onsets belonging to the same strong cluster are long. As an example, a song with equally spaced drum onsets consisting of "Kick, Snare, Kick, Snare, etc.." was assumed to have a *higher* perceived speed than a song where the drums instead plays "Kick, Kick, Snare, Kick, etc..". This is accounted for in the *Tempo* feature as well, because the tempo in the second example would be half the tempo of the first example. Cluster IOIs shorter than 750 ms were discarded based on the idea that they can both represent a drum fill in a slow song or represent a regular part of the drum pattern in a fast song.

3.6 Tempo

The tempo detection algorithm is part of an ongoing project, and a detailed description is in preparation. All distances between onsets within 5 seconds from each other are used to detect the tempo. The histogram in Figure 3 is based on the song presented in Figure 2.

First, the period length of the percussive waveform is detected. A histogram of onset distances is generated, where the contribution of each onset-pair is increased with increasing *similarity* in spectrum as well as increasing onset strength. The leftmost peak in the low pass filtered histogram, within 92 % of the highest peak, is chosen as the period length.

Secondly, the tempo (beat length) is detected. A histogram over onset distances is once again generated, where the contribution of each onset-pair is increased with increasing *dissimilarity* in spectrum as well as increasing onset strength. The final probability distribution for tempo (Figure 3) is the Hadamard product of the histogram and several filters. One filter is based on the determined period length. The idea is that the beat will be a simple ratio of the period length, so Hanning windows are produced at the positions given by

$$P_{len} \times \left(\frac{1}{2}\right)^n, \quad P_{len} \times \left(\frac{1}{2}\right)^n \times \left(\frac{1}{3}\right) \qquad n = 0,1,2,.. \tag{3}$$

Another filter is based on IOIs within strong clusters as described in Section 3.5 and several filters are based on onset density. The highest peak in the final probability distribution is chosen as the tempo. In compliance with the findings that speed is a shallower function of tempo for fast and slow music [12], differences in tempo between 60 and 160 BPM are given the highest impact.

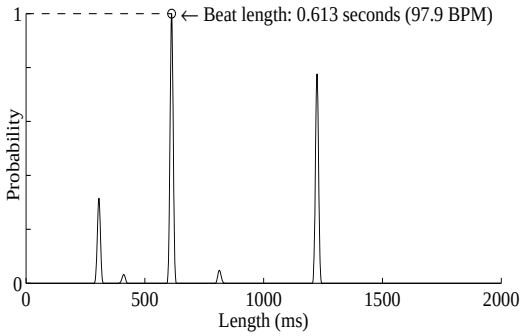


Figure 3. The histogram used to determine tempo.

3.7 Percussiveness

An estimate of how percussive the music is was computed as well. This estimate is derived from the height *h* of the peaks in the SF of the percussive waveform, as shown in Figure 2. Equation 4 gives the mean peak height when *p* is 0, an estimate closer to the lowest peaks when *p* is negative, and an estimate closer to the highest peaks when *p* is positive. In this study *p* was set to 0.4.

$$Percussiveness = \frac{\sum_{i=1}^{n} h(\mathrm{i})^{1+\mathrm{p}}}{\sum_{i=1}^{n} h(\mathrm{i})^{\mathrm{p}}} \tag{4}$$

3.8 SF CQT

When extracting information from the harmonic waveform the integral of the SF was used; indicated as the area in the bottom pane of Figure 2. Onset detection was avoided as the HP-separation had removed all transients from the harmonic waveform.

4. PREDICTING SPEED FROM THE FEATURES

4.1 Speed Data and Audio Examples

The music examples were taken from two earlier studies. To ensure that the songs contained percussive elements, songs where the RMS of the percussive waveform was less than 1/8 of the RMS of the harmonic waveform were not included in the data sets. The training set was 89 popular songs, originally in MIDI format and converted

to audio in a previous experiment [7, 10]. The speed estimations were previously determined using 20 listeners who rated speed for each example on a quasi-continuous scale marked slow-fast (range 1-9). The test set consisted of 47 real audio examples previously used for studying the relation between tempo and speed [12]. They were selected for exhibiting a large variation of tempi and genres within popular music styles. The speed was previously estimated in a similar way to the training set using continuous scales (range 0-10). Due to a difference in the design of the original experiment [12], the medium tempo examples were rated by 60 listeners while the fast and slow examples were rated by 12 listeners.

4.2 Modelling Speed of the Training Set

Two regression techniques were used to analyze the mapping between the computed audio features and the listener ratings. First, a multiple linear regression (MLR) was used, justified by a predictor-to-case ratio higher than 1:10. Secondly, partial least square regression (PLS) was used. PLS regression carries out data reduction, whilst maximizing covariance between features and predicted data [5]. It constructs new predictor variables (components), as linear combinations of the features.

The MLR prediction of listener ratings from computed audio features is presented in Table 1. As shown, a linear combination of the computed audio features was able to explain more than 90 % of the variability. In comparison, the agreement among the listeners, estimated by the mean intersubject correlation was 0.71 and Cronbach's alpha 0.98 [7].

8 Features	**R^2 = 0.909**	**Adjusted R^2 = 0.900**	
Variable	**beta**	***sr*²**	**p-value**
On Dens. - Harmonic	0.205	0.033	0.000***
On Dens. - Bass	0.130	0.007	0.016*
On Dens. - Perceptual	0.302	0.018	0.000***
On Dens. - Strong	-0.155	0.010	0.004**
Strong Cluster IOI	0.127	0.006	0.021*
Tempo	0.430	0.056	0.000***
Percussiveness	-0.095	0.005	0.041*
SF CQT	0.107	0.004	0.053
5 Features	**R^2 = 0.887**	**Adjusted R^2 = 0.880**	
On Dens. - Harmonic	0.239	0.049	0.000***
On Dens. - Perceptual	0.224	0.020	0.000***
Strong Cluster IOI	0.132	0.007	0.027*
Tempo	0.404	0.053	0.000***
SF CQT	0.225	0.032	0.000***

Table 1. MLR prediction of the perceptual feature *speed* from computed audio features. The variable sr^2 is the squared semipartial correlation coefficient.

For the model based on 8 features, 2 features (*Onset Density – Strong* and *Percussiveness*) gave a negative contribution. Notice that the difference in explained variance is only about 2 % between the two models, indicating that the features in the 5-feature model may contain almost all relevant information.

The PLS regression of the 8 features is shown in Table 2. With 3 components, the cross-validated adjusted R^2 indicates that just below 90 % of the variability could be explained. Note also that the cross-validation procedure only lowers the result marginally, supporting the validity of the present features.

PLS Regression – Speed (3 PLS-components)		
R^2 = 0.907	**Adj. R^2 = 0.903**	**Adj. R^2 cv = 0.878**
Component	**Explained variance**	**Cum. variance**
1	0.845	0.845
2	0.052	0.897
3	0.017	0.914

Table 2. PLS prediction of the perceptual feature *speed* from computed audio features. The squared correlation coefficient R^2 was derived using PLS, including 10-fold cross validation ("cv"). Also, R^2 as a function of the number of components is shown.

The fitted values of the linear regression from Table 1 (8-feature model) are shown in Figure 4 below. As seen in the Figure, the deviations from the target are rather evenly distributed across the range and with a maximal deviation of about one unit.

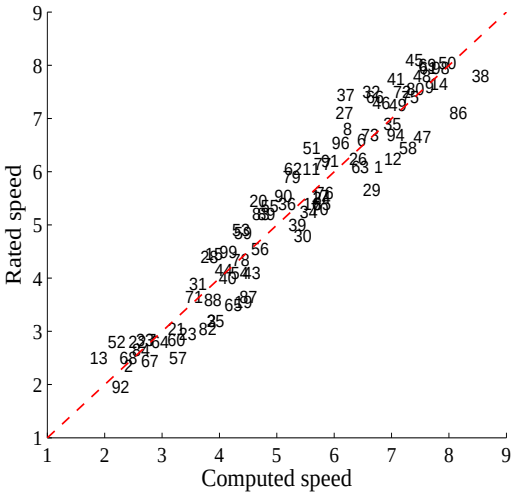


Figure 4. The fitted values in the MLR prediction of perceptual speed, where higher means faster. For each song, the x-axis represents the estimated speed and the y-axis represents the ground truth (derived from listeners).

4.3 Predicting Speed of the Test Set

Two linear models of speed (5 and 8 features) were derived from the multiple linear regression analysis of the training set shown in Table 1. The models were applied to the test set and the squared correlation between rated speed and computed speed is shown in Table 3.

No. of Features/Regression coefficients	**R^2**
5	0.934
8	0.894

Table 3. The prediction of the perceptual feature *speed* from a linear model using computed audio features.

The 5-feature model's prediction of speed for each song of the test set is shown in Figure 5. Computed speed

is approximately 1 unit higher than rated speed and this is probably due to the differences in the music examples of the databases. Furthermore, different scales were used for the listener ratings in the two data sets (1-9 and 0-10).

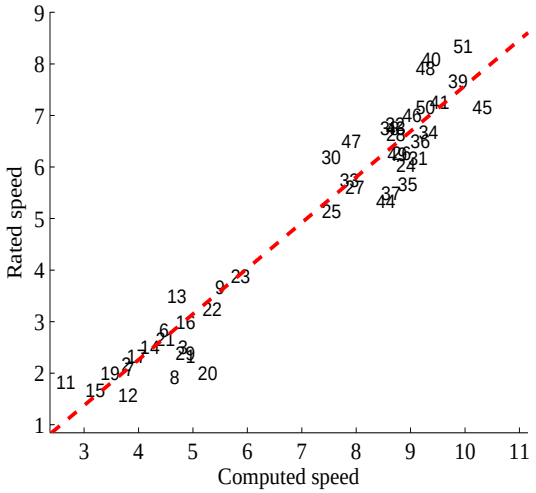


Figure 5. The prediction of the perceptual feature speed, where higher means faster. For each song, the x-axis represents the estimated speed and the y-axis represents the ground truth (derived from listeners).

5. CONCLUSIONS AND DISCUSSION

The models were able to explain about 90 % of the variability in listener ratings. The most important features were tempo together with onset densities for different layers of the music as well as spectral fluctuations in the harmonic part of the audio. The validity of the features was supported by cross-validation, and verified by using the extracted regression coefficients from the training set to accurately predict speed in the test set.

The results show that it was possible to reach the same high explained variance on audio data as on the symbolic data in [7] using similar features. This indicates that the appropriate low-level audio features have been extracted, which is reassuring for the ongoing study. The model based on 5 features was able to explain more of the variance in the test set than the model based on 8 features. This indicates that the 8-feature model was overfitting the training set.

The segmentation of audio (HP-separation and clustering) seems to be a promising path forward. By clustering onsets we can detect onsets belonging to the same source and thus use the rhythmic pattern of this source in the model. By using several onset detection functions on separate parts of the audio, different aspects of the music can be captured. Source separation can be motivated from an ecological perspective; it seems reasonable to assume that listeners distinguish between sounds from different sources to better understand the soundscape. A drawback with the proposed system is that the computation of several STFTs and CQTs is relatively time consuming.

In future work we intend to include songs without percussive elements. We also intend to investigate other high level rhythmic features such as rhythmic complexity and dynamics. We expect the audio segmentation to be a fruitful way forward. Data from this study is freely available for research purposes[2].

6. ACKNOWLEDGEMENT

This work was supported by the Swedish Research Council, Grant Nr. 2009-4285 and 2012-4685.

7. REFERENCES

[1] M. Alonso, G. Richard and B. David: "Accurate Tempo Estimation Based on Harmonic + Noise Decomposition," *Journal on Advances in Signal Processing*, 2007.

[2] R. Bresin, and A. Friberg: "Emotion Rendering in Music: Range and Characteristic Values of Seven Musical Variables," *Cortex*, Vol. 47, No. 9, pp. 1068-1081, 2011.

[3] S. Dixon: "Onset detection revisited," In *Proc. of DAFx*, pp. 133–137, 2006.

[4] C. Duxbury, J. P. Bello, M. Sandler, and M. Davies: "A Comparison between Fixed and Multiresolution Analysis for Onset Detection in Musical Signals," In *Proc. of DAFx,* pp. 207-212, 2004.

[5] T. Eerola, O. Lartillot, P. Toiviainen: "Prediction of Multidimensional Emotional Ratings in Music from Audio using Multivariate Regression Models," In *Proc. of ISMIR,* pp. 621-626, 2009.

[6] D. FitzGerald: "Harmonic/Percussive Separation Using Median Filtering," In *Proc. of DAFx,* 2010.

[7] A. Friberg, E. Schoonderwaldt, A. Hedblad, M. Fabiani, and A. Elowsson: "Perceptually derived features can be used in music information retrieval," submitted.

[8] A. Friberg, E. Schoonderwaldt, and A. Hedblad: "Perceptual Ratings of Musical Parameters," In von H. Loesch, and S. Weinzierl (Eds.), *Gemessene Interpretation - Computergestützte Aufführungsanalyse im Kreuzverhör der Disziplinen,* pp. 237-253, Mainz: Schott, 2011.

[9] A. Gabrielsson: *Studies in rhythm*, doctoral dissertation, Uppsala University, 1973.

[10] A. Hedblad: *Evaluation of musical feature extraction tools using perceptual ratings*. Master thesis, KTH Royal Institute of Technology, 2011.

[11] J. Hockman and I. Fujinaga: Fast vs Slow: Learning Tempo Octaves from User Data. In *Proc. of ISMIR*, pp. 231-236, 2010.

[12] G. Madison, and J. Paulin: "Ratings of Speed in Real Music as a Function of both Original and Manipulated Beat Tempo." *Journal of the Acoustical Society of America*, Vol. 128, No. 5, pp. 3032-3040, 2010.

[13] A. Klapuri: "Sound Onset Detection by Applying Psychoacoustic Knowledge," In *Proc. IEEE Conf. Acoustics, Speech and Signal Processing*, 1999.

[14] O. Lartillot and P. Toiviainen: A Matlab Toolbox for Musical Feature Extraction from Audio. In *Proc. of DAFx*, pp. 237-244, 2007.

[15] M. Levy: Improving Perceptual Tempo Estimation With Crowd-Sourced Annotations. In *Proc. of ISMIR*, pp. 317-322, 2011.

[16] G. Peeters and J. Flocon-Cholet: Perceptual Tempo Estimation Using GMM Regression. In *Proc. of ACM MIRUM*, pp. 45-50, Japan, November 2012.

[17] C. Schörkhuber and A. Klapuri: "Constant-Q Transform Toolbox for Music Processing," In 7th Sound and Music Conference, Barcelona, 2010.

[2] www.speech.kth.se/music/speed

INTER AND INTRA ITEM SEGMENTATION OF CONTINUOUS AUDIO RECORDINGS OF CARNATIC MUSIC FOR ARCHIVAL

Padi Sarala
Computer Science and Engineering
Indian Institute of Technology, Madras
padi.sarala@gmail.com

Hema A.Murthy
Computer Science and Engineering
Indian Institute of Technology, Madras
hema@cse.iitm.ac.in

ABSTRACT

The purpose of this paper is to segment carnatic music recordings into individual items for archival purposes using applauses. A concert in carnatic music is replete with applauses. These applauses may be inter-item or intra-item applauses. A property of an item in carnatic music, is that within every item, a small portion of the audio corresponds to the rendering of a composition which is rendered by the entire ensemble of lead performer and accompanying instruments. A concert is divided into segments using applauses and the location of the ensemble in every item is first obtained using Cent Filterbank Cepstral Coefficients (CFCC) combined with Gaussian Mixture Models (GMMs). Since constituent parts of an item are rendered in a single *raga*, *raga* information is used to merge adjacent segments belonging to the same item. Inter-item applauses are used to locate the end of an item in a concert. The results are evaluated for fifty live recordings with 990 applauses in total. The classification accuracy for inter and intra item applauses is 93%. Given a song list and the audio, the song list is mapped to the segmented audio of items, which are then stored in the database.

1. INTRODUCTION

Indian classical music consists of two popular traditions, namely, Carnatic and Hindustani. In most of the carnatic music concerts audio recordings are continuous and unsegmented [1]. A concert in carnatic music is replete with applauses. As most Indian music is improvisational, the audience applauds the artist spontaneously. Thus, in a given performance, there can be a number of applauses even within a single item. The general structure of a concert is shown in Figure 1. As shown in Figure a concert is made up of a number of different items. Each item can optionally have a solo vocal, a solo violin, a solo percussion (referred to as Thani in the Figure) but a composition is mandatory. Generally, the audience applauds the artist at the end of every item. Owing to the spontaneity that can be leveraged by an artist in a carnatic music concert, the concert is more like a dialogue between the artist and the audience [10]. Aesthetic phrases are immediately acknowledged by the audience. Owing to this aspect, applauses can occur anywhere in the concert. Different types of items and their constituent segments are shown in Figure 2. An item can be i) a single composition, ii) a vocal followed by the violin and then a composition, iii) a composition followed by ThaniAvarthanam, iv) a ragam tanam pallavi (RTP) that consists of solo pieces of vocal , violin, and the pallavi, which is equivalent to a composition for the purpose of analysis. The composition itself can be rendered in tandem by a lead performer, followed by accompanying instruments. For better understanding of carnatic music terms, we refer the reader to the supplemental material [2].

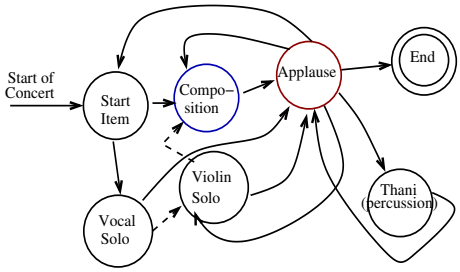


Figure 1: *General structure of carnatic music concert.*

From the Figure 2 it is clear that a composition segment is mandatory in an item. Furthermore an item can consist of one or more composition segments as shown in Figure 3. This is especially true for the Main Song in a concert, where different parts of the composition lend itself to significant improvisation. This leads to the composition being fragmented into a number of segments owing to applauses. A composition segment is also the last segment in an item. Since compositions are mandatory in every item, the begin and end of an item can be found in the vicinity of composition segments. To determine whether a sequence of composition segments belong to the same item or not, information about *raga* is required. In general, an item can be performed in single *raga*. The characteristics of *raga* can be detected by

[1] "http://www.sangeethapriya.org"

[2] "http://xa.yimg.com/kq/groups/2785355/2047532645/name/Carnatic+music+terminology.pdf"

melodic histograms [6]. So, to identify different items , melodic histograms can be used. This will not work when an item is intrinsically performed in multiple *ragas*. Such items are quite rare in a concert and seldom have multiple composition segments like *viruttam* or a vocal solo sung in isolation.

The first task for archival of carnatic music recordings requires that the audio should be segmented into individual items. A finer segmentation of the concert into Vocal solo, Violin solo, Song (Composition), and percussion solo, vocal-violin tandem, different percussion instruments in tandem may be of relevance to a keen learner, listener or researcher. In this paper, an attempt is made to determine the fine segments using applauses. The fine segments are first labeled using supervised Gaussian Mixture Models (GMM). Most of the segments in an item are rendered in single *raga*, *raga* information is used to merge the adjacent segments belonging to same item. Given an item list (text), the items of audio are aligned. This results in a database where the items can be queried using the text corresponding to that of the item.

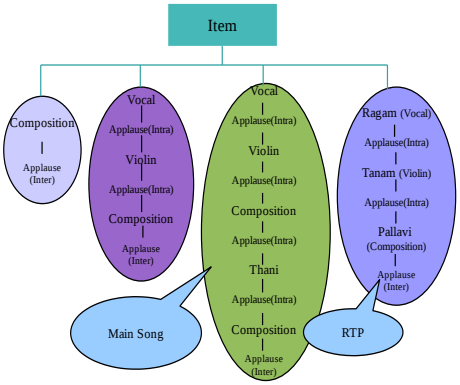


Figure 2: *Different types of items in a carnatic music concert.*

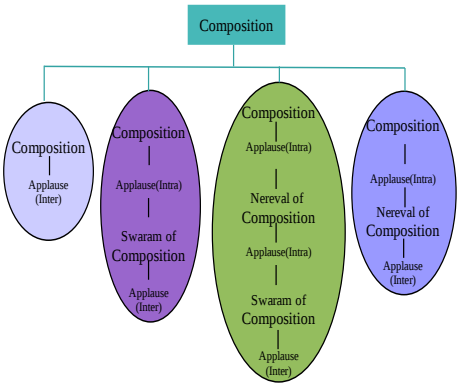


Figure 3: *Different types of compositions in a carnatic music concert.*

The remaining part of the paper is organised as follows. In Section 2 we discuss the process of segmentation of a concert and identification of different composition segments . Section 2 also discusses previous work for locating the applauses in a concert using different spectral domain features [11]. Based on composition segments in a concert usage of pitch histograms to merge the composition segments belonging to a same *raga* is discussed in Section 3. Section 4 discusses for a given meta-data in terms of text corresponding to items in a concert the meta-data is used to index the audio. In Section 5 we discusses about the database used for the experimental evaluation and the results obtained for segmentation and, inter and intra item classification. Finally, Section 6 concludes the work.

2. SEGMENTATION OF CARNATIC MUSIC CONCERT

In Subsection 2.1 we discuss identification of applause locations in a concert using different features. And Subsection 2.2 explains how these identified applause locations are used for segmentation of a concert for identifying the composition segments.

2.1 Applause Analysis

In a carnatic music concert applauses can occur at the end of an item or within an item, as explained in Section 1. The paper [11] discusses applause identification in carnatic music and its applications to archival of carnatic music. The paper also discusses the use of Cumulative Sum (CUSUM) [3] to highlight the important aspects of a concert. Spectral domain features like spectral flux and spectral entropy are used for detecting the applause locations for the given concert. Figure 4 shows applause locations (peaks) in a 3 hour concert using CUSUM of spectral entropy as the feature. This concert has 25 applauses and 9 items. The complete details about the applause identification and thresholds used for detection of applauses are explained in Section 5.

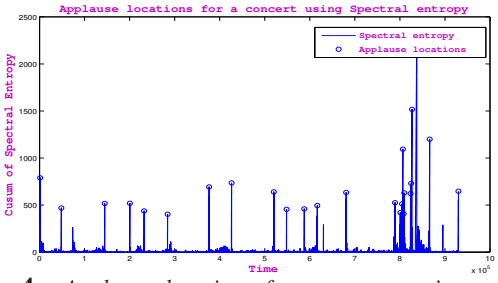


Figure 4: *Applause locations for a concert using spectral entropy feature.*

2.2 Identifying the Composition Segment Locations in a Concert

In order to find inter or intra item applauses we need to find change in *raga* among all the segments. As explained in Section 1, the end of an item is always after the composition segment. Therefore, composition segment locations probably indicates an end of an item. Owing to this, composition segments are identified in a concert. In order to find the composition segments in a concert, GMMs are built for four classes, namely a Vocal solo, a Violin solo, a Composition ensemble and ThaniAvarthanam using Cent Filterbank based Cepstral Coefficients (CFCC).

In carnatic music each singer renders every item with respect to a reference called Tonic (*sadja*) [1]. Any melodic analysis of Indian music therefore requires a

normalisation with the tonic. Within a concert the lead performer chooses the tonic and the accompanying instruments are also tuned it. Across concerts the same musician can have different tonic. Nevertheless, the tonic chosen for a concert is maintained throughout the concert using an instrument called the tambura that provides the drone. Therefore, the analysis of a concert depends on the tonic. CFCCs are proposed in this paper to perform normalisation on the spectrum. The fundamental difference between Mel Frequency Cepstral Coefficients (MFCC) and CFCC is that, in CFCC the frequency spectrum of the signal is mapped to the cent scale (after normalising with tonic) as opposed to the Mel scale. Extraction of CFCCs is detailed below:

1. The audio signal is divided into frames.
2. The short-time discrete Fourier is computed for each frame.
3. The power spectrum is then multiplied by a bank of filters that are spaced uniformly in the tonic normalised cent scale. The cent scale is defined as:

$$cent = 1200 \times \log_2 \left(\frac{f}{tonic} \right) \quad (1)$$

4. The energy in each filter is computed. The logarithm of filterbank energies is computed.
5. Discrete cosine transform (DCT-II) of log filter bank energies is computed.
6. The cepstral coefficients obtained after DCT computation are used as features for building the GMMs.

Alternatively, the chroma filterbanks can also be used . The chroma filterbanks discussed in [7], is primarily for the Western classical music where the scale is equitemperament and is characterised by a unique set of 12 semitones, subsets of which are used in performances. As indicated in [12], Indian music pitches follow a just intonation rather than an equitemperament intonation. In [9], it is shown that even just intonation is not adequate because pitch histogram across all *raga*s of carnatic music appears to be more or less continuous. To account for this, the chroma filterbanks include a set of overlapping filters. Furthermore, the filters can span less than a semitone, to account for the fact that in Indian music two adjacent *svaras* need not be separated by a semitone. Cepstral coefficients derived using the Chroma filterbanks are referred to as ChromaFCC.

Figure 5 shows MFCC, ChromaFCC and CFCC features for three types of segments, Vocal, Violin, and Composition. As shown in the Figure 5 the mel filterbanks emphasises mostly timbre, the chroma filterbanks emphasises the *svaras*, and the cent filterbanks emphasises both *svara* and the timbre around the melodic frequency. In this paper, we therefore use CFCC to distinguish different types of segments. To find CFCC cepstral coefficients we need to identify the tonic for every concert. To this extent pitch histograms are used to find the tonic as explained in [1]. Table 1 (Subsection 5.1) provides concerts for each singer and the tonic values identified using the pitch histograms. GMMs [2] are

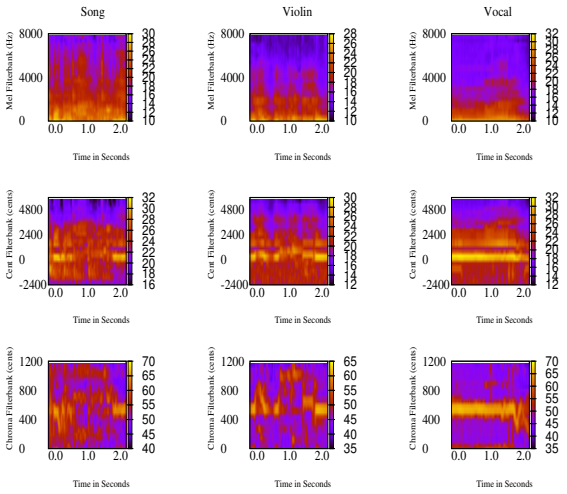


Figure 5: *Time frequency representations of Composition (Song), Vocal, Violin using MFCC, CFCC and ChromaFCC.*

frequently used for speaker verification and identification, and for segmentation tasks. In this paper, GMMs are used to segment the given concert using CFCC features. Complete details about the training of GMMs, segmentation of the concert and its accuracy is explained in Section 5.

3. INTER AND INTRA ITEM CLASSIFICATION

As explained in Subsection 2.2, composition segment locations might probably indicate the end of an item. Figure 3 shows that a composition can be fragmented into number of segments owing to applauses. As a result, an item can have one or more composition segments. We first explain how the adjacent composition segments belonging to same the *raga* are merged into a single item using the pitch histograms. Following it we provide an algorithm used for classifying the inter-item and intra-item applauses for finding the number of items in a concert for archival purpose.

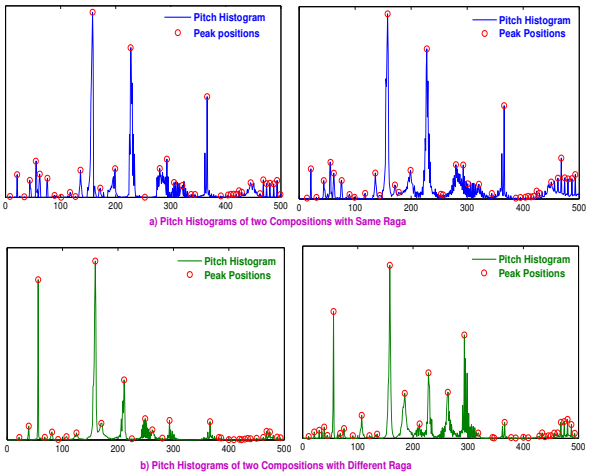


Figure 6: *Pitch histograms for same raga and different raga segments.*

Pitch is an auditory sensation in which a listener assigns musical tones to relative positions on a musical

scale based primarily on the frequency of vibration. *Raga* is one of the melodic modes used in Indian classical music. *Raga* uses a sequence of *svaras* upon which a melody is constructed [9]. However, the way the *svaras* are approached and rendered in musical phrases and the mood they convey are important in defining a *raga* than the *svaras* themselves. The note locations themselves are not absolute in carnatic music. Even for two *raga*s with the same set of notes, the rendered note locations can be distinct. As the purpose in this paper is not to identify the *raga*s but to detect the change in *raga*, pitch histograms are used [6]. Also, for the given task, it is irrelevant if the pitch corresponding to that of the lead artist or the accompanying artists is picked up.

Yin algorithm [5] is used for extracting the pitch for the segments. Pitch histograms for a segment show the relative notes sung by a musician [6]. Based on the pitch histograms, adjacent segments can be compared. Figure 6 (a) shows the pitch histogram for the adjacent segments belonging to the same *raga*, while Figure 6 (b) shows the pitch histogram for the adjacent segments that belong to different *raga*s. It can be observed that, in the pitch histograms corresponding to that of the same *raga* the positions of the peaks are more or less the same. On the other hand, for adjacent segments belonging to different *raga*s, the peaks are different. Clearly, the heights of the peaks are not necessarily be the same. Further, some peaks may be missing. This primarily means that in that specific segment, the musician's improvisation was restricted to a set of notes. Similarity between adjacent segments can be calculated by warping along the cent axis using dynamic programming [8]. If the similarity measure between the adjacent segments is above threshold 'α', then adjacent segments belong to different *raga*s otherwise they belong to the same *raga*. Finding a single threshold 'α' across all the concerts is crucial. A line search is performed and a threshold is chosen empirically. For the concerts in question, 'α' value '750' is chosen.

We now summarise the overall algorithm distinguishing inter-item and intra-item applauses as follows:

1. Applauses are identified for a concert using spectral domain features.
2. GMMs are built for Vocal solo, Violin solo, Composition ensemble and ThaniAvarthanam using CFCC based cepstral coefficients.
3. Audio segments between a pair of the applauses are labeled as Vocal solo, Violin solo, Composition, and ThaniAvarthanam using the trained GMMs.
4. The composition segments are located and the pitch histograms are calculated for the composition segments.
5. Similarity measure is computed on the warped histograms of the adjacent composition segments. If the similarity measure is above the threshold 'α', then the composition segment is inter-item otherwise, intra-item.
6. Based on the inter-item locations, intra-item segments are merged into the corresponding items. A concert is thus segmented into items for archival purposes.

Figure 7 shows the overall procedure for identifying the inter and intra items in a concert. In this Figure, the first column corresponds to the audio with the applause locations marked. The second column indicates the different segments based on applause locations. The third column corresponds to the labeling of all these segments into Vocal solo, Violin solo, Composition and ThaniAvarthanam. In the last column, some of the segments are merged based on the similarity measure. Observe that in Figure 7 two composition segments 4 and 5 are merged into single item 4 based on the distance measure.

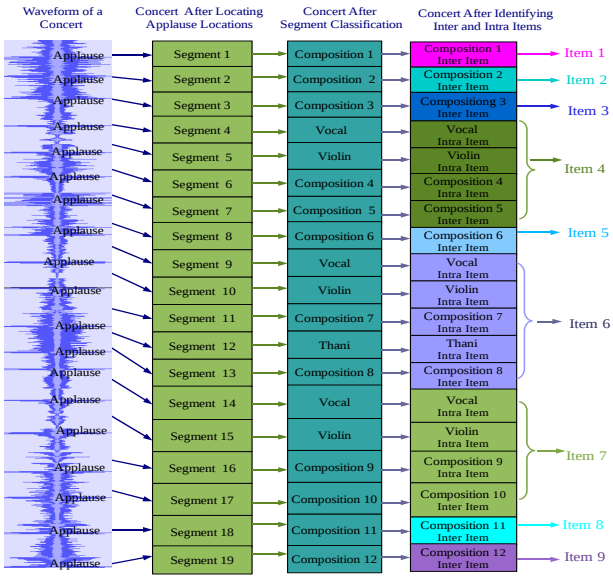


Figure 7: *Overall procedure for identifying inter and intra items in a concert.*

4. MAPPING THE COMPOSITION LIST TO ITEMS

In many performances, the meta-data is available for a concert in terms of composition lists from the audience [3]. The composition list obtained from the audience is matched to the items obtained from the algorithm discussed in Section 3. Figure 8 shows the mapping of the composition list obtained from the audience to inter-items for archival purposes. We evaluate 10 live recordings of male and female singers for mapping. The details of mapping and the mismatches (if any) are explained in Section 5.

5. EXPERIMENTAL EVALUATION

In this section we first give brief introduction to the database used for our study and then describe the experimental setup. We then provide the results obtained from various steps of the algorithm.

5.1 Database Used

For evaluation purpose, 50 live recordings of male and female singers are taken [4]. All recordings correspond to

[3] "http://www.rasikas.org"

[4] These live recordings were obtained from a personal collection of audience, musicians. These were made available for research purposes only.

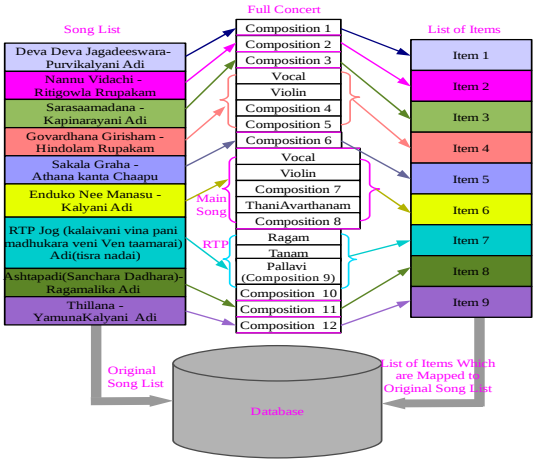


Figure 8: *Figure illustrates how the songs list is mapped to list of inter-items.*

concerts where the lead performer is a vocalist. Each concert is about 2-3 hours long. The total number of applauses across all the concerts is 990, where 394 applauses are inter-item applauses and 596 applauses are intra-item applauses. The 50 concerts consist of 394 items. All recordings are sampled at 44.1KHz sampling frequency with 16 bit resolution. For feature extraction, the analysis window chosen is 100 $msec$ and hop size is set to 10 $msec$ Table 1 gives the details of the database used. It can be observed that for a given singer, there are different possible tonic values, calculated using pitch histograms.

Singer Name	No. of Concerts	Duration (Hrs)	No. of Applause	Different Tonic
Male 1	4	12	89	158,148,146,138
Female 1	4	11	81	210, 208
Male 2	5	14	69	145, 148,150,156
Female 2	1	3	16	198
Male 3	4	12	113	145,148
Female 3	1	3	15	199
Male 4	26	71	525	140,138,145
Male 5	5	14	62	138,140

Table 1: *Database used for study, different Tonic values identified for each singer using pitch histograms.*

5.2 Experimental Setup

1. For all the 50 recordings spectral flux and spectral entropy features are extracted with a window size of 100 $msec$ with overlap of 10 $msec$. Applauses are marked for all the recordings using the Sonic-Visualizer [4]. These marked applauses are used as the ground truth for finding the applause detection accuracy.

2. To build the GMMs for the segmentation of the concert Vocal, Violin, ThaniAvarthanam, and Composition segments are manually segmented from the database. From male (female) recordings three segments are chosen for training the GMM for each class. MFCC, ChromaFCC, and CFCC features of 20 dimensions are extracted. GMMs with 32 mixtures are built for each of the four classes.

3. All the 50 recordings are segmented based on applause locations and all these segments are manually labeled as Vocal, Violin, ThaniAvarthanam, and Composition. Labeled data is used as the ground truth for finding the segmentation performance [5] . After segmenting the concerts based on the applauses, there are 990 segments in total and these segments are tested against the GMMs. All these segments are labeled using 1-best result.

4. For each concert, composition segments are highlighted using GMMs, and for these segments pitch is extracted using Yin algorithm [5]. These segments are merged into a single item if they belong to the same *raga* using the pitch histograms.

5. The performance measure used for evaluating each of the applause detection , segmentation performance and the inter-item, intra-item classification is

$$\text{Accuracy} = \frac{\text{Correctly Predicted Data}}{\text{Total Amount of Data}} \times 100\%$$

5.3 Experimental Results

The analysis of finding the end of item locations for all the concerts is done in the following three stages.

1. As explained in Section 3, first, applauses are located for using spectral domain features like spectral entropy and spectral flux. Table 2 shows the decision thresholds for applause and music discrimination and the applause detection accuracy based on the same thresholds. The applause detection accuracy is calculated at frame level.

Feature	Threshold range	Accuracy (%)
Spectral Flux (Nonorm)	0.2-1.0	85%
Spectral Flux (Peaknorm)	0.35-1.0	96%
Spectral Entropy	0.79-1.0	95%

Table 2: *Decision thresholds for applause and music discrimination and applause detection accuracy.*

2. Secondly, based on applause locations all concerts are segmented. The segments are labeled using GMMs with CFCC features. Table 3 shows the segmentation performance using MFCC, ChromaFCC, and CFCC features. Separate GMMs are built for male and female singers. Table clearly shows that CFCC features performs well in locating the composition segments in the concert. The overall segmentation accuracy for 50 concerts, including male and female recordings, is 95%.

Model	MFCC	ChromaFCC	CFCC
Male singers	78%	60%	90%
Female singers	92%	70%	97%

Table 3: *Segmentation performance using MFCC, ChromaFCC and CFCC.*

3. Finally, the composition segments in a concert are merged into a single item if they belong to the same *raga* using the pitch histograms. Table 4 shows the inter-item and intra-item classification accuracy for every singer. The overall classification accuracy for the 50 concerts is found to be 93%. We also evaluate our approach for mapping the songs list to the set of items which we obtain from the algorithm. Table 5 also shows the mapping

[5] Labeling was done by the first author and verified by a professional musician

performance. This mapping is evaluated for 10 live recordings of performances by six musicians for which songs lists were available.

Musician	Intra-item	Inter-item	Accuracy(%)
Male 1	65	24	90
Female 1	50	31	100
Male 2	37	32	100
Female 2	10	6	98
Male 3	75	38	93
Female	8	7	99
Male 4	317	227	93
Male 5	33	29	100

Table 4: *Inter and intra item classification accuracy.*

Musician	No. of concerts	Actual inter items	Predicted inter items	Mapping accuracy (%)
Male 1	1	6	7	96
Female 1	1	9	9	100
Male 2	1	7	7	100
Male 3	2	19	21	97
Male 4	2	13	14	97
Male 5	3	19	19	100

Table 5: *Performance of mapping the songs list to the set of items.*

In a concert every artist can optionally render a *ragamalika*, where a single item can be performed in different *raga*s. When a single composition is sung in *ragamalika*, the algorithm will still work, since the applause will be at the end of the item. On the other hand, for items like *viruttam* or *ragam tanam pallavi* with multiple *raga*s, owing to the extensive improvisational aspects in them, a number of applauses can occur intra-item. Such special cases are limits of the algorithm. Every concert may have one such item. For these types of items the lyrics could be same or the meter could be same. For such cases, the algorithm can be extended to include matching of the lyrics. The last item in a concert is a closure composition. This is identical in all concerts and is called the *mangalam*. When a musician continues without a pause before closure composition, the mangalam is combined with the penultimate item in the concert. This is not considered as an error in this paper.

6. CONCLUSION AND FUTURE WORK

In this work we proposed an algorithm for automatically segmenting continuous audio recordings of carnatic music into items for archival purpose. An item in a concert has a structure and each segment has specific spectral properties. These properties are exploited to label the segments. In particular, a new feature called CFCC performs well for labeling the segments when compared to MFCC and ChromaFCC features. Given meta-data in terms of names of items in a concert, the audio and the meta-data are aligned. The directions for future work will be, we need to evaluate the inter-item and intra-item classification technique for the special cases like *ragamalika*, *RTP*, *viruttam*, and a vocal solo (sung in isolation), where adjacent segments belong to different *raga*s but correspond to a single item. This requires additional analysis in terms of lyrics, meter which are quite nontrivial.

7. ACKNOWLEDGEMENTS

This research was partly funded by the European Research Council under the European Unions Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583).

8. REFERENCES

[1] Ashwin Bellur, Vignesh Ishwar, Xavier Serra, and Hema A. Murthy. A knowledge based signal processing approach to tonic identification in indian classical music. In *International CompMusic Wokshop*, Instanbul, Turkey, 2012.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*, chapter 7. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] B E Brodsky and B S Darkhovsky. *Non-parametric Methods in change-point problems*. Kluwer Academic Publishers, New York, 1993.

[4] C Cannam, C Landone, M Sandler, and J.P Bello. The sonic visualiser: A visualisation platform for semantic descriptors from musical signals. In *7th International Conference on Music Information Retrieval (ISMIR-06)*, Victoria, Canada, 2006.

[5] A De Cheveigne and H Kawahara. Yin, a fundamental frequency estimator for speech and music. *Journal of the Acoustical Society of America*, page 111(4):19171930, 2002.

[6] P Chordia and A Rae. Raag recognition using pitch-class and pitch-class dyad distributions. *In Proc. of ISMIR*, pages 431–436, 2007.

[7] Dan Ellis. Chroma feature analysis and synthesis. `http://www.ee.columbia.edu/~dpwe/resources/Matlab/chroma-ansyn`, 2007.

[8] Toni Giorgino. Computing and visualizing dynamic time warping alignments in r: The dtw package. *Journal of Statistical Software*, 31(7):1–24, 8 2009.

[9] T M Krishna and Vignesh Ishwar. Svaras, gamaka, motif and raga identity. In *Workshop on Computer Music*, Instanbul, Turkey, July 2012.

[10] M V N Murthy. Applause and aesthetic experience. http://compmusic.upf.edu/zh-hans/node/151, 2012.

[11] Padi Sarala, Vignesh Ishwar, Ashwin Bellur, and Hema A Murthy. Applause identification and its relevance to archival of carnatic music. In *Workshop on Computer Music*, Instanbul, Turkey, July 2012.

[12] J Serra, G K Koduri, M Miron, and X Serra. Tuning of sung indian classical music. *In Proc. of ISMIR*, pages 157–162, 2011.

ESSENTIA: AN AUDIO ANALYSIS LIBRARY FOR MUSIC INFORMATION RETRIEVAL

Dmitry Bogdanov[1], **Nicolas Wack**[2], **Emilia Gómez**[1], **Sankalp Gulati**[1], **Perfecto Herrera**[1]
Oscar Mayor[1], **Gerard Roma**[1], **Justin Salamon**[1], **José Zapata**[1] **and Xavier Serra**[1]
Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
[1]{name.surname}@upf.edu
[2]essentia@wackou.otherinbox.com

ABSTRACT

We present Essentia 2.0, an open-source C++ library for audio analysis and audio-based music information retrieval released under the Affero GPL license. It contains an extensive collection of reusable algorithms which implement audio input/output functionality, standard digital signal processing blocks, statistical characterization of data, and a large set of spectral, temporal, tonal and high-level music descriptors. The library is also wrapped in Python and includes a number of predefined executable extractors for the available music descriptors, which facilitates its use for fast prototyping and allows setting up research experiments very rapidly. Furthermore, it includes a Vamp plugin to be used with Sonic Visualiser for visualization purposes. The library is cross-platform and currently supports Linux, Mac OS X, and Windows systems. Essentia is designed with a focus on the robustness of the provided music descriptors and is optimized in terms of the computational cost of the algorithms. The provided functionality, specifically the music descriptors included in-the-box and signal processing algorithms, is easily expandable and allows for both research experiments and development of large-scale industrial applications.

1. INTRODUCTION

There are many problems within the Music Information Research discipline which are based on audio content and require reliable and versatile software tools for automated music analysis. Following the research necessities, different audio analysis tools tailored for MIR have been developed and used by academic researchers within the last fifteen years. These tools include the popular MIRtoolbox [39] and MARSYAS [69], jAudio [48], jMIR [47], Aubio [11], LibXtract [12], yaafe, [1] Auditory Toolbox [60], and CLAM Music Annotator [2]. Furthermore, a number of Vamp plugins [2] were developed by different researchers for computation and visualization of music descriptors using hosts such as Sonic Visualiser software [13], Sonic Annotator, [3] and Audacity. [4] Apart from these software tools, there is the online service Echonest, [5] which provides a collection of audio features for any uploaded track via an API. This service, however, does not disclose the implementation details of the features offered to a user, which might be a significant limitation at least in the context of academic research. The above-mentioned tools provide different limited sets of descriptors, and require different programming languages (Matlab, C++, Java) and software environments (standalone GUI-based or command-line applications, running within a Vamp host) to be extracted. The feature sets vary from tool to tool and one may have to combine different tools in order to obtain the desired expanded combination of features. The comparative overview of the majority of the available tools is presented in [50]. Surely, all the above-mentioned tools are very important for the research community, nevertheless we believe there is a lack of generic robust tools which provide more comprehensive sets of state-of-the-art music descriptors and are optimized for faster computations on large collections.

2. ESSENTIA 2.0

We present Essentia 2.0, an extensive open-source library for audio analysis and audio-based music information retrieval released under the Affero GPL [6] license and well-suited for both research and industrial applications. [7] In its core, Essentia is comprised of a reusable collection of algorithms to extract features from audio. The available algorithms include audio file input/output functionality, standard digital signal processing (DSP) building blocks, filters, generic algorithms for statistical characterization, and spectral, temporal, tonal and high-level music descriptors. The library is written in C++, which provides considerable performance benefits. Importantly, it also includes Python bindings to facilitate the usage of the library for the users

[1] http://sourceforge.net/projects/yaafe

[2] http://vamp-plugins.org/download.html
[3] http://omras2.org/sonicannotator
[4] http://audacity.sourceforge.net
[5] http://developer.echonest.com
[6] http://gnu.org/licenses/agpl.html
[7] Commercial licensing is offered in addition to the Affero GPL.

who are familiar with the matlab/python environment. Using Essentia's dedicated python modules, one can rapidly get familiar with the available algorithms and design research experiments, explore and analyze data on-the-fly. In addition, the majority of the MIR descriptors' algorithms are wrapped into a Vamp [8] plugin and can be used with the popular Sonic Visualiser software [13] for visualization of music descriptors.

The design of Essentia is focused on robustness, time and memory performance, and ease of extensibility. The algorithms are implemented keeping in mind the use-case of large-scale computations on large music collections. One may develop his own executable utility with a desired processing flow using Essentia as a C++ library. Alternatively a number of executable extractors are included with Essentia, covering a number of common use-cases for researchers, for example, computing all available music descriptors for an audio track, extracting only spectral, rhythmic, or tonal descriptors, computing predominant melody and beat positions, and returning the results in yaml/json data formats.

Essentia has been in development for more than 6 years incorporating the work of more than 20 researchers and developers through its history. The 2.0 version contains the latest refactoring of the library, including performance optimization, simplified development API, and a number of new descriptors such as the state-of-the-art beat tracking [16, 74, 75] and predominant melody detection [56] algorithms. In addition, Essentia can be optionally complemented with Gaia, [9] a library released under the same license, which allows to apply similarity measures and classifications on the results of audio analysis, and generate classification models that Essentia can use to compute high-level description of music. Gaia is a C++ library with python bindings for working with points in high-dimensional spaces, where each point represents a song and each dimension represents an audio feature. It allows to create datasets of points, apply transformations (gaussianization, principal component analysis, relevant component analysis, classification with support vector machines), and compute custom distance functions. The functionality of Gaia can be used to implement search engines relying on item similarity of any kind (not limited to music similarity).

3. OVERALL ARCHITECTURE

The main purpose of Essentia is to serve as a library of signal-processing blocks. As such, it is intended to provide as many algorithms as possible, while trying to be as little intrusive as possible. Each processing block is called an Algorithm, and it has three different types of attributes: inputs, outputs and parameters. Algorithms can be combined into more complex ones, which are also instances of the base Algorithm class and behave in the same way. An example of such a composite algorithm is presented in Figure 1. It shows a composite tonal key/scale extractor based on [28], which combines the algorithms for frame cutting, windowing, spectrum computation, spectral peaks detection, chroma features (HPCP) computation and finally the algorithm for key/scale estimation from the HPCP (itself a composite algorithm).

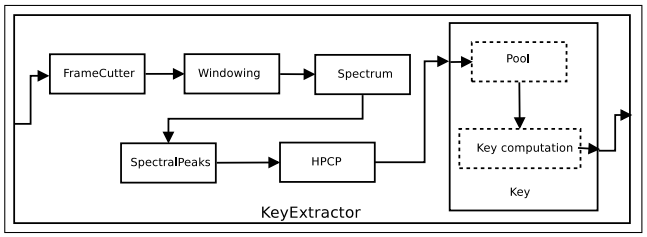


Figure 1. An example of the composite algorithm KeyExtractor combining the algorithms FrameCutter, Windowing, Spectrum, SpectralPeak, HPCP, and the Key algorithm composite itself.

The algorithms can be used in two different modes: *standard* and *streaming*. The standard mode is imperative while the streaming mode is declarative. The standard mode requires to specify the inputs and outputs for each algorithm and call their processing function explicitly. If the user wants to run a network of connected algorithms, he/she will need to manually run each algorithm. The advantage of this mode is that it allows very rapid prototyping (especially when the python bindings are coupled with a scientific environment in python, such as ipython, numpy, and matplotlib).

The streaming mode, on the other hand, allows to define a network of connected algorithms, and then an internal scheduler takes care of passing data between the algorithms inputs and outputs and calling the algorithms in the appropriate order. The scheduler available in Essentia is optimized for analysis tasks, and does not take into account the latency of the network. For real-time applications, one could easily replace this scheduler with another one that favors latency over throughput. The advantage of this mode is that it results in simpler and safer code (as the user only needs to create algorithms and connect them, there is no room for him to make mistakes in the execution order of the algorithms), and in lower memory consumption in general, as the data is streamed through the network instead of being loaded entirely in memory (which is the usual case when working with the standard mode).

Even though most of the algorithms are available for both the standard and streaming mode, the code that implements them is not duplicated as either the streaming version of an algorithm is deduced/wrapped from its standard implementation, or vice versa.

4. ALGORITHMS

In this section we briefly review the most important algorithms included in the library. The interested reader is referred to the documentation for a complete reference for all available algorithms. [10] A great part of them computes a variety of low-level, mid-level, and high-level descrip-

[8] http://vamp-plugins.org

[9] http://github.com/MTG/gaia

[10] http://essentia.upf.edu

tors useful for MIR. In addition there are tools for working with audio input/output and processing, and gathering data statistics.

4.1 Audio input/output and filtering

Essentia has a variety of audio loaders to provide a very convenient way to load audio files from disk. These loaders output the stream of stereo or mono samples using the FFmpeg [11] /Libav [12] libraries. Almost all existing audio formats are supported, and additionally audio can be loaded from video files or even Flash files. It is possible to downmix a file to mono, resample it, trim the audio to a given start/end time, normalize the resulting samples using a given ReplayGain [13] value, and apply an Equal-loudness filter [14] to the resulting audio. A special loader reads the metadata tags stored in the given file (e.g., ID3 tags). Essentia can also write audio files to any format supported by FFmpeg/Libav, and may add beeps to the audio according to the given (onset) time positions.

In addition, Essentia provides algorithms for basic processing of audio streams: it allows to apply replay gain, frame cutting, windowing, resampling, FFT, auto-correlation computation, etc. A variety of audio filters are implemented in Essentia, including the generic IIR filtering, first and second order low/band/high/all-pass filtering and band rejection, DC component removal, moving average filter, and the filter approximating an equal-loudness curve.

4.2 Spectral descriptors

A variety of algorithms for computation of low-level spectral descriptors is included. In particular, they compute:

- the energy of the given frequency band or the set of bands of a spectrum, the Bark band energies [51] of a spectrum, the Mel band energies and the Mel-frequency cepstral coefficients of a spectrum using the MFCC-FB40 algorithm [21];
- the ERB band energies of a spectrum [49] and the Gammatone feature cepstral coefficients [58] similar to MFCCs;
- the Linear Predictive Coding coefficients and the associated reflection coefficients [43];
- the spectral flux [17, 68];
- the high-frequency content (HFC) measure [33, 45], the roll-off frequency of a spectrum [51], and the spectral contrast feature [1];
- the spectral peaks and the spectral complexity [41], the inharmonicity and dissonance measures [51, 52], and the Strong Peak of a spectrum [23];
- the stereo panorama distribution [30];
- spectral whitening based on spectrum envelope estimation in [54].

[11] http://ffmpeg.org

[12] http://libav.org

[13] http://wiki.hydrogenaudio.org/index.php?title=ReplayGain_1.0_specification

[14] http://replaygain.hydrogenaudio.org/proposal/equal_loudness.html

4.3 Time-domain descriptors

A number of algorithms for computation of time-domain descriptors are included. They compute the total duration and the duration of the perceptually meaningful part of the signal above a certain energy level [51], Zero-crossing rate [51], and loudness estimations based on the Stevens' power law [65], the LARM model [59], the Equivalent sound level (Leq) [64], and the Vickers' model [70].

4.4 Tonal descriptors

A number of algorithms for computation of tonal descriptors are included. In particular, they provide:

- the pitch salience function of a signal, the estimation of the fundamental frequency of the predominant melody by the MELODIA algorithm [56], and the pitch estimation by the YinFFT method [11];
- the Harmonic Pitch-Class Profile (HPCP) of a spectrum (also called chroma features) [28], the tuning frequency [27], the key and the scale of a song [28];
- the sequence of chords present in a song [28], chords histogram, chords change rate, key and scale of the most frequent chord;
- the tristimulus [51] and the ratio of a signal's odd to even harmonic energy [44] computed based on harmonic peaks;
- the tonic frequency of the lead artist in Indian art music [55];
- a set of descriptors derived from high-resolution (10 cents) HPCP features related to tuning system or scale and used for comparative analysis of recordings from Western and non-Western traditions [29].

4.5 Rhythm descriptors

A number of the algorithms are related to a rhythmic representation of the audio signal. They include:

- the beat tracker based on the complex spectral difference feature [16], the multifeature beat tracker [74, 75] (which combines 5 different beat trackers taking into account the maximum mutual agreement between them), and the beat tracker based on the BeatIt algorithm [22];
- an extractor of the locations of large tempo changes from a list of beat ticks;
- the statistics of the BPM histogram of a song;
- the novelty curve for the audio signal, and the BPM distribution and tempogram based in it [25];
- onset detection functions for the audio signal including HFC [33, 45], Complex-Domain spectral difference [4] and its simplified version [10], spectral flux and Mel-bands based spectral flux [18], overall energy flux [38], spectral difference measured by the modified information gain, and the beat emphasis function [15], and the list of onsets in the audio signal given a list of detection functions [10];
- rhythm transform [26] (kind of the FFT of the MFCC representation);
- beat loudness (the loudness of the signal on windows centered around the beat locations).

4.6 SFX descriptors

A number of the algorithms are intended to be used with short sounds instead of full-length music tracks. They return the logarithm of the attack time for the sound, a measure of whether the maximum/minimum value of a sound envelope is located towards its beginning or end, the pitch salience, the normalized position of the temporal centroid of a signal envelope, the Strong Decay [23], the estimation of flatness of a signal envelope and descriptors based on its derivative.

4.7 Other high-level descriptors

In addition to the low-level descriptors, Essentia also contains the following mid- and high-level descriptors:

- the "danceability" of a song based on the Detrended Fluctuation Analysis [67], the intensity of the input audio signal (relaxed, moderate, or aggressive), and the dynamic complexity related to the dynamic range and the amount of fluctuation in loudness [66];
- the fade-ins/fade-outs present in a song and audio segmentation using the Bayesian Information Criterion [24];
- the principal component analysis and Gaia transformations to a set of descriptors, in particular, classification using the models pre-trained in Gaia. Currently, the following classifier models are available: musical genre (4 different databases), ballroom music classification, moods (happy, sad, aggressive, relaxed, acoustic, electronic, party), western/non-western music, live/studio recording, perceptual speed (slow, medium, fast), gender (male/female singer), dark/bright timbre classification, and speech/music. The approximate accuracies of these models are presented in [5].

4.8 Statistics

The algorithms for computing statistics over an array of values, or some kind of aggregation, are also provided. These algorithms allow to compute:

- the mean, geometric mean, power mean, median of an array, and all its moments up to the 5th-order, its energy and the root mean square (RMS);
- flatness, crest and decrease of an array, typically used to characterize the spectrum [51];
- variance, skewness, kurtosis of a probability distribution, and a single Gaussian estimate for the given list of arrays (returns the mean array, its covariance and inverse covariance matrices).

4.9 Extractors

As Essentia algorithms can themselves be composed of multiple algorithms, a few useful extractors have been written as algorithms. Given an audio track, they compute the loudness, the tuning frequency, all tonal information (key, scale, chords sequence, chords histogram, etc), the BPM and beat positions of a music track as well as other rhythm-related features, a variety of low-level features with/without an applied equal-loudness filter, and the overall track description with the majority of the available low-level, mid-level and high-level features.

In addition, a number of executable extractors are included with the library as examples of its application. These standalone examples can be used straight away for batch computation of descriptors with no need to dive into the API of the library. They include a number of specific extractors (predominant melody, beat tracking, key, MFCCs, etc.) as well as generic extractors returning the majority of the available descriptors in yaml/json formats. For industrial applications, one may need to decide on the descriptors and type of processing required, and implement his own extractor.

5. APPLICATIONS

Essentia has served in a large number of research activities conducted at Music Technology Group since 2006. It has been used for music classification [6, 71, 72], and in particular for mood classification [40, 41], semantic autotagging [61, 73], music similarity and recommendation [5, 6, 9, 14], visualization and interaction with music [5, 6, 34, 42, 63], sound indexing [31, 32, 53], detection of musical instruments in polyphonies [19], cover detection [57], instrument solo detection [20], and acoustic analysis of stimuli for neuroimaging studies [37]. Currently, it is actively used within the CompMusic [15] research project [35, 36]. The systems based on Essentia/Gaia have been enrolled in the the Music Information Retrieval Evaluation eXchange (MIREX) campaigns for the tasks of music classification [71,72], music similarity [7,8], autotagging [62], and beat detection [3], and they have usually ranked among the best ones.

Essentia and Gaia have been used extensively in a number of research projects, [16] including the CANTATA EU (recommendation system for music videos), SALERO EU (search engine for sound effects), PHAROS EU (music search and recommendation), Buscamedia, PROSEMUS and DRIMS (automated low-level and high-level semantic description of music), and SIEMPRE EU [46] (audio analysis for multimodal databases of music performances).

Furthermore, previous versions of Essentia have been exploited for industrial applications: it has been used in the non-commercial sound service Freesound [17] (large-scale content-based search of sound recordings), industrial products by BMAT [18] and Stromatolite [19] (music recommendation), Yamaha's BODiBEAT (automatic playlist generation for runners), and Steinberg's LoopMash (audio-based sample matching for music production).

In addition to the off-line audio analysis we expect our library to be potential for real-time applications. However

[15] http://compmusic.upf.edu

[16] Detailed information about the research projects can be found online: http://mtg.upf.edu/research/projects

[17] http://freesound.org

[18] http://bmat.com

[19] http://stromatolite.com

not all of the present algorithms can be used for such applications due to their computational complexity.

6. CONCLUSIONS

We have presented a cross-platform open-source library for audio analysis and audio-based music information research and development, Essentia 2.0. The library is versatile and may suit the needs of both researchers within MIR community and the industry. In our future work we will focus on expanding the library and the community of users, We plan to add new music descriptors, in particular, adding new semantic categories to the set of high-level classifier-based descriptors, and update the library for real-time applications. All active Essentia users are encouraged to contribute to the library.

The detailed information about Essentia is located at the official web page. [20] It contains the complete documentation for the project including the installation instructions. The source code is available at the official Github repository. [21]

7. ACKNOWLEDGMENTS

The work on Essentia has been partially funded by the PHAROS (EU-IP, IST-2006-045035), Buscamedia (CEN-20091026), CANTATA (ITEA 05010, FIT-350300-2006-33), SIGMUS (TIN2012-36650), CompMusic (ERC 267583), and TECNIO (TECCIT12-1-0003) projects. We acknowledge all the developers who took part in working on this project and Alba Rosado for her help in coordination.

8. REFERENCES

[1] V. Akkermans, J. Serrà, and P. Herrera. Shape-based spectral contrast descriptor. In *Sound and Music Computing Conf. (SMC'09)*, page 143–148, 2009.

[2] X. Amatriain, J. Massaguer, D. Garcia, and I. Mosquera. The clam annotator: A cross-platform audio descriptors editing tool. In *Int. Conf. on Music Information Retrieval (ISMIR'05)*, volume 5, 2005.

[3] E. Aylon and N. Wack. Beat detection using plp. In *Music Information Retrieval Evaluation Exchange (MIREX'10)*, 2010.

[4] J. P. Bello, C. Duxbury, M. Davies, and M. Sandler. On the use of phase and energy for musical onset detection in the complex domain. *Signal Processing Letters, IEEE*, 11(6):553–556, 2004.

[5] D. Bogdanov. *From music similarity to music recommendation: Computational approaches based on audio and metadata analysis*. PhD thesis, UPF, Barcelona, Spain, 2013. In press.

[6] D. Bogdanov, M. Haro, F. Fuhrmann, A. Xambó, E. Gómez, and P. Herrera. Semantic audio content-based music recommendation and visualization based on user preference examples. *Information Processing & Management*, 49(1):13–33, Jan. 2013.

[7] D. Bogdanov, J. Serrà, N. Wack, and P. Herrera. Hybrid similarity measures for music recommendation. In *Music Information Retrieval Evaluation Exchange (MIREX'09)*, 2009.

[8] D. Bogdanov, J. Serrà, N. Wack, and P. Herrera. Hybrid music similarity measure. In *Music Information Retrieval Evaluation Exchange (MIREX'10)*, 2010.

[9] D. Bogdanov, J. Serrà, N. Wack, P. Herrera, and X. Serra. Unifying low-level and high-level music similarity measures. *IEEE Trans. on Multimedia*, 13(4):687–701, 2011.

[10] P. Brossier, J. P. Bello, and M. D. Plumbley. Fast labelling of notes in music signals. In *Int. Symp. on Music Information Retrieval (ISMIR'04)*, page 331–336, 2004.

[20] `http://essentia.upf.edu`

[21] `http://github.com/MTG/essentia`

[11] P. M. Brossier. *Automatic Annotation of Musical Audio for Interactive Applications*. PhD thesis, QMUL, London, UK, 2007.

[12] J. Bullock and U. Conservatoire. Libxtract: A lightweight library for audio feature extraction. In *Int. Computer Music Conf. (ICMC'07)*, volume 9, 2007.

[13] C. Cannam, C. Landone, and M. Sandler. Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files. In *ACM Int. Conf. on Multimedia (MM'05)*, page 1467–1468, 2010.

[14] O. Celma, P. Cano, and P. Herrera. Search sounds an audio crawler focused on weblogs. In *7th Int. Conf. on Music Information Retrieval (ISMIR)*, 2006.

[15] M. E. P. Davies, M. Plumbley, and D. Eck. Towards a musical beat emphasis function. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2009. WASPAA '09*, pages 61–64, 2009.

[16] N. Degara, E. A. Rua, A. Pena, S. Torres-Guijarro, M. E. Davies, and M. D. Plumbley. Reliability-informed beat tracking of musical signals. *IEEE Trans. on Audio, Speech, and Language Processing*, 20(1):290–301, 2012.

[17] S. Dixon. Onset detection revisited. In *Int. Conf. on Digital Audio Effects (DAFx'06)*, volume 120, page 133–137, 2006.

[18] D. P. W. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[19] F. Fuhrmann and P. Herrera. Quantifying the relevance of locally extracted information for musical instrument recognition from entire pieces of music. In *Int. Society for Music Information Retrieval Conf. (ISMIR'11)*, 2011.

[20] F. Fuhrmann, P. Herrera, and X. Serra. Detecting solo phrases in music using spectral and pitch-related descriptors. *Journal of New Music Research*, 38(4):343–356, 2009.

[21] T. Ganchev, N. Fakotakis, and G. Kokkinakis. Comparative evaluation of various MFCC implementations on the speaker verification task. In *Int. Conf. on Speach and Computer (SPECOM'05)*, volume 1, page 191–194, 2005.

[22] F. Gouyon. *A computational approach to rhythm description: Audio features for the computation of rhythm periodicity functions and their use in tempo induction and music content processing*. PhD thesis, UPF, Barcelona, Spain, 2005.

[23] F. Gouyon and P. Herrera. Exploration of techniques for automatic labeling of audio drum tracks instruments. In *MOSART: Workshop on Current Directions in Computer Music*, 2001.

[24] G. Gravier, M. Betser, and M. Ben. Audio segmentation toolkit, release 1.2. Technical report, 2010. Available online: https://gforge.inria.fr/frs/download.php/25187/audioseg-1.2.pdf.

[25] P. Grosche and M. Müller. A mid-level representation for capturing dominant tempo and pulse information in music recordings. In *Int. Society for Music Information Retrieval Conf. (ISMIR'09)*, page 189–194, 2009.

[26] E. Guaus and P. Herrera. The rhythm transform: towards a generic rhythm description. In *Int. Computer Music Conf. (ICMC'05)*, 2005.

[27] E. Gómez. Key estimation from polyphonic audio. In *Music Information Retrieval Evaluation Exchange (MIREX'05)*, 2005.

[28] E. Gómez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, 18(3):294–304, 2006.

[29] E. Gómez and P. Herrera. Comparative analysis of music recordings from western and non-western traditions by automatic tonal feature extraction. *Empirical Musicology Review*, 3(3):140–156, 2008.

[30] E. Gómez, P. Herrera, P. Cano, J. Janer, J. Serrà, J. Bonada, S. El-Hajj, T. Aussenac, and G. Holmberg. Music similarity systems and methods using descriptors, 2009. WIPO Patent No. 2009001202.

[31] M. Haro, J. Serrà, P. Herrera, and A. Corral. Zipf's law in short-time timbral codings of speech, music, and environmental sound signals. *PLoS ONE*, 7(3):e33993, 2012.

[32] J. Janer, M. Haro, G. Roma, T. Fujishima, and N. Kojima. Sound object classification for symbolic audio mosaicing: A proof-of-concept. In *Sound and Music Computing Conf. (SMC'09)*, pages 297–302, 2009.

[33] K. Jensen and T. H. Andersen. Beat estimation on the beat. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'03)*, page 87–90, 2003.

[34] C. F. Julià and S. Jordà. SongExplorer: a tabletop application for exploring large collections of songs. In *Int. Society for Music Information Retrieval Conf. (ISMIR'09)*, 2009.

[35] G. K. Koduri, S. Gulati, P. Rao, and X. Serra. Raga recognition based on pitch distribution methods. *Journal of New Music Research*, 41(4):337–350, 2012.

[36] G. K. Koduri, J. Serrà, and X. Serra. Characterization of intonation in carnatic music by parametrizing pitch histograms. In *Int. Society for Music Information Retrieval Conf. (ISMIR'12)*, pages 199–204, 2012.

[37] S. Koelsch, S. Skouras, T. Fritz, P. Herrera, C. Bonhage, M. Kuessner, and A. M. Jacobs. Neural correlates of music-evoked fear and joy: The roles of auditory cortex and superficial amygdala. *Neuroimage*. In press.

[38] J. Laroche. Efficient tempo and beat tracking in audio recordings. *Journal of the Audio Engineering Society*, 51(4):226–233, 2003.

[39] O. Lartillot, P. Toiviainen, and T. Eerola. A matlab toolbox for music information retrieval. In C. Preisach, P. D. H. Burkhardt, P. D. L. Schmidt-Thieme, and P. D. R. Decker, editors, *Data Analysis, Machine Learning and Applications*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 261–268. Springer Berlin Heidelberg, 2008.

[40] C. Laurier. *Automatic Classification of Musical Mood by Content-Based Analysis*. PhD thesis, UPF, Barcelona, Spain, 2011.

[41] C. Laurier, O. Meyers, J. Serrà, M. Blech, P. Herrera, and X. Serra. Indexing music by mood: design and integration of an automatic content-based annotator. *Multimedia Tools and Applications*, 48(1):161–184, 2009.

[42] C. Laurier, M. Sordo, and P. Herrera. Mood cloud 2.0: Music mood browsing based on social networks. In *Int. Society for Music Information Retrieval Conf. (ISMIR'09)*, 2009.

[43] J. Makhoul. Spectral analysis of speech by linear prediction. *IEEE Trans. on Audio and Electroacoustics*, 21(3):140–148, 1973.

[44] K. D. Martin and Y. E. Kim. Musical instrument identification: A pattern-recognition approach. *The Journal of the Acoustical Society of America*, 104(3):1768–1768, 1998.

[45] P. Masri and A. Bateman. Improved modelling of attack transients in music analysis-resynthesis. In *Int. Computer Music Conf. (ICMC'96)*, page 100–103, 1996.

[46] O. Mayor, J. Llop, and E. Maestre. RepoVizz: a multimodal on-line database and browsing tool for music performance research. In *Int. Society for Music Information Retrieval Conf. (ISMIR'11)*, 2011.

[47] C. McKay and I. Fujinaga. jMIR: tools for automatic music classification. In *Int. Computer Music Conf. (ICMC'09)*, page 65–68, 2009.

[48] C. McKay, I. Fujinaga, and P. Depalle. jAudio: a feature extraction library. In *Int. Conf. on Music Information Retrieval (ISMIR'05)*, page 600–3, 2005.

[49] B. C. Moore and B. R. Glasberg. Suggested formulae for calculating auditory-filter bandwidths and excitation patterns. *The Journal of the Acoustical Society of America*, 74(3):750–753, 1983.

[50] K. R. Page, B. Fields, D. De Roure, T. Crawford, and J. S. Downie. Reuse, remix, repeat: the workflows of MIR. In *Int. Society for Music Information Retrieval Conf. (ISMIR'12)*, 2012.

[51] G. Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. *CUIDADO Project Report*, 2004. Available online: http://recherche.ircam.fr/equipes/analyse-synthese/peeters/ARTICLES/.

[52] R. Plomp and W. J. M. Levelt. Tonal consonance and critical bandwidth. *The Journal of the Acoustical Society of America*, 38(4):548–560, 1965.

[53] G. Roma, J. Janer, S. Kersten, M. Schirosa, P. Herrera, and X. Serra. Ecological acoustics perspective for content-based retrieval of environmental sounds. *EURASIP Journal on Audio, Speech, and Music Processing*, 2010.

[54] A. Röbel and X. Rodet. Efficient spectral envelope estimation and its application to pitch shifting and envelope preservation. In *Int. Conf. on Digital Audio Effects (DAFx'05)*, 2005.

[55] J. Salamon, S. Gulati, and X. Serra. A multipitch approach to tonic identification in indian classical music. In *Int. Society for Music Information Retrieval Conf. (ISMIR'12)*, 2012.

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

[57] J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Trans. on Audio, Speech, and Language Processing*, 16(6):1138–1151, 2008.

[58] Y. Shao, Z. Jin, D. Wang, and S. Srinivasan. An auditory-based feature for robust speech recognition. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'09)*, pages 4625–4628, 2009.

[59] E. Skovenborg and S. H. Nielsen. Evaluation of different loudness models with music and speech material. In *The 117th AES Convention*, 2004.

[60] M. Slaney. Auditory toolbox. *Interval Research Corporation, Technical Report*, 10, 1998. Available online: http://www.tka4.org/materials/lib/Articles-Books/Speech%20Recognition/AuditoryToolboxTechReport.pdf.

[61] M. Sordo. *Semantic Annotation of Music Collections: A Computational Approach*. PhD thesis, UPF, Barcelona, Spain, 2012.

[62] M. Sordo, O. Celma, and D. Bogdanov. MIREX 2011: Audio tag classification using weighted-vote nearest neighbor classification. In *Music Information Retrieval Evaluation Exchange (MIREX'11)*, 2011.

[63] M. Sordo, G. K. Koduri, S. Şentürk, S. Gulati, and X. Serra. A musically aware system for browsing and interacting with audio music collections. In *The 2nd CompMusic Workshop*, 2012.

[64] G. A. Soulodre. Evaluation of objective loudness meters. In *The 116th AES Convention*, 2004.

[65] S. S. Stevens. *Psychophysics*. Transaction Publishers, 1975.

[66] S. Streich. *Music complexity: a multi-faceted description of audio content*. PhD thesis, UPF, Barcelona, Spain, 2007.

[67] S. Streich and P. Herrera. Detrended fluctuation analysis of music signals: Danceability estimation and further semantic characterization. In *The 118th AES Convention*, 2005.

[68] G. Tzanetakis and P. Cook. Multifeature audio segmentation for browsing and annotation. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'99)*, page 103–106, 1999.

[69] G. Tzanetakis and P. Cook. Marsyas: A framework for audio analysis. *Organised sound*, 4(3):169–175, 2000.

[70] E. Vickers. Automatic long-term loudness and dynamics matching. In *The 111th AES Convention*, 2001.

[71] N. Wack, E. Guaus, C. Laurier, R. Marxer, D. Bogdanov, J. Serrà, and P. Herrera. Music type groupers (MTG): generic music classification algorithms. In *Music Information Retrieval Evaluation Exchange (MIREX'09)*, 2009.

[72] N. Wack, C. Laurier, O. Meyers, R. Marxer, D. Bogdanov, J. Serra, E. Gomez, and P. Herrera. Music classification using high-level models. In *Music Information Retrieval Evaluation Exchange (MIREX'10)*, 2010.

[73] Y. Yang, D. Bogdanov, P. Herrera, and M. Sordo. Music retagging using label propagation and robust principal component analysis. In *Int. World Wide Web Conf. (WWW'12). Int. Workshop on Advances in Music Information Research (AdMIRe'12)*, 2012.

[74] J. Zapata, M. Davies, and E. Gómez. MIREX 2012: Multi feature beat tracker (ZDG1 and ZDG2). In *Music Information Retrieval Evaluation Exchange (MIREX'12)*, 2012.

[75] J. R. Zapata, A. Holzapfel, M. E. Davies, J. L. Oliveira, and F. Gouyon. Assigning a confidence threshold on automatic beat annotation in large datasets. In *Int. Society for Music Information Retrieval Conf. (ISMIR'12)*, 2012.

MOTIF SPOTTING IN AN *ALAPANA* IN CARNATIC MUSIC

Vignesh Ishwar
Dept. of
Computer Sci. & Engg.
IIT Madras, India

Shrey Dutta
Dept. of
Computer Sci. & Engg.
IIT Madras, India

Ashwin Bellur
Dept. of
Electrical Engg.
IIT Madras, India

Hema A Murthy
Dept. of
Computer Sci. & Engg.
IIT Madras, India

ishwar@cse.iitm.ac.in shrey@cse.iitm.ac.in ashwin@ee.iitm.ac.in hema@cse.iitm.ac.in

ABSTRACT

This work addresses the problem of melodic motif spotting, given a query, in Carnatic music. Melody in Carnatic music is based on the concept of *raga*. Melodic motifs are signature phrases which give a *raga* its identity. They are also the fundamental units that enable extempore elaborations of a *raga*. In this paper, an attempt is made to spot typical melodic motifs of a *raga* queried in a musical piece using a two pass dynamic programming approach, with pitch as the basic feature. In the first pass, the rough longest common subsequence (RLCS) matching is performed between the saddle points of the pitch contours of the reference motif and the musical piece. These saddle points corresponding to quasi-stationary points of the motifs, are relevant entities of the *raga*. Multiple sequences are identified in this step, not all of which correspond to the the motif that is queried. To reduce the false alarms, in the second pass a fine search using RLCS is performed between the continuous pitch contours of the reference motif and the subsequences obtained in the first pass. The proposed methodology is validated by testing on Alapanas of 20 different musicians.

1. INTRODUCTION

Carnatic music is a sub-genre of Indian classical music prominent in south India. It is a heterophonic musical form which involves multiple instruments performing at the same time along with the voice. This form of music is highly melody centric and thrives on the melodic concept of *ragas*. The *ragas* in Carnatic music consist of a set of inflected musical notes called *svaras*. *Svaras* are the Indian classical equivalent to the solfege and are annotated as *Sa Ri Ga Ma Pa Da Ni Sa*. Theoretically frequencies of *svaras* follow more or less the just intonation ratios unlike the notes in western classical music which follow the equi-temperament scale [16]. In Carnatic music, the *svaras* are seldom rendered as discrete notes. They are rendered as a seamless meandering across the notes using *gamakas* [1] [7, 9, 17]. Figure 1 shows a pitch histogram of two *ragas* namely, *Kalyani* and *Sankarabharana*. Observe that there is a significant band of frequencies around every peak that are also frequent. This band of frequencies correspond to a single *svara* and is referred to as the intonation of that *svara*. This characteristic is observed due to the inflected nature of the *svaras*. It has been conjectured by musicians that a quantization of the pitch on the basis of the absolute frequencies of the *svaras* is erroneous [7].

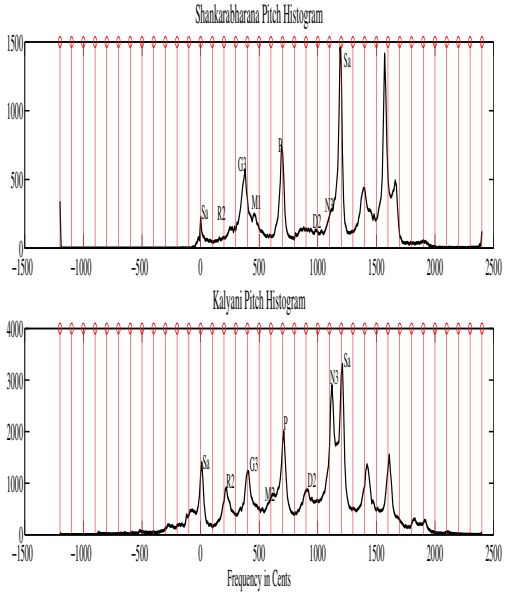


Figure 1. Pitch Histograms of Ragas Kalyani and Shankarabharana

A *raga* in Carnatic music can be characterised by a set of distinctive motifs. Distinctive motifs can be characterised by the trajectory of inflected *svaras* over time. These motifs are of utmost aesthetic importance to the *raga*. Carnatic music is a genre abundant with compositions. These compositions are replete with many distinctive motifs. These motifs are used as building blocks for extempore improvisational pieces in Carnatic music. These motifs can also be used for distinguishing between two *ragas*, and also for archival and learning purposes. The objective of this paper is to spot the location of the

[1] *Gamaka* is a meandering of a note encompassing other permissible frequencies around it.

distinctive motifs in an extempore enunciation of a *raga* called the *Alapana*. For more details on the concepts of *svara, gamaka, phraseology* pertaining to Carnatic music, the reader is advised to refer to [7].

In this paper, pitch is used as the main feature for the task of motif spotting. Substantial research exists on analysing different aspects of Carnatic music computationally, using pitch as a feature. Krishnaswamy et al [8], characterize and analyse *gamakas* using pitch contours. Serra et al [16] study the tuning of Indian classical music using pitch histograms. M. Subramaniam [17] has extensively studied the motifs in the raga Thodi using pitch histograms and pitch contours. All of the above prove the relevance and importance of pitch as a feature for computational analysis of Carnatic music.

In the previous work, the uniqueness of the characteristic motifs was established using a closed set motif recognition experiment using HMMs. As a continuation, in this work an attempt is made to spot motifs given a long *Alapana* interspersed with motifs. Time series motif recognition has been attempted for Hindustani music. J.C. Ross et. al. [13] use the onset point of the rhythmic cycle [2] emphasized by the beat of the tabla (an Indian percussion instrument) as a cue for potential motif regions. In another work, J.C.Ross et. al. [14] attempt motif spotting in a *Bandish* (a type of composition in Hindustani music) using elongated notes(*nyaas svara*).

Spotting motifs in a *raga alapana* is equivalent to finding a subsequence in trajectory space. Interestingly, the duration of these motifs may vary, but the relative duration of the *svaras* is preserved across the motif. The attempt in this work is to use pitch contours as a time series and employ time series pattern capturing techniques to identify the motif. The techniques are customized using the properties of the music. There has been work done on time series motif recognition in fields other than music. Pranav et. al. in their work [11] define a time series motif and attempt motif discovery using the EMMA algorithm. In [3, 18], time series motifs are discovered adapting the random projection algorithm by Buhler and Tompa to time series data. In [2], a new warping distance called Spatial Assembling distance is defined and used for pattern matching in streaming data. In the work of Hwei-Jen Lin et. al. [10], music matching is attempted using a variant of the Longest Common Subsequence (LCS) algorithm called Rough Longest Common Subsequence. This paper attempts similar time series motif matching for Carnatic Music. Searching for a 2-3 second motif (in terms of a pitch contour) in a 10 min *Alapana* (also represented as a pitch contour) can be erroneous, owing pitch estimation errors. To address this issue, the pitch contour of the *Alapana* is first quantized to a sequence of quasistationary points which are meaningful in the context of a *raga*. A two-pass search is performed to determine the location of the motif. In the first pass, a Rough Longest Common Subsequence approach is used to find the region corresponding to the location of the motif. Once the region is located, another pass is made using a fine-grained RLCS algorithm using the raw pitch contour.

The paper is organised as follows. Section 2 discusses the approach employed to extract the stationary points in a *raga*. Section 3 discusses the algorithm to perform the two-level RLCS approach to spot the motif. In Section 3.1 the Rough Longest Common Subsequence(RLCS) algorithm is discussed. In Section 4, the database used in the study is discussed. Section 5 discusses the results. Finally, conclusions are presented in Section 6.

2. SADDLE POINTS

2.1 Saddle Points: Reducing the Search Space

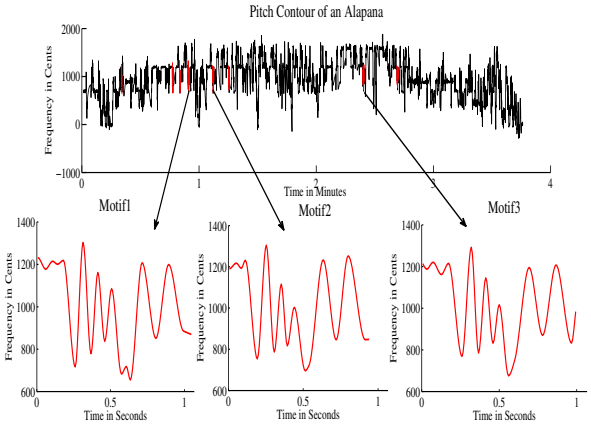


Figure 2. a) Motifs Interspersed in an Alapana ; b) Magnified Motif

The task of this paper is to attempt automatic spotting of a motif that is queried. The motif is queried against a set of *Alapanas* of a particular *raga* to obtain locations of the occurrences of the motif. The task is non-trivial since in *Alapanas*, rhythm is not maintained by a percussion instrument. Figure 2 (a) shows repetitive occurrences of motifs in a piece of music. An enlarged view of the motif is given in Figure 2(b). Since the *Alapana* is much longer than the motif, searching for a motif in an *Alapana* is like searching for a needle in a haystack. After an analysis of the pitch contours and discussions with professional musicians, it was conjectured that the pitch contour can be quantized at saddle points. Figure 3 shows an example phrase of the *raga* Kamboji with the saddle points highlighted.

Musically, the saddle points are a measure of the extent to which a particular *svara* is intoned. In Carnatic music since *svaras* are rendered with *gamakas*, there is a difference between the notation and the actual rendition of the phrase. However, there is a one to one correspondence with the saddle point frequencies and what is actually rendered by the musician (Figure 3). Figure 4 shows the pitch histogram and the saddle point histogram of an *Alapana* of the *raga Kamboji*. The similarity between the two pitch histograms indicates our conjecture that saddle points are important.

[2] The rhythmic cycle in Indian Music is called Tala and the onset point is called the Sam.

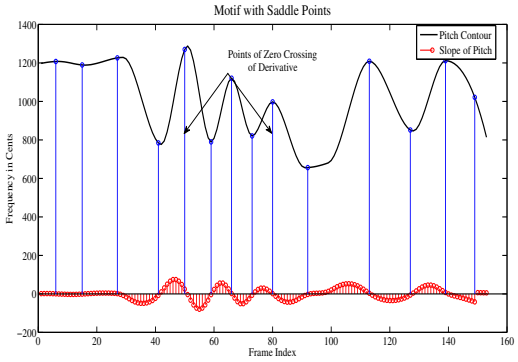


Figure 3. A Phrase with Saddle Points

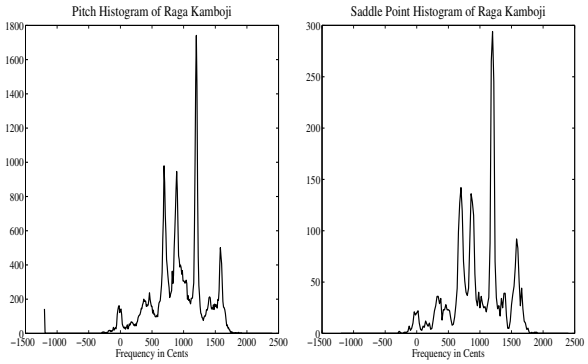


Figure 4. The Pitch and Saddle Point Histograms of the raga Kamboji

2.2 Method of obtaining Saddle Points

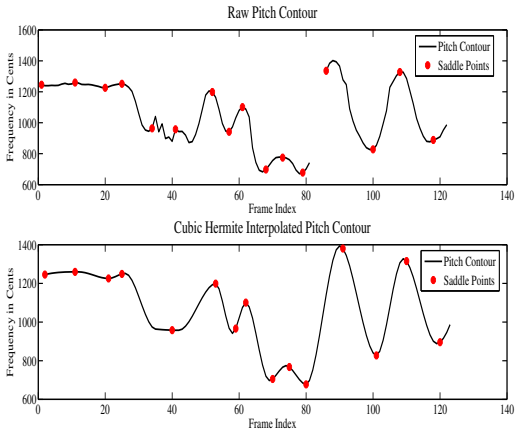


Figure 5. Distorted and Cubic Interpolated pitch contours

Carnatic music is a heterophonic musical form. In a Carnatic music concert, a minimum of two accompanying instruments play simultaneously along with the lead artist. These are the violin and the *mridangam* (a percussion instrument in Carnatic music). Carnatic music is performed at a constant tonic [1] to which all instruments are tuned. This tonic is chosen by the lead artist and is provided by an instrument called the *Tambura*. Thus, the simultaneous performance of many instruments in addition to the voice renders pitch extraction of the predominant voice a tough task. This leads to octave errors and other erroneous pitch values. For this task it is necessary that pitch be continuous. After experimenting with various pitch algorithms, it was observed that the Melodia-Pitch Extraction algorithm [15] produced the fewest errors. This was verified after re-synthesis using the pitch contours. In case of an octave error or any other such pitch related anomaly, the algorithm replaces the erroneous pitch values with zeros. The saddle points are obtained by processing the pitch contour extracted from the waveform. The pitch extracted is converted to the cent scale using Equation 1 to normalise with respect to the tonic of different musicians.

$$centFrequency = 1200 \cdot \log_2 \left(\frac{f}{tonic} \right) \qquad (1)$$

Least squares fit (LSF) [12] was used to compute the slope of the pitch extracted. The zero crossings of the slope correspond to the saddle points (Figure 3). A Cubic Hermite interpolation [4] was then performed with the initial estimation of saddle points to get a continuous curve (Figure 5). The saddle points are then obtained by sampling the interpolated spline using LSF. The interpolated pitch contours were validated by re-synthesizing and listening tests [3] .

3. A TWO PASS DYNAMIC PROGRAMMING SEARCH

In Section 2 it is illustrated that the sequence of saddle points are crucial for a motif. Therefore, RLCS is used to query for the saddle points of the given motif in the *Alapana*.

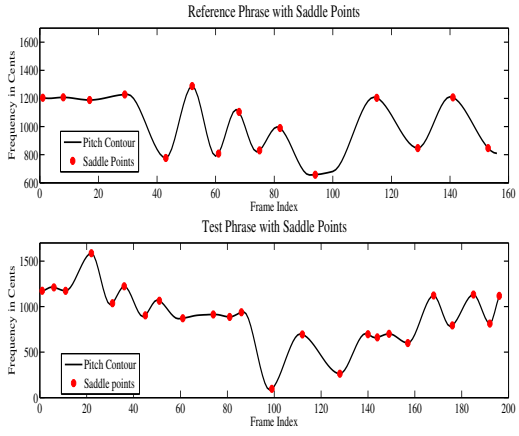


Figure 6. Similar Saddle Points, different contours

Music matching using LCS methods for western music is performed on symbolic music data [5]. The musical

[3] Original and re-synthesized waveforms are available at http://lantana.tenet.res.in/motif_analysis.html

notes in this context are the symbols. However, in the context of Carnatic music, a one to one correspondence between the notation and sung melody does not exist. Hence, in this paper, saddle points are used instead of a symbolic notation. One must keep in mind that saddle points are not symbols but are continuous pitch values (Figure 6). In order to match such pitch values, a rough match instead of an exact match is required. A variant of the LCS known as the Rough Longest Common Subsequence [10] allows such a rough match.

In this paper, a two pass RLCS matching is performed. In the first pass, the saddle points of the reference and query are matched to obtain the candidate motif regions. Nevertheless, given two saddle points, the pitch contour between two saddle points can be significantly different for different phrases (Figure 6). This leads to many false alarms. A second pass of RLCS is then performed on the regions obtained from the first pass to filter out the false alarms from the true motifs.

3.1 Algorithm for the Rough Longest Common Subsequence

RLCS is a variant of LCS which performs an approximate matching between a query and a reference retaining local similarity. Here, a cubic distance measure is used to determine the similarity between two saddle points. If the distance between the two saddle points in the query and reference is less than a threshold, *Td*, they are said to be roughly similar. In the RLCS algorithm, unlike LCS, the similarity measure (cost) is incremented by a weighted quantity (between 0 and 1) depending on the proximity of the points. To account for the local similarity, the width across query (WAQ) and width across reference (WAR) are incorporated. WAR and WAQ are the lengths of the shortest sub-strings, in the reference and query respectively, which contain the longest common subsequence. These measures convey the density of the match given a query and reference. Thus, lesser the WAQ and WAR, denser is the distribution of the RLCS and better is the alignment. The RLCS algorithm gives five matrices, namely, cost matrix, WAQ matrix, WAR matrix, score matrix and direction matrix (for tracing back the common subsequences).

3.2 First Pass: Determining Candidate Motif Regions using RLCS

The RLCS algorithm used in this paper is illustrated in this section. The *Alapana* is first windowed and then processed with the RLCS algorithm. The window size chosen for this task is 1.5 times the length of the motif queried for. The matrices obtained from the RLCS are then processed as follows:

- From the cells of the score matrix with values greater than a threshold, *seqFilterTd*, sequences are obtained by tracing the direction matrix backwards.
- The duplicate sequences which may be acquired are neglected, preserving unique sequences of length greater than a percentage, ρ, of the length of reference. These are then added to a sequence buffer.
- This process is repeated for every window. The window is shifted by a hop of one saddle point.
- The sequences obtained thus are grouped.
- Each group, taken from the first element of the first member to the last element of the last member, represents a potential motif region.

3.3 Second Pass: Determining Motifs from the Groups

In the first pass a matching of only the saddle points is performed. As mentioned above, eventhough the saddle points are matched it is not necessary that the trajectory between them match. This leads to a large number of false alarms. Now that the search space is reduced, the RLCS is performed between the entire pitch contour of the potential motif region obtained in the first pass and the motif queried. The entire pitch contour is used in order to account for the trajectory information contained in the phrases (Figure 7). The threshold *Td* used for the first pass is tightened in this iteration for better precision while matching the entire feature vector. In this iteration, the cell of the score matrix having the maximum value is chosen and the sequence is traced back using the direction matrix from this cell. This sequence is hypothesized to be the motif. The database and experimentation are detailed in the following sections.

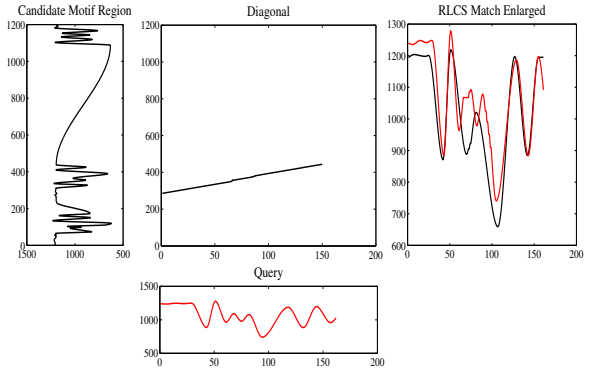


Figure 7. RLCS Matching

4. DATABASE

Table 1 gives the details of the database used in this work. As mentioned above, this task will be performed on *Alapanas*. The reason for using only *Alapana*s and not compositions for spotting is due to the limitations of the pitch extraction algorithms. The presence of multiple instruments and percussion along with the voice makes pitch extraction a non trivial task due to which pitch obtained for compositions is distorted. Pitch extracted for *Alapanas* have less distortion as compared to that of compositions.

Table 1. Database of *Alapanas*; N-Al- No. of *Alapanas*, N-Art- No. of Artists, Avg-Dur- Average Duration, Tot-Dur -Total Duration

Rāga Name	N-Al	N-Art	Avg-Dur (mins)	Tot-Dur (mins)
Kamboji	27	12	9.73	262.91
Bhairavi	21	15	10.30	216.49

Table 2. Phrases Queried

Raga Name	Phrase Notation	Average Duration(seconds)
Kamboji	S..N2 D2 P D2...	1.8837
Bhairavi	R2 G2 M1 P D1 P..	1.3213

The details of motifs of the ragas queried are given in Table 2. The average duration is obtained from the ground truth labeled in the previous work [6].

5. EXPERIMENTS AND RESULTS

RLCS was performed on the database *Alapana*s querying for the motifs of the ragas mentioned in Table 1. The distance function used for RLCS is cubic in nature with the equation given below.

$$\delta(i,j) = \begin{cases} \frac{|i-j|^3}{300^3} & ;if\ |i-j| \leq 300 \\ 1 & ;\ otherwise \end{cases} \quad (2)$$

Here, i and j correspond to the i^{th} saddle point in the *alapana* that matches the j^{th} saddle point in the motif. Due to different styles of various musicians, an exact match between the saddle points of the motif, and the *Alapana* cannot be expected. Hence in this paper a leeway of *extrTd = 300 cents* is allowed between two saddle points. Musically two points 300 cents (3 semitones) apart cannot be called similar, but in this case, due to the different styles of singing and artefacts introduced due to pitch extraction, the saddle points for the same phrase sung by different artists does not match exactly but is generally within a limited range of frequencies. The upper limit of this range is set to 300 cents empirically.

In this work, the phrases sung across octaves are ignored. For this experiment the parameters set were as follows: *Td* = 0.45 ; ρ = 0.8; *seqFilterTd* = 0.45. The parameter $0 < \rho < 1$ is a user defined parameter that ensures that $\rho\times$ *length of the query motif* is matched with that of the *Alapana*. As a sanity check, the regions obtained from the RLCS were verified with the ground truth motifs labelled for the experiment in the previous work [6]. The parameters were tuned to retrieve as many labelled ground truth motifs as possible for the *raga Kamboji*'s phrase. A high percentage of regions coinciding with the ground truth labelled, were retrieved by the RLCS [4] .

The regions obtained from the RLCS were then subjected to a listening test performed by three professional musicians. The regions which contained the phrase queried were marked as true, and those which did not as false. It is observed that the correlation with respect to the carefully marked ground truth by one musician with verification by two other musicians is an average of 0.833. The details of the number of ground truth motifs retrieved and the total number of trues retrieved after verification by the listening test are given in Table 3. The number of false positives, retrieved are however substantial. This is affordable since the objective in the first pass is to obtain the maximum number of the regions similar to the motif. The second iteration of RLCS is performed to filter out the false positives. Now

Table 3. Retrieved regions First Pass: TR-Total Retrieved, LGT-Labelled Ground Truth, TrR-True Retrieved, PR-Percentage Retrieved

Rāga Name	No of Apalanas	TR	LGT	TrR	PR
Kamboji	27	719	70	58	82.86%
Bhairavi	20	474	103	91	88.35%

that the candidate motif regions are known, the second pass of RLCS is conducted wherein the same motif from four different artists are queried in the regions retrieved by the first pass. The entire pitch contour of the query and reference are used for this task in order to account for the information of trajectory of pitches between the saddle points. Since the task is to locate the motif in a smaller continuous search space, the threshold *extrTd* was tightened and the allowable leeway for the distance function was reduced to 200 cents (See Eq 2). The parameter *seqFilterTd* is not used in the second pass since the best match with the candidate region is sought (See Section 3.2). The RLCS is repeated for four examples of the same motif by different musicians and the scores are obtained.

5.1 Evaluation and Discussion

This work illustrates the method of spotting a query motif in an improvisational form of Carnatic music called an *alapana*. The requirement of a query motif for such a search is due to the scarce rendition of certain characteristic phrases in an *alapana*. The spotting of such phrases proves to be useful to musicians and students for analysis purposes. To quantify this and evaluate the performance of the algorithm in this context, it was decided to compute the recall, precision and F2-Measure for the motifs retrieved. The F2-Measure was chosen in order to give a higher weightage to the recall. The objective of this work being music exploration through motif spotting, the recall of the motif queried is of greater importance.

The hits obtained in the second pass are sorted according to the RLCS scores. The precision, recall and F2-Measure per *Alapana* are calculated and the average is computed across all *Alapanas*. The motifs are not exact since they correspond to the extempore enunciation by an artist. The results are reported per query, per *alapana*. The hits of the RLCS in an *alapana* are sorted according to their scores and the precision, recall and F2-Measure are calculated for the top 10 sorted hits. The relevant motifs

[4] In the previous experiment not all occurrences of the phrase were labelled. The instrumental phrases were ignored and some were missed due to manual error.

are all the motifs which were marked as true in that *alapana*. The results are illustrated in Table 4. The experiments on the phrase of the *raga Kamboji* were treated as the development set, with parameters optimised to maximise the match. To verify that this works in general, another *raga Bhairavi* was taken up for study. The same parameters used for the *raga Kamboji* were used for the *raga Bhairavi*. The results are described in the Tables 3 and 4. From the results obtained above, it is clear that even-

Table 4. Results: Average Precision - Pr%, Recall - Rec%, F2-Measure - F2M% across *alapanas*, Average of Query Scores - AQS

Queries	*Kamboji*-27 *Alapanas*			*Bhairavi*-20 *Alapanas*		
	Pr	Rec	F2M	Pr	Rec	F2M
Query 1	43.18	81.52	64.19	43.13	94.80	71.94
Query 2	33.33	63.318	50.50	38.15	85.10	64.33
Query 3	43.63	83.16	65.28	36.87	81.04	61.04
Query 4	25.90	58.62	42.85	38.12	83.33	63.07
AQS	40.45	76.00	60.00	41.25	91.04	68.91

though the precision is low, the recall is high in most of the cases. Certain partial matches are also obtained where either the first part of the query is matched or the end of the query is matched. These are movements similar to those of the phrases and are interesting for a listener, learner, or researcher. High scores were obtained for certain false alarms. This is primarily due to some significant similarity between the false alarm and the original phrase.

6. CONCLUSION

In this work, RLCS is used for motif discovery in *alapanas* in Carnatic music. It is illustrated that the saddle points of the pitch contour of a musical piece hold significant music information. It is then shown that quantizing the pitch contour of the *alapana* at the saddle points leads to no loss of information while it results in a significant reduction in the search space. The RLCS method is shown to give a high recall for the motif queried. Given that the objective is to explore the musical traits of a *raga* by spotting interesting melodic motifs rendered by various artists, the recall of the motif queried is of higher importance than the precision. The future work would involve spotting of motifs occurring across multiple octaves. It would also be interesting see if similarity measures can be obtained by this approach across ragas.

7. ACKNOWLEDGEMENTS

This research was partly funded by the European Research Council under the European Union's Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583).

8. REFERENCES

[1] Ashwin Bellur and Hema A Murthy. A cepstrum based approach for identifying tonic pitch in indian classical music. In *National Conference on Communications*, 2013.

[2] Yueguo Chen, Mario A. Nascimento, Beng Chin, Ooi Anthony, and K. H. Tung. Spade: On shape-based pattern detection in streaming time series. *in ICDE, 2007*, pages 786–795, 2007.

[3] Bill Chiu, Eamonn Keogh, and Stefano Lonardi. *Probabilistic discovery of time series motifs*. 2003.

[4] F. N. Fritsch and R. E. Carlson. *Monotone Piecewise Cubic Interpolation*. SIAM Journal on Numerical Analysis, Vol. 17, No. 2., 1980.

[5] F Scholer I S H Suyoto, A L Uitdenbogerd. Searching musical audio using symbolic queries audio, speech, and language processing. *IEEE Transactions on In Audio, Speech, and Language Processing, IEEE Transactions on, Vol. 16, No. 2.*, pages 372–381, 2008.

[6] Vignesh Ishwar, Ashwin Bellur, and Hema A Murthy. Motivic analysis and its relevance to raga identification in carnatic music. In *Workshop on Computer Music*, Instanbul, Turkey, July 2012.

[7] T M Krishna and Vignesh Ishwar. Svaras, gamaka, motif and raga identity. In *Workshop on Computer Music*, Instanbul, Turkey, July 2012.

[8] A Krishnaswamy. Application of pitch tracking to south indian classical music. *In Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 557–560, 2003.

[9] A Krishnaswamy. Inflexions and microtonality in south indian classical music. *Frontiers of Research on Speech and Music*, 2004.

[10] HWEI-JEN LIN, HUNG-HSUAN WU, and CHUN-WEI WANG. Music matching based on rough longest common subsequence. *Journal of Information Science and Engineering*, pages 27, 95–110., 2011.

[11] Pranav Patel, Eamonn Keogh, Jessica Lin, and Stefano Lonardi. Mining motifs in massive time series databases. *In Proceedings of IEEE International Conference on Data Mining (ICDM)*, pages 370–377, 2002.

[12] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing. Second Edition*. Oxford University Press, 1992.

[13] Joe Cheri Ross and Preeti Rao. Detecting melodic motifs from audio for hindustani classic music. In *ISMIR*, Portugal, October 2012.

[14] Joe Cheri Ross and Preeti Rao. Detection of raga-characteristic phrases from hindustani classical music audio. *Proc. of the 2nd CompMusic Workshop*, July 12-13, 2012.

[15] J. Salamon and E. Gomez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech and Language Processing*, pages 20(6):1759–1770, Aug. 2012.

[16] J Serra, G K Koduri, M Miron, and X Serra. Tuning of sung indian classical music. *In Proc. of ISMIR*, pages 157–162, 2011.

[17] M Subramanian. Carnatic ragam thodi - pitch analysis of notes and gamakams. *in Journal of the Sangeet Natak Akademi*, pages 3–28, 2007.

[18] Dragomir Yankov, Eamonn Keogh, Jose Medina, Bill Chiu, and Victor Zordan. Detecting time series motifs under uniform scaling. *In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, August 12-15, 2007*, 2007.

EMPIRICAL ANALYSIS OF TRACK SELECTION AND ORDERING IN ELECTRONIC DANCE MUSIC USING AUDIO FEATURE EXTRACTION

Thor Kell
IDMIL, CIRMMT, McGill University
thor.kell@mail.mcgill.ca

George Tzanetakis
University of Victoria
gtzan@cs.uvic.ca

ABSTRACT

Disc jockeys are in some ways the ultimate experts at selecting and playing recorded music for an audience, especially in the context of dance music. In this work, we empirically investigate factors affecting track selection and ordering using DJ-created mixes of electronic dance music. We use automatic content-based analysis and discuss the implications of our findings to playlist generation and ordering. Timbre appears to be an important factor when selecting tracks and ordering tracks, and track order itself matters, as shown by statistically significant differences in the transitions between the original order and a shuffled version. We also apply this analysis to ordering heuristics and suggest that the standard playlist generation model of returning tracks in order of decreasing similarity to the initial track may not be optimal, at least in the context of track ordering for electronic dance music.

1. INTRODUCTION

The invention of recording lead to the possibility of selecting recorded music to entertain a group of people. The idea of listening to records instead of listening to bands took off after the second world war, when sound systems and record players began to appear in night clubs and cafes in New York, Jamaica, London, Paris, and beyond [5].

Since then, the disk jockey (DJ) has evolved from a simple selector and orderer of music into a sophisticated performer with considerable skill and training. Although these performance aspects are compelling, the primary focus of this paper is the basic selection and ordering of music. DJs generally bring a limited amount of their music collection to any given gig, and play a reasonably large subset of it. Two important questions to consider are 'What tracks go into a playlist?', and 'What is the best ordering of these tracks?'. Track ordering is not a well understood process, even by DJs. Many DJs will say only that two tracks 'work' or 'do not work' together, and not be able to comment further. [5] We investigate this selection and ordering process in terms of the automatically computed similarity between tracks, in terms of features representing timbre, key, tempo and loudness. The source data for this investigation is the British Broadcasting Corporation's Essential Mix radio program [1] . Broadcast since 1993, the Essential Mix showcases exceptional DJs of various genres of electronic dance music (EDM), playing for one or two hours. It is considered one of the most reputable and influential radio programs in the world. By investigating the relationships between tracks in DJ sets we hope to better understand track selection by DJs and inform the design of algorithms and audio features for automatic playlisting.

The automatic estimation of music similarity between two tracks has been a primary focus of music information retrieval (MIR) research. [4] Several methods for computing music similarity have been proposed based on content-analysis, metadata (such as artist similarity, web reviews), and usage information (such as ratings and download patterns in peer to peer networks). [1] Music similarity is the basis of query-by-example which is a fundamental MIR task, and also one of the first tasks explored in MIR literature. In this paradigm the user submits a query consisting of one or more 'seed' pieces of music, sometimes also including metadata and user preferences. The system then responds by returning a playlist of music pieces ranked by their similarity to the query, and set in some order. In contrast, our approach is analytic. Rather than generating playlists, we investigate existing DJ sets through audio feature extraction and examine the transitions between tracks in terms of audio features representing timbre, loudness, tempo, and key. We also compare the results of our empirical investigation to common ordering methods, and offer some suggestions for improving current playlisting heuristics.

2. RELATED WORK

Early MIR work investigating the automatic calculation of music similarity and how to evaluate different approaches formulated a general methodology that is followed by the majority of existing work to this day. In this methodology, the primary goal is assessing the relative performance of different algorithms for computing music similarity by somehow evaluating the 'quality' of the generated playlists.

The most common approach of generating a playlist is to consider the N closest neighbors in terms of automatically calculated similarity to a particular query. Several

[1] http://www.bbc.co.uk/programmes/b006wkfp

automatic playlists are generated from seed queries representing the desired diversity of the music considered. This set of automatically generated playlists is then evaluated, typically using one or both of two approaches: objective evaluation using proxy ground truth for relevance, or subjective evaluation through user studies. The basic idea is to evaluate a playlist by considering it good if it contains a high number of 'relevant items' to the query [1]. The relevance ground truth can be provided by users in subjective evaluation but this is a time consuming and labor intensive process that does not scale well.

Objective evaluation has the advantage that it can scale to any number of queries and playlists, as long as the tracks have some associated meta-data that can be used as a proxy for relevance. Common examples of such proxy sources include artist, genre, and song [1, 10]. In addition to music similarity calculated based on audio content analysis, other sources of information such as web reviews, download patterns, ratings and explicit editorial artist similarity [7] can also be used for estimating music similarity [6]. Playlists themselves have also been used to calculate artist and track similarity based on co-occurrence. Sources of playlist data include the Art of the Mix, a website that contains a large number of hobbyist playlists [4], and listings of radio stations [12]. DJ sets remain an untapped resource, however.

The most common relevance-based evaluation measures (such as precision, recall and F-measure [2]) are borrowed from text information retrieval and only consider the items contained in the set of returned results, without taking into account their order. The paradigm of a single seed query song creating a list of N items ranked by similarity has remained a common approach to automatic playlisting and music recommendation. Some notable exceptions in terms of ordering include: heuristics about trajectory for the ordering of returned items [10], using song sets instead of single seeds [11], ordering based on the traveling salesman problem [16], and considering both a start track and an end track for the playlist [8]. The assumption of similarity has also been challenged by the finding that in many cases users prefer diverse playlists [17] as measured by automatic feature analysis. This is the closest work in terms of approach to the work described in this paper.

Another theme of more recent work has been providing more user control to the process of automatic playlist generation. If the tracks considered are associated with a rich set of attribute/value pairs then techniques from constraint satisfaction programming and inductive learning can be used to generate playlists that to some extent optimally satisfy the user preferences [15]. The ability to control what attributes are used for estimating music similarity has also been investigated [19]. One of the simplest forms of user control is skipping behavior, which has been used to iteratively improve playlist generation [14]. A more elaborate method for steerable playlist generation is based on tag clouds and and a music similarity space derived from radio station playlists [12].

Physiological data such as heart rate has also been investigated for playlist generation [13]. The novelty aspect of playlist generation by tracking user listening information has also been explored [9]. A different approach altogether is to create playlists visually, based on some graphical representation of the music collection [18]. In all of this literature, different approaches to playlist generation are also evaluated with a combination of objective measures and user studies, comparing different configurations to a random or a simple algorithmic baseline. For example, a recent study compared two recommender systems (based on artist similarity and acoustic content) with the Apple iTunes Genius recommender which is believed to be based on collaborative filtering [3].

3. MOTIVATION AND PROBLEM FORMULATION

The motivation behind our work is to investigate the process of playlist/mix creation by analyzing existing mixes created by experts - in this case, DJs. Existing work has mostly focused on more general playlists created by average listeners. Rather than relying on user surveys, we focus on empirical analysis based on audio feature extraction. This allows us to investigate what audio attributes DJs use when selecting and order their tracks. We further compare these attributes to collections of random EDM tracks, and to artist albums. We specifically investigate whether track order matters. We also examine important assumptions that are frequently made by automatic playlist generation systems. Specifically, we investigate if ordering based on similarity ranking is a good choice, and if so, in what manner.

In existing literature these assumptions are typically manifested in the design of an automatic playlist algorithm, and the results are evaluated through objective or subjective approaches. Issues such as the sameness problem in playlists formed from collections of music that do not have stylistic diversity, or the playlist drift problem in large diverse collections are also discussed but are not empirically supported [8]. In contrast, our approach is complimentary and attempts to test these assumptions directly on existing mixes. Our methodology can be also viewed as an empirical musicological approach to understanding how DJs select and order music.

4. METHODOLOGY

4.1 Data

We obtained 114 Essential Mix DJ sets from the archival website `themixingbowl.org` (**DJS**). These sets cover three years (2009-2011) of the radio show. In addition, a collection of 189 artist albums (**ALBS**) (from the author's own collection, covering a wide range of music and genres) was used for comparison purposes. Finally, 100 random EDM (**REDM**) track sets were created by randomly picking tracks from the collection of electronic dance music (covering 1,261 tracks) of one of the authors who frequently performs as a DJ. [2] In order to investigate track or-

[2] It is possible, but unlikely, that there is overlap between the collections. If there is overlap, our method of using transitions should make its

dering we created a random shuffle of each Essential Mix DJ set (**RDJS**).

The DJ sets consist of continuous audio, without timing information about the individual tracks. Each set was split into two-minute exceprts followed by two-minute gaps, giving 30 excerpts of audio per two hour set. This accounts for any mixing that the DJ might be doing, and factors in track length. We considered more elaborate manual approaches such as extracting each track from each set manually or selecting the most representative part of each track for processing. As this is a very time consuming process this would have severely limited the amount of audio data we would have been able to process.

The main issue with the two-minute exceprt approach is that it may miss vital sections, in terms of audio features, and include transitions between sections of the same track. By using the exact same approach for albums and for the randomized EDM sets we believe that whatever effects this arbitrary segmentation has will be approximately the same for all data sets and the relative comparisons we make are valid. It must be noted that the REDM set was not mixed, unlike the DJS and RDJS set. However, the audio was concatenated and the same 2-minute exceprt / 2-minute gaps methodology was applied to it. Thus, some exceprts will contain parts of two tracks, which we hope will approximate the impact of mixing.

4.2 Audio Feature Extraction

Our goal is to examine different factors affecting selection and ordering, based on automatic audio feature extraction. More specifically, we examine the effects of features representing timbre, key, loudness, and tempo. The audio features used for the evaluation were computed using two sources: the Echo Nest Analyze API [3] and Marsyas [4]. The Echo Nest Analyze API returns timbre data as a 12-dimensional vector, where each element matches a spectral characteristic: the first dimension is loudness, the second indicates a strong attack, and so on. Timbre data is given for each 'segment', which roughly corresponds to musical events detected by onsets (on average about half a second of audio). The mean of each dimension was taken to supply a single vector for each 2-minute excerpt of audio. The Analyze API returns loudness as decibels, and tempo as beats per minute. Key is returned as a tuple of pitch class and major or minor mode.

Marsyas returns a 63-dimensional vector for representing timbre. The features used are based on the Spectral Centroid, Roll-Off, Flux and Mel-Frequency Cepstral Coefficients (MFCC). To capture feature dynamics, a running mean and standard deviation over the past M frames of 23 milliseconds is computed. The features are computed at the same rate as the original feature vector but depend on the past M frames (e.g. M=40, corresponding approximately to a so-called 'texture window' of 1 second). This results in a feature vector of 32 dimensions at the same rate as the original 16-dimensional one. The sequence of feature vectors is collapsed into a single feature vector representing the entire audio clip by taking again the mean and standard deviation across the track (the sequence of dynamics features) resulting in the final 64-dimensional feature vector per 2-minute excerpt. For tempo, a method based on autocorrelation of an onset detection function and the creation of a beat histogram is utilized. There is no direct key estimation implemented in Marsyas.

4.3 Metrics

From this audio featured data, we characterize the transitions between successive feature vectors corresponding to 2-minute excerpts in order to examine track selection during the course of a set. Tempo and loudness were simply subtracted, and key was represented by the change in key signature. In order to characterize the transition of the timbre vectors, we considered both the L1 (Manhattan) distance and L2 (Euclidean) distance between successive vectors after each dimension was max/min normalized across the data set under consideration. The analysis conclusions were similar for the two distance metrics (L1 and L2). Due to space limitations, we only report numbers based on the L1 metric. Although more elaborate distance metrics between the timbre vectors can be devised, we prefer to a use a simple metric and normalize each feature dimension as this provides a consistent, easily repeatable method.

Another objective of our analysis is to investigate the importance of ordering in DJ sets assuming a fixed set of tracks (in this case, 2-minute excerpts) to be played. In order to characterize different orderings of a set of tracks, we use ideas from combinatorics and permutations. More specifically, we want to compare excerpt orders created using different heuristics to the excerpt order of the original DJ set. Therefore, we require some measure of similarity between different permutations.

We utilize the concept of inversions, which are a way of measuring the differences between an ordered list and a permutation of that ordered list. For example, given some ordered list (e.g. [A, B, C, D, E]) and a permutation of it (e.g. [B, C, A, E, D]), an inversion is a pair of positions where the entries are in the opposite order. The permutation in our example has three inversions: (0, 2) for the pair (B, A); (1, 2) for the pair (C, A), and (3, 4) for the pair (E, D). The number of inversions between two different orderd lists of the same items gives a measure of how similar they are in terms of ordering. We consider the original DJ set order as the original sequence and the sequences created by four different ordering heuristics as permutations.

Assuming a fixed set of excerpts corresponding to a particular DJ set, we consider the following heuristics: *Rank* - the excerpts are ranked by increasing distance to the initial excerpt of the original order; *NN* - each successive excerpt is the nearest neighbor of the previous one, without allowing repetitions; *Median* - the distances of all the clips to the initial excerpt are computed and the one that is closest to the median distance is selected as the next excerpt, without allowing repetitions; *Furthest Neighbor* - each successive

impact minimal.

[3] http://echonest.com

[4] http://marsyas.info

Table 1. Timbre transition statistics (EN: Echonest, MRS: Marsyas)

	Mean ± Std	Q1	Median	Q3
DJS-EN	1.09 ± 0.44	0.76	1.02	1.34
ALBS-EN	0.86 ± 0.4	0.58	0.80	1.05
REDM-EN	1.44 ± 0.57	1.03	1.39	1.80
RDJS-EN	1.20 ± 0.44	0.87	1.13	1.45
DJS-MRS	5.75 ± 2.02	4.35	5.41	6.82
ALBS-MRS	4.43 ± 1.91	3.11	4.11	5.31
REDM-MRS	6.85 ± 2.51	5.18	6.64	8.40
RDJS-MRS	6.24 ± 2.09	4.76	5.9	7.37

Table 2. Tempo transition statistics (EN: Echonest, MRS: Marsyas), Values in BPM.

	Mean ± Std	Q1	Median	Q3
DJS-EN	8.93 ± 23	0.03	0.12	1.51
ALBS-EN	17.82 ± 20.69	1.87	10	27.92
REDM-EN	6.53 ± 15.25	0.02	0.88	5.05
RDJS-EN	11.49 ± 24.07	0.06	1.04	7.88
DJS-MRS	3.86 ± 13.17	0	0	1
ALB-MRS	20.19 ± 22.62	1	11	35.5
REDM-MRS	7.41 ± 16.18	0	0.5	6
RDJS-MRS	6.6 ± 15.01	0	1	4

clip is the furthest neighbor of the previous one, without allowing repetitions. The *Rank* heuristic corresponds to the common scenario of ranked list retrieval.

5. DATA ANALYSIS

5.1 Transition Analysis

We characterize the distribution of transition values (L1 distances) for each configuration (DJ sets, artist albums, random EDM and random shuffle DJ sets) by computing statistics (mean, standard deviation, median and quartiles) for each different factor (timbre, tempo, loudness, key). In order to characterize statistical significance we use the Welch *t-test* which is appropriate for samples that may have unequal variance. All differences reported here are strongly statistically significant, with $p < 0.0001$.

The numbers for timbral transitions in Table 1 [5] can be used to support various assumptions that are commonly made in automatic playlist generation systems. For example, the relations between DJS and REDM show that there is more to DJ selection than just randomly selecting tracks of EDM. The average timbral transition between exceprts for the DJ sets is smaller than the timbral transition between exceprts of random EDM tracks. This implies that that DJs try to pick tracks that are similar in timbre, and order them in ways that further minimize the timbral differences. The timbral transitions between excerpts of albums are the smallest. This is reasonable, as albums tend to be sonically coherent, featuring the same instruments and orchestration throughout. Figure 1 shows timbre transition data for single examples of each category. Furthermore, by comparing DJS and RDJS it can be seen that track ordering is important: the original ordering results in smaller transitions on average than a random shuffle of the same clips. These findings are supported by both the Echo Nest (EN) and Marsyas (MRS) feature extraction, increasing our confidence in their validity.

Another factor to examine is tempo. As can be observed from Table 2, DJ sets have the least amount of tempo change. Tempo is something that can be (and is almost always) controlled by the DJ. Therefore it is not surprising that small tempo transitions are observed. As expected, due to the large variety of genres and the unreliability of tempo detection for some genres, the ALBS dataset shows the highest tempo transitions. The effect of ordering is less pronounced than in the case of timbre, as can be observed by comparing DJS and RDJS, but is still there. Somewhat surprisingly, the random selection of EDM tracks also exhibits small tempo transitions, probably due to the consistent use of a small range of tempi in this style of music (House music ranges from 110 BPM to 130 BPM, with a large peak around 120 BPM, for example). However, note the increase in the median tempo transition from the DJS dataset to RDJS and REDM: the tempo transitions for randomized DJ sets are similar to those of randomized EDM tracks. When a DJ takes control of the tempo, the median transition drops significantly.

Table 3. Loudness transition statistics. Values in dB.

Loudness	Mean ± Std	Q1	Median	Q3
DJS	0.76 ± 0.71	0.28	0.59	1.06
ALBS	3.39 ± 4.13	0.92	2.16	4.32
REDM	3.18 ± 3.07	1.02	2.26	4.37
RDJS	0.88 ± 0.85	0.31	0.67	1.19

Loudness can easily be (and is almost always) controlled by the DJ. We expect that DJ sets will be relatively homogeneous. This is clearly shown by examining the data in Table 3, and contrasting DJS with REDM. Ordering also has a small effect, as can be seen by examining the relation between DJS and RDJS. Thus, DJs appear to vary volume slightly over the course of a set. We also examined key using the Echo Nest's analysis, but did not find any stastical significance in the differences observed. We can thus suggest that DJs do not use key as a primary concern when selecting and ordering their tracks - unlike classical musicians, for whom key and harmony are paramount.

5.2 Analysis of Ordering Heuristics

In addition to transition analysis, we also examined different heuristics for ordering playlists and compared them to the 'golden' order of the original DJ set. Table 4 shows the

[5] The numbers reported here are the L1 distance between each successive timbre vector, as described in section 4.3

Table 4. Average number of inversions per heuristic

Method	**Echo Nest**	...	**Marsyas**	...
	HEUR	RND	HEUR	RND
Ranked	186.27	201.10	183.71	192.28
NN	187.64	202.5	186.77	189.43
Median	171.85	203.86	175.89	172.59
FN	192.71	202.00	191.33	189.35
Equal-Step	172.75	199.50	170.16	174.71

results of this comparison using the number of inversions as an estimate of how 'close' two orderings are. Based on the above transition analysis, timbre is the dominant factor affecting playlist selection and ordering. Thus, it is the parameter used in Table 4. For each heuristic we report the average number of inversions, across all sets in DJS, comparing the original order of the tracks with the heuristic order (HEUR) and as a baseline the number of inversions comparing the original order of the trakcs with the order of a random shuffle (RND). From Table 4 it can be seen that all ordering heuristics come closer to the original ordering than to a random shuffle, indicating that they are reasonable choices.

One of the most interesting findings is that the traditional ordering based on ranked similarity is not the best heuristic. Both the *Median* and *Equal-step* heuristics appear to be closer to the original set order. This implies that consistent transitions are more important than tracks near the start of the mix being similar to the initial track. Figure 2 shows the timbral transitions for a specific DJ set as well as two orderings. As can be seen in the middle subfigure the *Rank* heuristic results in playlist drift near the end while the *Median* heuristic provides more balanced transitions.

6. DISCUSSION AND FUTURE WORK

We have proposed and demonstrated the use of automatic audio feature extraction to examine the selection and order of tracks in DJ mixes. Our approach is distinct and complimentary to the traditional approach of generating playlists automatically and evaluating them through proxy ground truth and user studies. It is an analytic approach that uses the playlists as data to be analyzed. We also specifically focus on DJ selection of EDM tracks rather than music in general.

Our transition analysis has shown timbre to be an important attribute used by DJs when selecting and ordering tracks. DJ mixes are more timbrally similar than random EDM tracks, though not as timbrally similar as artist albums. Tempo and loudness tend to be controlled by the DJ, and this is also reflected in our findings. Our results support the intuitive idea that DJs tend to play tracks that broadly 'sound the same', and fits with the typical statement by DJs that two tracks 'work' together, although there are probably many more subtle factors involved in DJ track selection and ordering. The results also support the emphasis on timbral similarity that is common in automatic playlist generation systems. Our findings are consistent between the two different audio feature front-ends and confirm design choices that have been made in music recommendation and automatic playlist generation systems.

The order of the selected tracks matters, as shown by statistically significant differences in the transitions between the original order and a shuffled version. These ordering differences were found in all factors considered except key. The investigation of ordering heuristics implies that the standard playlist generation model of returning tracks in order of decreasing similarity to the initial track may not be optimal (at least in the context of DJ ordering for EDM). Returning results ranked by similarity may not be optimal, and transitions of roughly equal size are probably a better choice for automatic playlist generation algorithms.

Future work includes the analysis of more data, such as the total 900 Essential Mixes rather than 114 mixes considered here. Commercial mixes are also a possibility, as are DJ mixes from the wider internet. We would also like to follow a similar approach to the analysis of playlists across a variety of genres, as well as playlists created by everyday listeners and music recommendation systems. In terms of informing automatic playlist generation algorithms, the most promising direction is to investigate the effectiveness of ordering heuristics that emphasize smoothness of transitions rather than absolute ranking.

Track selection and ordering is a tricky process that is not totally understood even by DJs themselves: It is hoped that this paper has shed some light on the role that timbre, key, volume and tempo play in this process. We hope that our work informs future work in automatic playlist generation and music recommendation, and that the proposed methodology inspires more empirical musicological analysis of how DJs select and order tracks.

7. REFERENCES

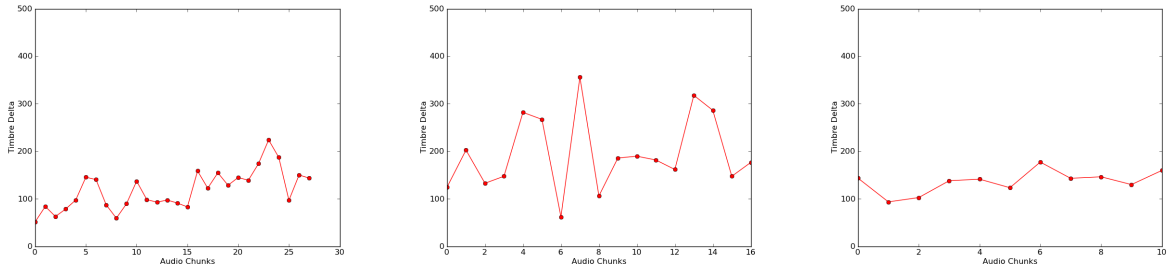
[1] J.J. Aucouturier and F. Pachet. Music similarity measures: Whats the use. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2002.

[2] J.J. Aucouturier and F. Pachet. Tools and architecture for the evaluation of similarity measures: case study of timbre similarity. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2004.

[3] L. Barrington, R. Oda, and G. Lanckriet. Smarter than genius? human evaluation of music recommender systems. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2009.

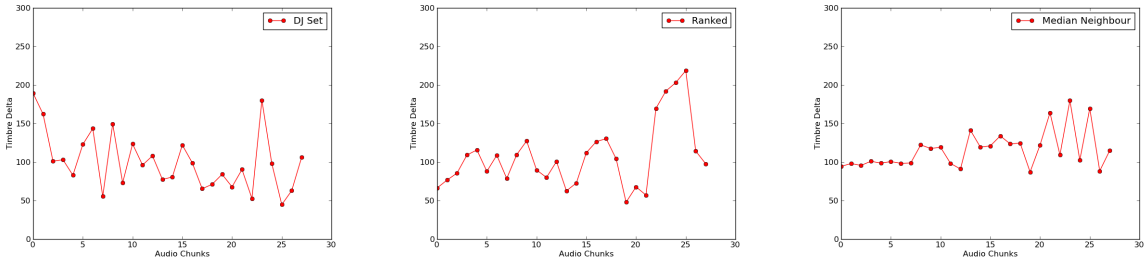
[4] A. Berenzweig, B. Logan, D.P.W. Ellis, and B. Whitman. A large-scale evaluation of acoustic and subjective music-similarity measures. *Computer Music Journal*, 28(2):63–76, 2004.

[5] B. Brewster and F. Broughton. *Last Night a DJ Saved My Life: The History of the Disc Jockey*. Grove Press, 2000.



(a) DJ set (b) Random Singles (c) Artist Album

Figure 1. Change in timbre over time, for examples of each dataset



(a) Original DJ set (b) Rank heuristic ordering (c) Median heuristic ordering

Figure 2. Specific example of transitions for different orderings

[6] D.P.W. Ellis, B. Whitman, A. Berenzweig, and S. Lawrence. The quest for ground truth in musical artist similarity. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2002.

[7] B. Fields, C. Rhodes, M. Casey, and K. Jacobson. Social playlists and bottleneck measurements: Exploiting musician social graphs using content-based dissimilarity and pairwise maximum flow values. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2008.

[8] A. Flexer, D. Schnitzer, M. Gasser, and G. Widmer. Playlist generation using start and end songs. In *Proc. Int. Conf. of Music Information Retrieval (ISMIR)*, 2008.

[9] Yajie Hu and Mitsunori Ogihara. Nextone player: A music recommendation system based on user behavior. In *Proc. Int. Conf. of the Soc. for Music Information Retrieval (ISMIR)*, 2011.

[10] B. Logan. Content-based playlist generation: Exploratory experiments. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2002.

[11] B. Logan. Music recommendation from song sets. In *Proc Int. Conf. on Music Information Retrieval (ISMRI)*, 2004.

[12] François Maillet, Douglas Eck, Guillaume Desjardins, Paul Lamere, et al. Steerable playlist generation by learning song similarity from radio station playlists. In *Proc. Int. Conf. on Music Information Retrieval*, 2009.

[13] N. Oliver and L. Kreger-Stickles. Papa: Physiology and purpose-aware automatic playlist generation. In *Proc. 7th Int. Conf. Music Inf. Retrieval*, pages 250–253, 2006.

[14] E. Pampalk, T. Pohle, and G. Widmer. Dynamic playlist generation based on skipping behavior. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2005.

[15] S. Pauws and S. van de Wijdeven. User evaluation of a new interactive playlist generation concept. In *Proc. Int.l Conf. on Music Information Retrieval (ISMIR)*, 2005.

[16] T. Pohle, E. Pampalk, and G. Widmer. Generating similarity-based playlists using traveling salesman algo- rithms. In *Proc. Int. Conf. on Digital Audio Effects (DAFX)*, 2005.

[17] M. Slaney and W. White. Measuring playlist diversity for recommendation systems. In *Proc. ACM workshop on audio and music computing multimedia*, 2006.

[18] R. Van Gulik and F. Vignoli. Visual playlist generation on the artist map. In *Proc. Int. Conf. on Music Information Retrieval(ISMIR)*, 2005.

[19] F. Vignoli and S. Pauws. A music retrieval system based on user-driven similarity and its evaluation. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2005.

IMPROVING THE RELIABILITY OF MUSIC GENRE CLASSIFICATION USING REJECTION AND VERIFICATION

Alessandro L. Koerich
Pontifical Catholic University of Paraná (PUCPR)
Federal University of Paraná (UFPR)
alekoe@computer.org

ABSTRACT

This paper presents a novel approach for post-processing the music genre hypotheses generated by a baseline classifier. Given a music piece, the baseline classifier produces a ranked list of the N best hypotheses consisting of music genre labels and recognition scores. A rejection strategy is then applied to either reject or accept the output of the baseline classifier. Some of the rejected instances are handled by a verification stage which extracts visual features from the spectrogram of the music signal and employs binary support vector machine classifiers to disambiguate between confusing classes. The rejection and verification approach has improved the reliability in classifying music genres. Our approach is described in detail and the experimental results on a benchmark dataset are presented.

1. INTRODUCTION

For over ten years the problem of classifying music genres has been the subject of intensive research and is the most widely studied area in Music Information Retrieval [2, 13]. Automatically classifying music by genre is a challenging problem considering that music is an evolving art and there are not clear edges between music genres. A variety of features and classification approaches have been proposed in the last years [1, 2, 6, 13]. Relatively high classification accuracies have been reported in recent papers that carry out experiments on benchmark datasets such as ISMIR 2004, GTZAN [15], and LMD [12]. Sturm [13] provides a comprehensive review of the approaches used for evaluating music genre classification. Sturm shows that over 92% of the papers approach evaluation of music genre classification systems by classifying several music excerpts and comparing the labels to a ground truth. Sturm states that the classification accuracy might not be a correct measure since it can not address the problem at all.

A few survey articles provide an overview of features and techniques used for music genre classification and related tasks [2, 9, 13]. Scaringella et al. [9] reviewed the techniques of audio feature extraction and classification for the task of genre classification only. Fu et al [2] provide a comprehensive review on audio-based classification and systematically summarize the state-of-the-art techniques for music classification, stressing the difference in the features and the types of classifiers used for different classification tasks such as music genre classification, mood classification, artist identification, instrument recognition and music annotation. The review paper of Sturm [13] focused on how music genre recognition systems are evaluated. Moreover, the field of music classification research is developing rapidly in the past few years, with new features and types of classifiers being developed and used. However, few works have evaluated the reliability of the current systems or make a deep analysis of the errors [14]. The definition of reliability used in this paper is borrowed from [5] which was referred to the proportion of correct answers among the accepted instances as a function of the rejection rate.

In this paper we propose the use of rejection and verification steps in an attempt to overcome the deficiencies of conventional classification approaches. The rejection focuses on not classifying instances that generate high uncertainty at the output of the classifier, while verification focuses on the confusions that may happen between particular music genres which is revealed by analyzing the confusion matrices. First, a baseline classifier, which takes into account all the possible music genres is used to classify the input music signal represented by a 68-dimensional feature vector. If this baseline classification scheme does not provide an output with high confidence, which is given by the *a posteriori* probability estimated to each possible music genre, the instance is rejected or may be post-processed at a verification stage, which extracts different features from the music signal and employs a set of binary SVM classifiers. We show in this paper that this approach is able to increase the reliability in music genre classification. It is important to notice that we tackle the problem from a pattern recognition perspective where music genres are treated as labels. The musicological aspect of the music genre classification task is not analyzed in this paper.

This paper is organized as follows. Section 2 presents a baseline music genre classification system. Section 3 introduces the concept of rejection in music genre classification. Section 4 introduces the concept of post-processing and verification of the output of a baseline classifier. Section 5

presents the experimental results of the proposed approach for rejection and verification on a benchmark dataset. Finally in the last section some conclusions are stated.

2. BASELINE MUSIC GENRE CLASSIFICATION SYSTEM

The baseline music genre classification system is composed of two main modules: feature extraction and classification as show in Figure 1. The feature extraction module uses the MARSYAS framework to extract seventeen audio features from $46ms$ frames of the audio signal with no overlap. The features are: zero crossing rate, the spectral centroid, roll-off frequency, spectral flux, and 13 Mel frequency cepstral coefficients, including MFCC0. Frame features are collapsed in a two-step process (texture windowing and computation of global mean and standard deviation) into a 68-dimensional feature vector for the whole audio excerpt [8]. For classification, two implementations were considered: A first version of the baseline system employs a multi-class support vector machine (SVM) algorithm with one-against all strategy. A second version of the baseline system employs the same features, but uses a multilayer perceptron neural network (MLP) trained with the backpropagation momentum algorithm. Such classification algorithms were chosen because both of them can provide estimations of the *a posteriori* probability of each class at the output [5, 16].

Let's consider a n-dimensional pattern recognition problem with c classes. The baseline classifiers perform a task of classifying an input music signal, represented by a n-dimensional feature vector $\overline{x} = [x_1, \ldots, x_n] \in \Re^n$ provided at the input and produce at the output *a posteriori* probability for each of the c possible classes, denoted as $P(\omega_1|\overline{x}), \ldots, P(\omega_c|\overline{x})$, where $\omega_1, \ldots, \omega_c$ denotes the c possible classes. Therefore, we can consider that the baseline classifiers produce at their output a c-dimensional vector $[P(\omega_1|\overline{x}), \ldots, P(\omega_c|\overline{x})]^T$ where $P(\omega_j|\overline{x})$ represents the support for the hypothesis that vector $\overline{x}$ submitted for classification comes from class ω_j. Furthermore, we can assume that such an output vector is ordered in a decreasing order according to the probability, from the most probable class, denoted as TOP 1 to the least probable class. The larger the probability, the more likely the class label ω_j. In real-life applications, the classification system has to come up with a single music genre hypothesis at the output or a rejection of the input if it is not certain enough about the hypothesis. Most of the current works on music genre classification does not consider the rejection option and force a decision using the MAX operator to select the music genre hypothesis which has the highest *a posteriori* probability as shown in Figure 1, i.e. the TOP 1 hypothesis, denoted as ω' and computed by Equation 1.

$$\omega' = \arg \max_{\omega_1 \leq \omega_i \leq \omega_c} P(\omega_i|\overline{x}) \quad (1)$$

The main drawback of using the MAX operator as decision rule is that it is very severe and it often overlooks the uncertainty that may be present at the output of the classifiers. Such an operator will return a class regardless if the highest probability is close to 1.0 or lower than 0.5. The probabilities estimated by the classifier can be associated to how confident it is in assigning a given class to a input vector.

3. REJECTION OF MUSIC GENRES

How can we handle the uncertainty at the output of a multiclass classifier? Maybe the simplest way is to employ rejection [3–5]. The concept of rejection admits the potential refusal of a music genre hypothesis if the classifier is not certain enough about the genre hypothesis. In our case, the probabilities assigned to each output of the baseline classifiers give evidence about the certainty. The refusal of a music genre hypothesis may be due to two different reasons: there is not enough evidence to come to a unique decision since more than one music genre hypothesis among the c possible music genres appear adequate; there is not enough evidence to come to a decision since no music genre among the c genres appears adequate. In the first case, it may happen that the confidence scores do not indicate a unique decision in the sense that there is not just one confidence score exhibiting a value close to one. In the second case, it may happen that there is no confidence score exhibiting a value close to one. Therefore, the confidence scores assigned to the music genre hypotheses in the N best hypothesis list should be used as a guide to establish a rejection criterion.

Bayes decision rule already embodies a rejection rule, namely, find the maximum of $P(\omega_i|\overline{x})$ but check whether the maximum found exceeds a certain threshold value or not. Due to decision-theoretic concepts this reject rule is optimum for the case of insufficient evidence if the closed-world assumption holds and if the *a posteriori* probabilities are known [10]. This suggests the rejection of a music genre hypothesis if the confidence score for that hypothesis is below a threshold λ. In the context of our baseline classifiers, the task of the rejection mechanism is to, based on output vector $[P(\omega_1|\overline{x}), \ldots, P(\omega_c|\overline{x})]^T$ provided at the output of the classifiers, which is ordered in a decreasing order according to the probability, decide whether the best music genre hypothesis, which is so far called the TOP 1, can be accepted or not. Therefore, the conventional classification approach is modified. Now, the highest *a posteriori* probability provided by the MAX operator is not simply accepted, but it is compared with a threshold (λ). If such a probability is greater than λ then the music genre is assigned to the input vector, otherwise, no label is assigned to the input vector $\overline{x}$ and it is rejected. This novel decision scheme is show in Figure 2).

In summary, the rejection rule is given by:

1. The TOP 1 music genre hypothesis is accepted whenever $P(\omega'|\overline{x}) \geq \lambda$

2. The TOP 1 music genre hypothesis is rejected whenever $P(\omega'|\overline{x}) < \lambda$

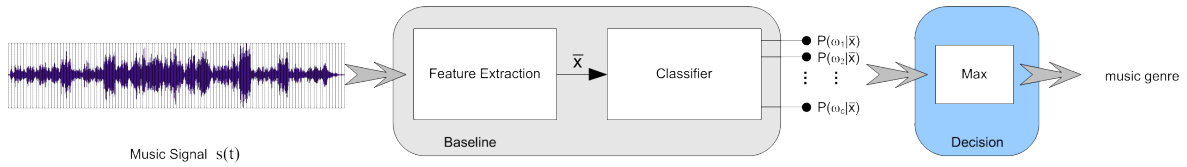


Figure 1. An overview of the baseline music genre classification system

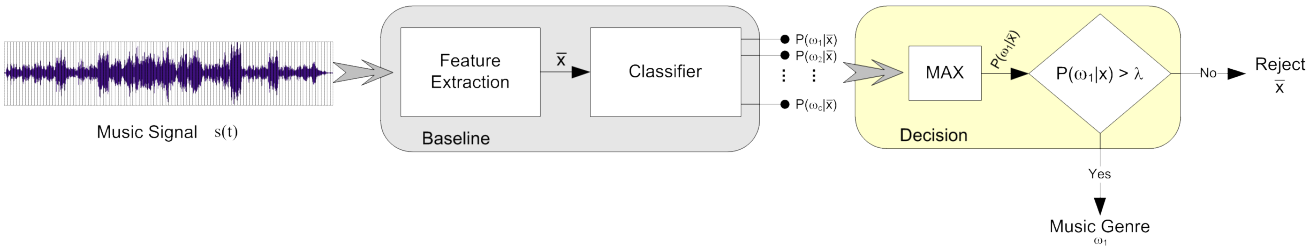


Figure 2. An overview of the rejection scheme for the baseline music genre classification

where λ is a rejection threshold and $P(\omega'|\overline{x})$ is the *a posteriori* probability assigned by the classifier to the best music genre hypothesis ω'.

4. VERIFICATION OF MUSIC GENRES

The question that may arise is if we can do better than simply rejecting the instances that the classifier is not able to classify with confidence. Probably the straightforward way to proceed is to re-classify the rejected instances using a different kind of classifier or even, represent such rejected instances using a different feature set and submit them to further classification steps [3, 5].

The solution that we propose in this paper is to make use of all information provided at the output of the classifier, that is, the ranked list of the N best music genre hypothesis $[P(\omega_1|\overline{x}), \ldots, P(\omega_c|\overline{x})]^T$, to deal with the rejected instances. The main idea is to re-classify the rejected instances using specialized classifiers. To such an aim we assume that the baseline classifiers are somewhat trustworthy. This means that even if the baseline classifier is not able to rank the correct music genre at the TOP 1 position, the correct genre hypothesis will show up among the first best hypothesis. Therefore, we need to analyze carefully the output of the baseline classifier to understand its behavior. In particular, our interest is to find out if there is any relevant information that can be used to guide us in building more specialized classifiers.

To such an aim, we look at the confusion matrix of the baseline classifiers. A confusion matrix shows us how the errors are distributed across the classes. Our prime interest is to find out where the most significant misclassification has occurred. In particular, we look for large off-diagonal entries of the matrix which might indicate a difficult two-class problem that needs to be tackled separately. Based on the analysis of the confusion matrix, we can build two-class classifiers to handle the most significant confusions.

Consider a set $\mathcal{D}$ of L two-class classifiers. Once the TOP 1 music genre hypothesis is rejected we look at the class of the second best music genre hypothesis, denoted as ω'' which is computed by Equation 2:

$$\omega'' = \arg\max 2_{\omega_1 \leq \omega_i \leq \omega_c} P(\omega_i|\overline{x}) \qquad (2)$$

where max2 is an operator that returns the second greatest probability provided by the classifier.

Let ω' and ω'' the music genre assigned to the TOP 1 and TOP 2 hypothesis respectively, then we verify if there exists a binary classifier, so far called verifier, $D(\omega', \omega'') \in \mathcal{D}$ to handle the confusion between the music genres ω' and ω''. If so, the verification stage is invoked. Otherwise a final decision is taken and the input vector $\overline{x}$ is rejected. The complete classification and verification approach can be summarized by the pseudo-code as follows:

```
CLASSIFY INSTANCE (x̄, B, D)
// Input: An instance represented by a feature vector x̄, a
// multiclass baseline classifier B, a set of two-class
// verifiers D
// Output: A music genre assigned to x̄ or the
// rejection of x̄
if P_B(ω' | x̄) ≥ λ then
    return ω' as the music genre
else
    read P(ω'' | x̄)
    if D_{ω',ω''} ∈ D then
        classify x̄ with D_{ω',ω''}
        return arg max_{ω',ω''} {P_D(ω' | x̄), P_D(ω'' | x̄)} as
                the music genre
    else
        reject the instance x̄
    endif
endif
```

The verification stage is loosely coupled to the baseline classifier. The only information provided by the baseline classifiers is the label of the two best hypothesis, say ω' and ω''. Such information is used to selected the proper verifier. The verifier, may or may not use the same feature set and classification algorithm of the baseline, however, in this paper we have chosen a different feature set which is based on the texture features extracted from the spectrogram of the music signal [1]. This feature set was chosen because it has provided very interesting discriminat-

N best hypothesis	Correct Classification Rate (%)	
	Baseline SVM	Baseline MLP
TOP 1	53.00 ± 0.67	52.11 ± 3.09
TOP 2	71.11 ± 1.39	70.34 ± 2.62
TOP 3	81.33 ± 0.00	79.82 ± 2.93
TOP 5	91.00 ± 1.45	92.10 ± 1.09
TOP 7	95.89 ± 0.84	95.21 ± 2.27
TOP 8	97.78 ± 0.96	98.33 ± 1.03

Table 1. Correct classification rate for the baseline classifiers

ing results in the previous works [1]. This 59-dimensional feature vector is used to train binary SVM verifiers. An overview of the complete approach is shown in Figure 3.

5. EXPERIMENTAL RESULTS

The performance of the classification-verification approach was evaluated on a subset of the LMD [11]. The LMD is made up of 3,227 full-length music pieces uniformly distributed along 10 classes: Axé (Ax), Bachata (Ba), Bolero (Bo), Forró (Fo), Gaúcha (Ga), Merengue (Me), Pagode (Pa), Salsa (Sa), Sertaneja (Se), and Tango (Ta). In our experiments, we use 900 music pieces from the LMD, which are split into 3 folds of equal size (30 music pieces per class). The splitting is done using an artist filter, which places the music pieces of an specific artist exclusively in one, and only one, fold of the dataset. Furthermore, in our particular implementation of the artist filter we added the constraint of the same number of artists per fold. Therefore, all results reported in this section refer to 3-fold cross validation, unless otherwise noted.

5.1 Baseline Classifiers

The first baseline classifier is a 10-class multilayer perceptron neural network with 68 input units, 40 units at the hidden layer and 10 units at the output layer and the following learning parameters: step width = 0.2, momentum = 0.5, flat spot elimination = 0.1, max non-propagated error = 0.1, and 100 learning cycles. After such a number of cycles, the generalization of the network starts to decrease according to the mean squared error measured on a validation dataset. The network provides estimates *a posteriori* probabilities and the value of each output necessary remains between zero and one because of the sigmoidal function used. The second baseline classifier is a 10-class SVM with Gaussian kernel. The gamma and cost parameters were found by a grid search on a validation dataset. Pairwise coupling is used to handle multi-class classification.

The performance of the baseline classifiers was evaluated using the correct classification rate which is defined as the ratio between number of samples correctly classified and the number of samples tested. Table 1 shows the performance of both baseline classifiers taken into account if the correct music genre is among the TOP N best hypothesis. These results support our previous assumption that the baseline classifiers provide a somewhat trustworthy output. For instance, the correct music genre is among the TOP 5 best hypotheses for more than 91% of the cases.

Class	Ax	Ba	Bo	Fo	Ga	Me	Pa	Sa	Se	Ta
Ax	41	3	0	0	9	2	10	7	18	0
Ba	2	67	4	5	1	3	3	3	2	0
Bo	1	4	45	6	2	1	10	11	5	5
Fo	0	6	3	43	13	3	8	3	11	0
Ga	13	0	5	8	44	8	6	5	1	0
Me	1	3	1	2	6	66	3	5	3	0
Pa	8	4	11	7	0	1	38	15	5	1
Sa	12	2	9	3	3	12	12	33	4	0
Se	14	1	7	9	7	2	6	9	35	0
Ta	1	0	7	0	3	0	0	0	0	79

Table 2. Confusion matrix for the verification set

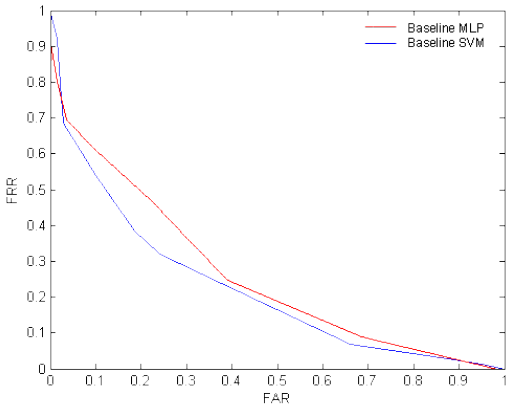


Figure 4. ROC curve for the SVM baseline and MLP baseline.

The information on Table 1 opens up a plenitude of ways to improve the performance of the baseline classifiers. However, this is not the goal of this paper. Here we focus on the concept of rejection in an attempt to improve the reliability of the baseline classifiers, recalling that reliability is the proportion of correct answers among the accepted instances as a function of the rejection rate.

5.2 Rejection Option

In this section we evaluate how the rejection can improve the reliability of the baseline classifiers. We measure the rejection accuracy in terms of its rate of erroneous behavior for each input as false rejection rate $FRR = FR/(FR + CA)$ on instances which had been recognized correctly, and false acceptance rate $FAR = FA/(FA + CR)$ on instances which had been misrecognized, where CA denotes correct acceptance, CR denotes correct rejection, FA denotes false acceptance, and FR denotes false rejection. These two types of error naturally trade off; for example, raising the rejection threshold reduces FAR but at the cost of increased FRR. Therefore, for each measure, we sweep a rejection threshold (λ) across its entire range of values, plotting the two error types as a receiver-operating characteristic (ROC) curve. A curve reaching closer to the origin indicates a superior confidence measure, one enabling low rates of both error types simultaneously. Figure 4 shows the ROC curves as a function of the rejection rate which is defined in terms of the λ value.

Figure 4 shows that the curve reaches close to the ori-

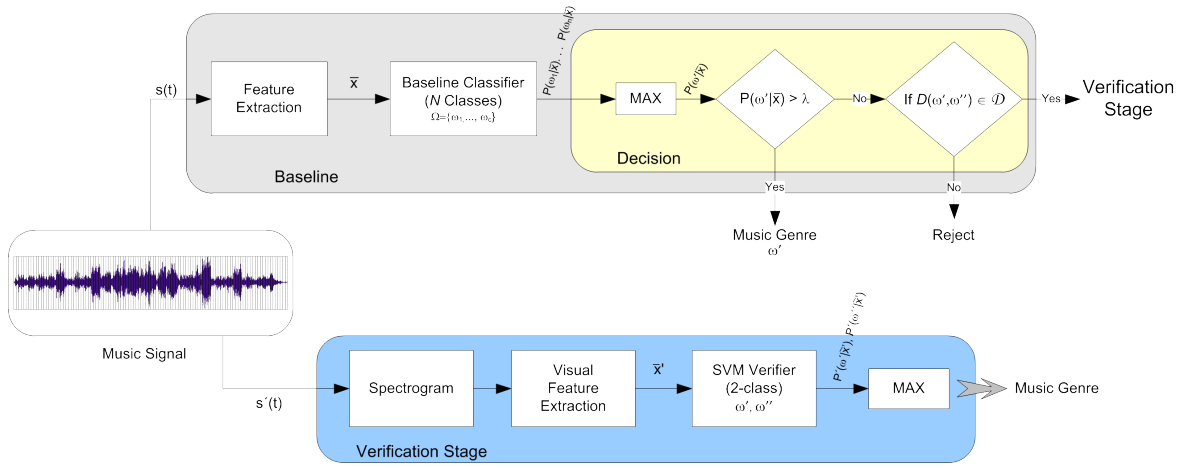


Figure 3. An overview of the complete approach including classification, rejection and verification

Rejection Rate (%)	Error Rate (%)	
	Baseline SVM	Baseline MLP
0	47.00	47.89
20	38.59	40.68
40	30.46	32.18
50	23.94	30.34

Table 3. Reduction on the error rate for different rejection rates

gin for FAR and FRR equal 0.3. This corresponds to rejection rates between 20% and 50%. Table 3 shows the corresponding error rates for these rejection levels. Therefore, for the remainder of the paper we consider 40% of rejection rate. At this rejection rate the reliability for music genre increases from 53.00% to 69.54% and from 52.11% to 67.82% for the SVM and MLP classifier respectively. This represents an improvement of about 16% which is very interesting in the context of music genre classification.

5.3 Verifiers

Since the aim of this paper is not to handle all possible confusions but to show how the rejection and verification can improve the reliability of music genre classification systems, we have built very few verifiers among the 45 possible ones. Therefore, there are binary verifiers only to deal with the most significant confusions which were found by analyzing the confusion matrix generated from the output of the baseline classifiers on a validation dataset. The two-class verifiers employ the support vector machines algorithm with Radial Basis Function kernel and trained with the sequential minimal optimization method. A grid-search algorithm was used to optimize the cost and the gamma parameters. The two-class classifiers were built based on the analysis of the confusion matrix shown in Table 2. Since our goal is not to handle all the confusion of the base classifiers, in this table are highlighted the two highest confusions, which are between classes Axé and Sertaneja and between classes Pagode and Salsa. Therefore, two binary verifiers were built to deal with the confusing classes $\mathcal{D} = \{D_{\text{Ax,Se}}, D_{\text{Pa,Sa}}\}$.

Classifier	Correct Classification Rate (%)	
	Ax-Se Class	Pa-Sa Class
Baseline SVM	42.22 ± 0.12	39.44 ± 0.01
Baseline MLP	43.33 ± 0.11	40.56 ± 0.13
Verifier	82.08 ± 13.05	78.33 ± 3.33

Table 4. Correct classification rates for two classes

Approach	Rejection Rate (%)	Reliability (%)
SVM	0	53.00 ± 0.67
MLP	0	52.11 ± 3.09
SVM + Rej	40	69.54 ± 0.99
MLP + Rej	40	67.82 ± 3.68
SVM + Rej + Verif	40	71.35 ± 1.00
MLP + Rej + Verif	40	69.19 ± 3.93

Table 5. Reliability for the baseline SVM and MLP, baseline+rejection (Rej) and baseline+rejection+verification (Verif)

The performance of the two-class verifiers is shown in Table 4. The results refer to the same 3-fold cross validation protocol but for the verifier we have just a subset where each fold holds only the instances labeled with the confusing classes. We include in this table also the performance of the baseline classifiers considering only the joint correct classification rate on the instances of these pair of classes.

Table 4 shows, that as expected, the verifier achieves a much higher rate than the baseline classifiers. However, our main aim is to improve the overall reliability using the verifier. Therefore, we apply the approach proposed in Section 4, where we submit an instance to the verifier if it is rejected and if it is classified by the baseline classifier at the TOP 1 and TOP 2 positions as one of the two pair of classes: Ax-Se or Se-Ax and Pa-Sa or Sa-Pa. Table 5 summarizes the results of the complete approach.

Table 5 shows that both rejection and verification are effective in improving the correct music genre classification rate. Rejection brings about an impressive increasing in the correction classification rate, however, 40% of the instances were rejected and should be handled in a different way, such as by humans. An alternative way is to have

a second stage to reprocess the rejected instances. Even if we have not handled all the rejected instances, but only those who felt on the most confusing classes that we have chosen, it is possible to observe a further, but moderated, improvement in the classification rates.

The Friedman test with the post hoc Shaffer's static procedure was employed to evaluate if there are statistically significant differences between the results show in Table 5. The multiple comparison statistical test has show that the p-value is lower than the corrected critical value in most of the cases, showing a statistically significant difference between the baseline, baseline+rejection and baseline+ rejection+verification at 95% confidence level.

6. CONCLUSIONS

In this paper we have presented a rejection and verification approach to automatic music genre classification that post-process the output of a baseline classifier in an attempt to improve the reliability in classifying music genres. The output of the baseline classifiers is evaluated and the probabilities provided by these classifiers serve as a guide to either reject or accept the input instance. Furthermore, the rejected instances may be re-classified at a verification stage using a different approach if they were previously classified by the baseline classifier as belonging to specific classes. The performance resulting from the combination of the baseline, rejection and verification is significantly better than that achieved by the baseline classifiers alone. For instance, the baseline classifiers alone achieves a classification rate of 53%. The rejection stage improves the reliability to 69.54% at a rejection level of 40% and the verification stage further improves it to 71.35%. In spite of the current verification stage deal with only two pair of confusing classes, it was able to improve the reliability in almost 2%. Given the high number of confusions between other classes, as show in Table 2, we expect to achieve further improvement by adding more verifiers out of the 45 possible ones.

Compared with previous works that use the same features, the same dataset, and the same experimental protocol [1, 7], the results reported in this paper represent a significant improvement in terms of classification rate and reliability. It is difficult to compare the performance of the proposed approach with other results available in the literature due to the differences in the experimental conditions.

In spite of the good results achieved, there are some shortcomings related to the use of the second stage. For instance, the second stage depends on the results of the first stage and on the availability of a binary classifier to handle the confusions between specific classes. As future work we plan to validate the proposed approach on other datasets, such as the Magnatagatune and the Million Song Dataset.

7. ACKNOWNLEDGMENTS

This research is partially supported under CAPES/Fulbright grant BEX1770/712-9, FA grant 203/12 and CNPq grants 306.703/2010-6 and 472.238/2011-6.

8. REFERENCES

[1] Y.M.G. Costa, L.E.S. Oliveira, A.L. Koerich, F. Gouyon, and J.G. Martins. Music genre classification using LBP textural features. *Signal Processing*, 92(11):2723–2737, 2012.

[2] Z. Fu, G. Lu, K. M. Ting, and D. Zhang. A survey of audio-based music classification and annotation. *IEEE Trans. on Multimedia*, 13(2):303–319, 2011.

[3] A.L. Koerich. Rejection strategies for handwritten word recognition. In *Int'l Workshop on Frontiers in Handwriting Recognition*, pp.479–484, Tokyo, Japan, 2004.

[4] A.L. Koerich, L.E.S. Oliveira, and A.S. Britto Jr. Verification of unconstrained handwritten words at character level. In *Int'l Conf. on Frontiers in Handwriting Recognition*, pp.39–44, Kolkata, 2010.

[5] A.L. Koerich, R. Sabourin, and C. Y. Suen. Recognition and verification of unconstrained handwritten words. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(10):1509–1522, 2005.

[6] T. Lidy, C.N. Silla, O. Cornelis, F. Gouyon, A. Rauber, C.A.A. Kaestner, and A.L. Koerich. On the suitability of state-of-the-art music information retrieval methods for analyzing, categorizing and accessing non-western and ethnic music collections. *Signal Processing*, 90(4):1032–1048, 2010.

[7] M. Lopes, F. Gouyon, A.L. Koerich, and L.E.S. Oliveira. Selection of training instances for music genre classification. In *Int'l Conf. on Pattern Recognition*, pp.4569–4572, Istambul, Turkey, 2010.

[8] S.R. Ness, A. Theocharis, G. Tzanetakis, and L.G. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic SVM outputs. In *ACM Multimedia Conf.*, pp.705–708, Beijing, China, October 2009.

[9] N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content - a survey. *IEEE Signal Processing Magazine*, 23(2):133–141, 2006.

[10] J. Schurmann. *Pattern Classification: A Unified View of Statistical and Neural Approaches*. John Wiley and Sons, 1996.

[11] C.N. Silla, A.L. Koerich, and C.A.A. Kaestner. The Latin music database. In *Int'l Conf. on Music Information Retrieval*, pages 451–456, Philadelphia, USA, 2008.

[12] C.N. Silla, A.L. Koerich, and C.A.A. Kaestner. A machine learning approach to automatic music classification. *Journal of the Brazilian Computer Society*, 14(3):7–18, September 2008.

[13] B. L. Sturm. A survey of evaluation in music genre recognition. *Adaptive Multimedia Retrieval*, 2012.

[14] B. L. Sturm. Two systems for automatic music genre recognition: What are they really recognizing. In *Int'l ACM Workshop on MIR with User-centered and Multimodal Strat.*, pp.69–74, Nara, Japan, 2012.

[15] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing*, 10(5):293–302, 2002.

[16] T.-F. Wu, C.-J. Lin, and R.C. Weng. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 5:975–1005, 2004.

ROBOTABA GUITAR TABLATURE TRANSCRIPTION FRAMEWORK

Gregory Burlet and Ichiro Fujinaga
Centre for Interdisciplinary Research in Music Media and Technology
McGill University, Montréal, Québec, Canada
gregory.burlet@mail.mcgill.ca, ich@music.mcgill.ca

ABSTRACT

This paper presents *Robotaba*, a web-based guitar tablature transcription framework. The framework facilitates the creation of web applications in which polyphonic transcription and guitar tablature arrangement algorithms can be embedded. Such a web application is implemented, and consists of an existing polyphonic transcription algorithm and a new guitar tablature arrangement algorithm. The result is a unified system that is capable of transcribing guitar tablature from a digital audio recording and displaying the resulting tablature in the web browser. Additionally, two ground-truth datasets for polyphonic transcription and guitar tablature arrangement are compiled from manual transcriptions gathered from the tablature website `ultimate-guitar.com`. The implemented transcription web application is evaluated on the compiled ground-truth datasets using several metrics.

1. INTRODUCTION

Tablature has become the primary form of communication between guitarists on the Internet. Guitar tablature is a music notation system with a six-line staff that represents the strings on a guitar. A numeric entry on a line represents the fret to depress on a particular string (Figure 1).

Manually transcribing guitar tablature from an audio recording is a difficult and laborious task, even for experienced guitarists. In response to the time-consuming process of manual transcription, automatic music transcription systems aim to extract a symbolic music score from an acoustical signal [5]. Specifically, automatic guitar tablature transcription systems transform a digital guitar signal into tablature notation. The task of automatic guitar tablature transcription can be decomposed into two subproblems: polyphonic transcription and guitar tablature arrangement. Polyphonic transcription algorithms extract the pitch, onset time, and duration of notes occurring in an audio recording. Guitar tablature arrangement algorithms assign a string and fret combination to each note occurring in an input music score. Adding more ambiguity to the transcription process, the guitar can produce the same note in several ways. For example, there exists six string and fret combinations that can produce the note E4 on a 24-fret guitar in standard tuning (Figure 1).

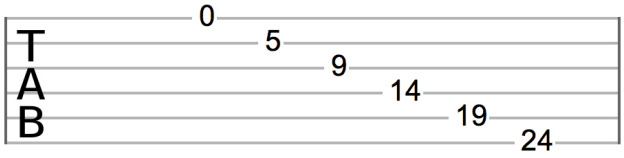


Figure 1. Tablature notation depicting six different string and fret combinations to perform the note E4 on a 24-fret guitar in standard tuning.

While several polyphonic transcription and guitar tablature arrangement algorithms have been proposed in the literature, no frameworks have been developed to facilitate the combination of these algorithms to produce an automatic guitar tablature transcription system. Moreover, after a new polyphonic transcription or guitar tablature arrangement algorithm is developed, the code has no immediately available vessel to be used by music researchers and the large community of guitarists on the Internet.

A web-based guitar tablature transcription framework, entitled *Robotaba*, has been designed and implemented to facilitate the creation of guitar tablature transcription web applications in which polyphonic transcription and guitar tablature arrangement algorithms can be embedded. An existing polyphonic transcription algorithm and a new guitar tablature arrangement algorithm is implemented; these algorithms are embedded in a transcription web application using the Robotaba framework. Two ground-truth datasets have been compiled to evaluate the performance of the implemented guitar tablature transcription system.

The structure of this paper is as follows: The next section provides an overview of polyphonic transcription and guitar tablature arrangement algorithms. Section 3 presents the design of Robotaba, followed by a description of the implemented transcription web application in Section 4. Section 5 presents the compiled ground-truth datasets and their use in the evaluation of the implemented polyphonic transcription and guitar tablature arrangement algorithms.

2. LITERATURE REVIEW

Though the transcription of monophonic musical passages is considered a solved problem [5], the transcription of

polyphonic music is still an open problem [1]. Several techniques have been proposed to accomplish the task of polyphonic transcription, including human audition modelling [12]; salience methods, which apply transformations to the input audio signal in order to emphasize the underlying fundamental frequencies [17]; iterative and joint fundamental frequency estimation algorithms followed by note tracking algorithms [2]; and a variety of machine learning methods such as non-negative matrix factorization [14], support vector machines [8], neural networks [6], and hidden Markov models [10].

Several algorithms have also been proposed to generate tablature from a symbolic music score according to criteria that minimizes the performance difficulty of the tablature arrangement. Heijink and Meulenbroek [4] established that guitarists have a disposition toward instrument fingering positions that are biomechanically easy to perform. Specifically, guitarists favour hand positions near the head of the guitar and avoid composing arrangements that require extensive hand repositioning and large finger spans. Shortest-path graph search algorithms [13], constraint satisfaction algorithms [9], neural networks [15], and genetic algorithms [16] have been used to search for tablature arrangements with minimal performance difficulty.

3. FRAMEWORK DESIGN

The Robotaba transcription framework is composed of three modules: a polyphonic transcription module, a guitar tablature arrangement module, and a guitar tablature engraving module (Figure 2).

Three benefits arise from this modular design: First, each module can be used independently or together. Used independently, an input file is sent directly to a module for processing, which returns a result instead of passing the output to the next module in the workflow. Using each module in sequence, guitar tablature can be generated from an input audio file and displayed in the web browser. Second, the modular design facilitates algorithm interchangeability. Assuming an algorithm produces valid output, it can be inserted into a module without disturbing the functionality of surrounding modules. As a result, the transcription framework can accommodate new state-of-the-art polyphonic transcription or guitar tablature arrangement algorithms without substantial changes to the web application. Third, the use of a single symbolic music file format for data interchange between modules promotes polyphonic transcription and tablature arrangement algorithms to adhere to a common interface. Robotaba uses the 2012 release of the music encoding initiative (MEI): an extensible markup language (XML) file format that encodes symbolic music notation in a hierarchical fashion [11].

3.1 Polyphonic Transcription Module

The polyphonic transcription module accepts an audio file as input, which is passed to the polyphonic transcription algorithm embedded in the module. The polyphonic transcription algorithm is responsible for generating an MEI file containing the estimates of note events occurring in the input audio file. The polyphonic transcription module optionally postprocesses the output symbolic music file by limiting the number of simultaneous notes to six, and discarding or transposing estimated notes that are outside of the range of a specific guitar. Properties of the guitar (number of frets, tuning, and capo position) [1] and postprocessing options are specified by the user of the web application.

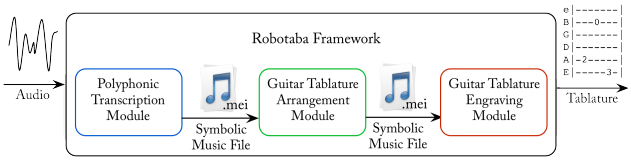


Figure 2. Modular architecture of the Robotaba guitar tablature transcription framework.

3.2 Guitar Tablature Arrangement Module

The guitar tablature arrangement module accepts an MEI file as input, which is mandatorily preprocessed in an identical manner as the postprocessing step of the polyphonic transcription module described in the previous section. The preprocessed MEI file is subsequently passed to the guitar tablature arrangement algorithm embedded in the module, which is responsible for assigning a guitar string and fret combination to each note occurring in the MEI file.

3.3 Guitar Tablature Engraving Module

The guitar tablature engraving module is responsible for parsing an MEI file containing a sequence of note events that have each been assigned a guitar string and fret combination and displaying the encoded tablature in the web browser. Robotaba uses the digital guitar tablature engraving library *AlphaTab* [2] to render tablature symbols on the hypertext markup language (HTML) canvas element. AlphaTab parses drawing scripts called AlphaTex, in which structured keywords inform the rendering engine about the contents of the tablature and how it should be displayed. When an MEI file is passed to the tablature engraving module, the contents of the file are converted to an AlphaTex drawing script to be rendered by AlphaTab. A rendered tablature score can be seen in Figure 1.

3.4 Framework Implementation

Robotaba is implemented using Django, [3] a Python web framework that facilitates rapid development of database-driven web applications by automatically translating Python classes called *models* into relational database tables. Models are created for audio and symbolic music files, as well as their associated metadata. Additionally, the database

[1] A capo is a device that is clipped onto the fretboard of a guitar and raises the pitches of the open strings.

[2] `www.alphatab.net`

[3] `www.djangoproject.com`

stores the user-specified parameters used to generate a transcription, such as the guitar properties and file pre and post-processing options, to allow reproducibility of results.

4. TRANSCRIPTION WEB APPLICATION

Using the Robotaba framework, a web application for guitar tablature transcription is developed that incorporates the polyphonic transcription and guitar tablature arrangement algorithms described in this section. [4]

4.1 Polyphonic Transcription Algorithm

A polyphonic transcription application, which uses the state-of-the-art algorithm proposed by Zhou and Reiss [17], is implemented. This algorithm was selected for several reasons: First, this algorithm ranked highest out of the polyphonic transcription algorithms evaluated in the Music Information Retrieval Evaluation eXchange (MIREX) on the piano dataset from 2007–2012 when considering the accuracy of pitch and note onset times only. Second, the authors tuned underlying parameters of the algorithm according to a dataset composed of both piano and guitar recordings [17]. Third, this algorithm is capable of performing polyphonic transcriptions in realtime. Finally, the source code of this algorithm is open source.

The aforementioned polyphonic transcription algorithm is distributed as a Vamp plugin written in the C++ programming language. A Vamp plugin is an audio feature extraction module that must be "plugged into" a host application. [5] In order to integrate the polyphonic transcription Vamp plugin into Robotaba, the plugin is first divorced from the host to produce a standalone application. A Python interface is created using the Boost.Python library [6] to access the standalone application from Robotaba. A Python application is implemented, which sets parameters of the polyphonic transcription algorithm, imports an audio file, sends the audio data to the Python bindings of the polyphonic transcription Vamp plugin, and generates an MEI document containing the resulting note event estimates.

4.2 Guitar Tablature Arrangement Algorithm

A new guitar tablature arrangement algorithm entitled *A-star-guitar* is developed, written in the Python programming language, and embedded in the Robotaba guitar tablature arrangement module. Extending the approach proposed by Sayegh [13], which uses the Viterbi algorithm to search for an optimal path through a weighted graph of candidate fretboard locations for notes in a *monophonic* musical passage, A-star-guitar uses the popular A* pathfinding algorithm [3] to search for an optimal tablature arrangement of a sequence of notes and chords in a *polyphonic* musical passage.

The A* pathfinding algorithm searches for an optimal path through a directed graph, in which vertices represent candidate string and fret combinations for a note or chord in a symbolic music score. Candidate string and fret combinations are calculated by considering the number of frets on the user's guitar, the tuning, and optional fret position of a capo. Vertices that correspond to adjacent notes or chords in the music score are connected by an edge.

The weight of an edge $w_{ij} \in \mathbb{N}$ between vertices i and j represents the biomechanical difficulty associated with the transition between the two hand positions on the fretboard. Following the study of left-hand movements of professional guitar players by Heijink and Meulenbroek [4], the edge weight between two vertices is the cumulation of three biomechanical complexity factors: the fretwise distance that the fretting hand must move to accommodate note transitions, the fretwise finger span required to perform chords, and a penalty of one if the fretting hand surpasses the seventh fret. The values of this penalty and fret threshold number were chosen on the basis of preliminary tests and encourage tablature arrangements near the beginning of the fretboard. In the event of a note played by depressing fret number f, followed by a chord comprised of multiple notes with the set of fret numbers g, the fretwise distance is calculated by the expression

$$\mathrm{abs}\left(f - \left(\frac{\max(g) - \min(g)}{2}\right)\right). \qquad (1)$$

The A* algorithm uses a heuristic function that returns an estimate of the summation of edge weights from a given vertex to the goal vertex. This must be an admissible heuristic: the cost of traversing the graph from a vertex to the goal vertex must not be overestimated. Using the proposed biomechanical complexity factors to assign weights to edges, the heuristic function is zero for all vertices and the resulting algorithm is effectively the Dijkstra pathfinding algorithm [3]. To illustrate why the heuristic function is zero, consider a music score with a sequence of identical notes that may be performed by repeatedly plucking an open string on the guitar. [7] In this case, each edge connecting the open-string vertices has a weight of zero: there is no fretwise distance between notes, there are no chords in the music score, and no frets are depressed.

Figure 3 displays a directed acyclic weighted graph of candidate string and fret combinations for an example music score consisting of four notes above the tablature arrangement produced by the A* search algorithm.

5. TRANSCRIPTION EVALUATION

This section focuses on the procedure for evaluating the implemented transcription web application, specifically the polyphonic transcription and guitar tablature arrangement algorithms embedded within.

5.1 Ground-truth Datasets

For the purposes of evaluating polyphonic transcription and guitar tablature arrangement algorithms, two new ground-truth datasets are presented. The datasets were compiled

[4] ddmal.music.mcgill.ca/robotaba
[5] www.vamp-plugins.org
[6] www.boost.org/libs/python/doc
[7] The term *open string* refers to a pluck whereby no fret is depressed.

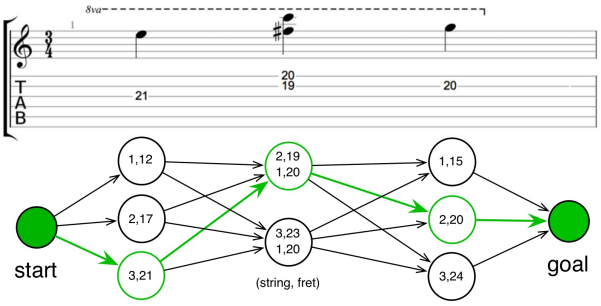


Figure 3. A directed acyclic weighted graph of candidate string and fret combinations for a music score consisting of four notes. The bold edges in the graph identify the optimal path found by the A* search algorithm. Edge weights are omitted for space considerations.

by harvesting the wealth of community-moderated and publicly available manual transcriptions of popular musical works uploaded to the *Ultimate Guitar* tablature website. [8]

Manual transcriptions were downloaded as *Guitar Pro* symbolic music files: a proprietary file format that is read and manipulated by the Guitar Pro desktop application. [9] 75 Guitar Pro files were selected from the Ultimate Guitar Top 100 list and the Ultimate Guitar Fresh Tabs list. The Ultimate Guitar Top 100 list sorts every Guitar Pro file uploaded to the website by its rating—from one to five stars—and displays the top 100 files. The Ultimate Guitar Fresh Tabs list is a catalogue of Guitar Pro files that have recently been uploaded to the website and is sorted by number of views. Only uploaded tablature with a five-star rating agreed upon by at least ten unique users was considered for selection. Tablature was selected on the basis of musical genre, average polyphony, and tempo, in order to compile a set of pieces with a variety of different attributes.

5.1.1 Polyphonic Guitar Transcription Dataset

To form the polyphonic guitar transcription dataset, several preprocessing steps were first applied to each Guitar Pro file. These symbolic music files often contain multiple instrument tracks, many of which are not guitar tracks. Extraneous tracks containing instruments such as the bass guitar, drums, and vocals were removed. Finally, all tempo, volume, and pan automations were removed from the remaining guitar track.

Each guitar track was then synthesized using Guitar Pro's *clean* and *distortion* guitar model to create two audio files. [10] Subsequently, a file listing the pitch, onset time, and duration of notes occurring in the synthesized audio files was generated. This ground-truth file was created by first exporting the Guitar Pro file as a MusicXML file and retrieving pertinent information about the music score from a MusicXML parser program, which was originally developed to gather information from symbolic music scores for the purpose of finding patterns in the relationships between melody, lyrics, and instrumentation [7].

The polyphonic guitar transcription dataset consists of 75 isolated guitar tracks; 125,192 note events; 30,914 chords; and an average tempo of 112 beats per minute. There are approximately five and a half hours of clean guitar recordings and five and a half hours of distortion guitar recordings, yielding approximately eleven hours of synthesized audio in total.

5.1.2 Guitar Tablature Arrangement Dataset

The ground-truth dataset for guitar tablature arrangement was compiled using the same 75 Guitar Pro files that were collected and preprocessed for the polyphonic guitar transcription dataset. Excerpts were selected from the symbolic music files on the basis of overall length (no one excerpt exceeds eight measures), the number of times the excerpt occurred throughout the entire piece, and the average polyphony of the excerpt. Each excerpt in Guitar Pro was exported as a MusicXML file, which was subsequently converted to an MEI file using an open-source MusicXML to MEI conversion application written in Python. [11]

The ground-truth dataset for guitar tablature arrangement consists of 75 tablature arrangements—with 4,845 notes and 1,143 chords—encoded in both the MEI and MusicXML symbolic music file formats.

5.2 Polyphonic Transcription Evaluation

The implemented polyphonic transcription algorithm is evaluated using the compiled ground-truth dataset for polyphonic guitar transcription. Using an evaluation procedure similar to the MIREX multiple fundamental frequency estimation and tracking task, the output of the polyphonic transcription algorithm on each audio recording in the dataset is compared to the corresponding ground-truth note events using the metrics of precision, recall, and f-measure. Precision describes the ratio of correctly transcribed note events to the total number of estimated note events. Recall describes the ratio of correctly transcribed note events to the total number of ground-truth note events. The f-measure statistic combines precision and recall into a single metric. An estimated note event is deemed to be correctly transcribed if the fundamental frequency is within 50 cents of the ground-truth note event and the onset time is within a 50-millisecond range of the ground-truth note event.

Four experiments were conducted: Experiment 1 reports the precision, recall, and f-measure of the polyphonic transcription algorithm on the clean synthesized guitar recordings, considering the accuracy of note pitch and onset time only. Experiment 2 is identical to the first experiment, except that octave errors are ignored. Experiment 3 and 4 are identical to Experiment 1 and 2, respectively, except that the polyphonic transcription algorithm is evaluated on the distortion synthesized guitar recordings in the ground-truth dataset. The results of these experiments are presented in Table 1, alongside the results of the polyphonic transcription algorithm in the 2008 MIREX experiments. In MIREX 2008, the polyphonic transcription

[8] `www.ultimate-guitar.com`
[9] `www.guitar-pro.com`
[10] *Clean* guitar refers to a guitar signal with no audio effects applied.
[11] `github.com/gburlet/musicxml-mei-conversion`

	PRECISION	RECALL	f-MEASURE
Experiment 1	0.71	0.42	0.50
Experiment 2	0.75	0.45	0.53
Experiment 3	0.48	0.36	0.39
Experiment 4	0.56	0.42	0.46
MIREX 2008	0.74	0.78	0.76

Table 1. Results of four experiments evaluating the implemented polyphonic transcription algorithm, alongside the results of the same algorithm in MIREX 2008.

algorithm was evaluated on a relatively small dataset of piano recordings and considered the accuracy of note pitch and onset time.

Analyzing the results, the performance of the polyphonic transcription algorithm on clean guitar recordings is significantly better than on distortion guitar recordings. A possible contributing factor to the reduced transcription performance on distortion guitar recordings could be the modification of the relative amplitudes of harmonics in the frequency domain of the guitar signal caused by the distortion audio effect. In line with past research, the performance of the polyphonic transcription algorithm improves when ignoring octave errors. The polyphonic transcription algorithm receives inferior results on the compiled dataset in comparison to the piano dataset used in the 2008 MIREX. Apart from containing pieces of a different genre, one possible explanation for the degraded performance is that the synthesized guitar recordings in the compiled dataset contain ornamentation such as slides, bends, hammer-ons, hammer-offs, palm-muting, and dead notes. The piano is incapable of replicating many of these ornaments.

5.3 Guitar Tablature Arrangement Evaluation

The implemented guitar tablature arrangement algorithm is evaluated using the compiled ground-truth dataset for guitar tablature arrangement. Since there is no standardized method for evaluating guitar tablature arrangement algorithms, a new evaluation method is used. The evaluation is performed by calculating a fitness value

$$f = \frac{1}{f_{GT}\left(1 + \sum_{i=1}^{N} w_i\right)} \tag{2}$$

for each generated tablature arrangement that is normalized with respect to the fitness of the corresponding ground-truth tablature arrangement f_{GT}. In Equation 2, N is the number of vertices in the optimal path returned by the A* search algorithm and w_i is the weight of the i^{th} edge along the optimal path. The normalized fitness value $f \in \mathbb{R}^+$ is interpreted as the biomechanical ease of performing a generated tablature arrangement relative to the ground-truth tablature arrangement.

By the central limit theorem, it can be assumed that the distribution of normalized fitness values for guitar tablature arranged using the A* search algorithm approximately

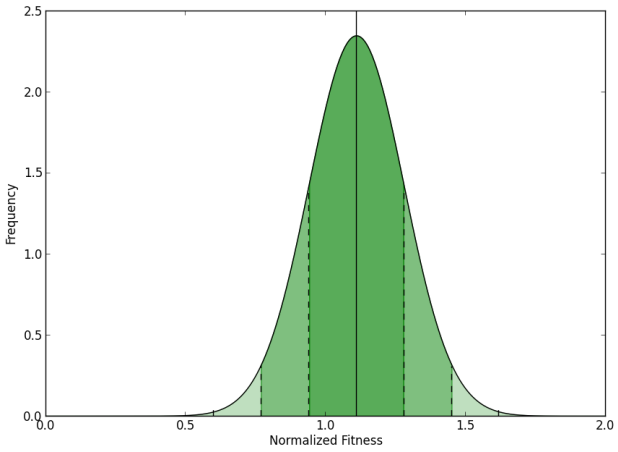


Figure 4. Gaussian distribution $N\left(\hat{\mu} = 1.11, \hat{\sigma} = 0.17\right)$ of normalized fitness values for guitar tablature arranged using the A* search algorithm.

follows a normal distribution. The median $\hat{\mu}$ and standardized median absolute deviation $\hat{\sigma}$ are robust descriptive statistics that are used to describe the central tendency and dispersion, respectively, of this distribution. Robust descriptive statistics are used because they are less sensitive to outliers present in small datasets in comparison to the sample mean and sample standard deviation. The resulting Gaussian distribution $N\left(\hat{\mu} = 1.11, \hat{\sigma} = 0.17\right)$ of normalized fitness values is displayed in Figure 4.

Having a median normalized fitness value of $\hat{\mu} = 1.11$, the A* search algorithm generates guitar tablature arrangements that, on average, are of similar performance difficulty as tablature arranged by humans. By the empirical rule, 68.2% of tablature arrangements generated by the A* search algorithm are estimated to have normalized fitness values that lie within one standard deviation of the mean (0.94–1.28).

6. CONCLUSION

An open-source and web-based guitar tablature transcription framework, entitled *Robotaba*, is designed and implemented. [12] The framework facilitates the rapid development of guitar tablature transcription web applications, providing a vessel for music researchers to publicize their polyphonic transcription and guitar tablature arrangement algorithms, while allowing researchers to focus on algorithm development instead of application development.

As a proof of concept, a guitar tablature transcription web application is developed using the Robotaba framework. An open-source polyphonic transcription application is implemented, which uses the state-of-the-art algorithm proposed by Zhou and Reiss [17]. [13] A new guitar tablature arrangement algorithm, which uses the popular A* pathfinding algorithm, is proposed and implemented as an open-source application. [14]

[12] github.com/gburlet/robotaba
[13] github.com/gburlet/zhoutranscription
[14] github.com/gburlet/astar-guitar

Two ground-truth datasets are compiled from manual tablature transcriptions downloaded from the *Ultimate Guitar* tablature website. The polyphonic guitar transcription dataset consists of 150 synthesized guitar recordings totalling approximately 11 hours of audio, which have been semi-automatically annotated. The guitar tablature arrangement dataset consists of 75 hand-arranged tablature scores encoded in the MEI file format. It is hoped that these datasets will stimulate future research in the area of automatic guitar tablature transcription.

Using the compiled ground-truth datasets, the implemented polyphonic transcription and guitar tablature arrangement algorithms are evaluated. Several experiments illustrate that the polyphonic transcription algorithm exhibits reduced precision and recall on guitar recordings with a distortion audio effect applied. Moreover, in comparison to the quality of transcriptions produced by the polyphonic transcription algorithm on piano recordings in the 2008 MIREX evaluation, the algorithm receives inferior results on the compiled dataset of synthesized guitar recordings. An evaluation of the proposed guitar tablature arrangement algorithm indicates that the algorithm arranges tablature that, on average, is of similar performance difficulty as human-arranged tablature.

Now that a framework exists to combine and evaluate polyphonic transcription and guitar tablature arrangement algorithms, more work can be done to improve the algorithms themselves.

7. ACKNOWLEDGEMENTS

This research was supported by the Social Sciences and Humanities Research Council of Canada. Special thanks are owed to the individuals who have uploaded manual tablature transcriptions to the Ultimate Guitar website, as well as Ruohua Zhou and Joshua Reiss for the open-source polyphonic transcription algorithm used in this research.

8. REFERENCES

[1] Benetos, E., S. Dixon, D. Giannoulis, H. Kirchoff, and A. Klapuri. 2012. Automatic music transcription: Breaking the glass ceiling. In *Proceedings of the International Society for Music Information Retrieval Conference*, Porto, Portugal, 1002–7.

[2] Chang, W., A. W. Su, C. Yeh, A. Roebel, and X. Rodet. 2008. Multiple-F0 tracking based on a high-order HMM model. In *Proceedings of the International Conference on Digital Audio Effects*, Espoo, Finland.

[3] Hart, P., N. Nilsson, and B. Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics* 4 (2): 100–7.

[4] Heijink, H., and R. Meulenbroek. 2002. On the complexity of classical guitar playing: Functional adaptations to task constraints. *Journal of Motor Behavior* 34 (4): 339–51.

[5] Klapuri, A. 2004. Automatic music transcription as we know it today. *Journal of New Music Research* 33 (3): 269–82.

[6] Marolt, M. 2004. A connectionist approach to automatic transcription of polyphonic piano music. *IEEE Transactions on Multimedia* 6 (3): 439–49.

[7] Nichols, E., D. Morris, S. Basu, and C. Raphael. 2009. Relationships between lyrics and melody in popular music. In *Proceedings of the International Society for Music Information Retrieval Conference*, Kobe, Japan, 471–6.

[8] Poliner, G., and D. Ellis. 2007. Improving generalization for polyphonic piano transcription. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 86–9.

[9] Radicioni, D., and V. Lombardo. 2005. Guitar fingering for music performance. In *Proceedings of the International Computer Music Conference*, Barcelona, Spain, 527–30.

[10] Raphael, C. 2002. Automatic transcription of piano music. In *Proceedings of the International Society for Music Information Retrieval Conference*, Paris, France, 1–5.

[11] Roland, P. 2002.The music encoding initiative (MEI). In *Proceedings of the International Conference on Musical Applications using XML*, Milan, Italy, 55–9.

[12] Ryynänen, M., and A. Klapuri. 2005. Polyphonic music transcription using note event modeling. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 319–22.

[13] Sayegh, S. 1989. Fingering for string instruments with the optimum path paradigm. *Computer Music Journal* 13 (3): 76–84.

[14] Smaragdis, P., and J. Brown. 2003. Non-negative matrix factorization for polyphonic music transcription. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 177–80.

[15] Tuohy, D., and W. Potter. 2006. An evolved neural network/HC hybrid for tablature creation in GA-based guitar arranging. In *Proceedings of the Computer Music Conference*, New Orleans, LA, 576–9.

[16] Tuohy, D., and W. Potter. 2006. GA-based music arranging for guitar. In *Proceedings of the IEEE Congress on Evolutionary Computation*, Vancouver, BC, 1065–70.

[17] Zhou, R., and J. Reiss. 2008. A real-time polyphonic music transcription system. In the *Music Information Retrieval Evaluation eXchange*, `www.music-ir.org/mirex/abstracts/2008/F0_zhou.pdf`.

COMPARING ONSET DETECTION & PERCEPTUAL ATTACK TIME

Dr Richard Polfreman
University of Southampton
r.polfreman@soton.ac.uk

ABSTRACT

Accurate performance timing is associated with the perceptual attack time (PAT) of notes, rather than their physical or perceptual onsets (PhOT, POT). Since manual annotation of PAT for analysis is both time-consuming and impractical for real-time applications, automatic transcription is desirable. However, computational methods for onset detection in audio signals are conventionally measured against PhOT or POT data. This paper describes a comparison between PAT and onset detection data to assess whether in some circumstances they are similar enough to be equivalent, or whether additional models for PAT-PhOT difference are always necessary. Eight published onset algorithms, and one commercial system, were tested with five onset types in short monophonic sequences. Ground truth was established by multiple human transcription of the audio for PATs using rhythm adjustment with synchronous presentation, and parameters for each detection algorithm manually adjusted to produce the maximum agreement with the ground truth. Results indicate that for percussive attacks, a number of algorithms produce data close to or within the limits of human agreement and therefore may be substituted for PATs, while for non-percussive sounds corrective measures are necessary to match detector outputs to human estimates.

1. INTRODUCTION AND MOTIVATION

This research forms part of a larger project involving evaluation of controller hardware and parameter mappings in the context of real-time physical modeling synthesis [10]. Thus a specific device (e.g. Microsoft Kinect) will have its control outputs (e.g. performer's 2D hand position) mapped onto synthesis model parameters (e.g. plectrum position in relation to a string). A number of techniques for controller evaluation have been proposed, e.g. [9], including qualitative and quantitative methods. One method of evaluation to be used will ask the performer to match as accurately as possible a given audio target phrase using a given combination of controller, mapping and synthesis configuration. The target and the attempt will then be compared to assess how well the task was completed, in addition to other qualitative assessments. Given that a number of participants, controllers and targets may be used, it would be helpful to complete the performance analysis computationally rather than rely on expert markup of the audio. While in some situations it would be possible to use the timing of control data such as MIDI NoteOn events directly, with perhaps a fixed latency, here the timing of a note or onset may vary significantly for a given control value dependant on other parameters. For example, the position of a plectrum along a string, pluck release threshold, current string displacement and velocity and tension (pitch) will all impact upon the distance from the string the plectrum will need to reach before releasing the string and generating the onset. This indirect control over event timing means that measuring the audio output is necessary. Previous work on onset detection generally does not consider timing accuracy in detail, justifiably prioritising detection rates (type 1 and type 2 errors) and using a temporal tolerance between ground truth and detections beyond which an onset is said to have been missed [3]. Here however, the detailed timing of the onsets is critical.

The measure of two performances being "in time" is a complex issue with a large number of contextual factors, but in this case the target and performance are short monophonic solo instrument phrases with a fixed tempo and it was felt that this case would be simple enough to be studied. More expressive timing feature are ignored and PAT synchronous events are considered the ideal.

2. ONSET TIME

2.1 When is a Note?

Three potential onset times are described in published work. *Physical onset time* (PhOT) is usually considered to be the audio signal first rising from zero, *perceptual onset time* (POT) the time at which a human listener can first detect this change and finally *perceptual attack time* (PAT) is the "perceived moment of rhythmic placement" [15], or rhythmic centre, and is similar to the p-centre concept in speech analysis [13]. A "correct" performance therefore places the events' PATs appropriately, rather than PhOTs or POTs.

While most studies have considered PAT to be a specific time, Wright proposes that PAT is distributed over a finite time period and should be considered as a probability density function describing the likelihood of a listener hearing the PAT at each time point (PAT-pdf) [15]. This

could account for variation between listeners and by individuals in repeated trials and implies that there will be span of time over which an event can remain in musical time with another. This spread of time values is of interest here, since this governs how well-localized the PAT for a particular sound is and how accurately a detection algorithm must match the ground truth.

2.2 PAT Measurement

2.2.1 Measurement Methods

PAT can be measured in a number of ways as identified in [4, 7, 13, 14, 15]. The intrinsic PAT of a sound is typically not measured directly, but rather the delta-PAT (ΔPAT) [7], calculated via comparison against a reference tone. If the reference sound is very short in time, its PAT will be very close to its PhOT and so the target sound PAT can be estimated from the ΔPAT. PAT must be expressed relative to a zero point, usually either the sound's PhOT or the offset from the beginning of the audio file where a sequence is being considered [15].

The most common measurement is rhythm adjustment, where two sounds are aligned by the listener until they either appear synchronous (sound together) or isochronous (sound evenly spaced rhythmically, alternately presented) [14]. Both synchronous and isochronous methods have problems (such as event fusion in synchronous presentation) while isochronous cannot be used where the PATs for a musical sequence are to be measured, rather than isolated events. Likewise Villing's phase correction response (PCR) method [14] is unsuitable for sequences and so the synchronous method was used here.

A tool was created for participants to align a reference sound against a series of test sounds containing a number of onsets (Figure 1). While the reference sound should be short, Wright found that if it is too short there are problems for accurate alignment. He also found that a reference click based on matching the spectrum of the test sound aided PAT alignment [15]. In our experiment, the reference was a simple sine tone, which is the same as the target we will use for the performer to follow, which will include pitch changes at a later stage. Wright gave users control over amplitudes to help avoid fusion of the two events and this was included here. Our tool also allowed the user to change the pitch of the reference, again to help limit fusion ([4] suggests frequency independence of PAT). Gordon [7] indicated that subjects had difficulty matching sounds with very different attack times, and so a user variable attack time was included to ameliorate this, although clearly this has the potential to add uncertainty to the ground truth and so was limited to <127ms.

The participant can choose a sound, select any part of it to be looped and place a marker on the sound that triggers the reference tone. The marker can be dragged with the mouse and fine-tuned by changing the value in a number box, in samples at 44.1kHz sample rate. Thus the location of the reference can be adjusted by ~0.02ms. Participants were instructed to adjust this value until the test event and reference sounded musically synchronous. The visual display is to aid users in finding physical onsets quickly before searching those regions for perceptual alignment. For each event the tool recorded the PAT and the other user settings so that these could also be analysed if necessary. Participants were each given a training session (in addition to a written manual) and asked to complete the task using headphones.

Figure 1. Software tool for ground truth collection.

2.2.2 Test Sounds

Five test sounds were used, four were synthesized with the IRCAM's Modalys software [6] and the performances made deliberately imperfect, so that each event in the sequence would not be identical and the timing of events not strictly metrical. The sequences provide a set of variations in timbre and attacks as one might expect in an instrumental performance. The dynamics were generally stable but with occasional deviations. The models were: plucked string (un-damped), legato bowed string, struck plate (un-damped) and a single reed-tube. Only in the reed sound was complete silence reached between onsets and not for all of those. The final sound was a sine tone, which is used as the target for performance matching, in this case precisely metrical. These beeps were 95ms long (5ms attack, 90ms decay) with a 500ms inter-PhOT interval. Each sound was normalized, had a fundamental frequency of 130.81Hz (an octave below middle C) and contained 16 onsets, providing 80 events in total.

2.2.3 Ground Truth Results

Nine participants completed the task for all 80 events and so each sound file had 144 marked-up onsets and there were 720 data points in total. All participants had some musical experience, typically in ensembles or bands and/or formal performance training. Where data seemed particularly erroneous, such as a missing or duplicated event, or in isolation extremely different to others , participants were asked to review and double-check their data to ensure they were content with the values originally supplied, and, only if not, amend them. As with other studies, participants reported that the task was challenging, particularly with the non-percussive sounds, while one reported that (in the reed case) there were a range of time values over which the reference and test sound were

equally "in time", and that they had simply tried to be consistent in where they placed the reference sound.

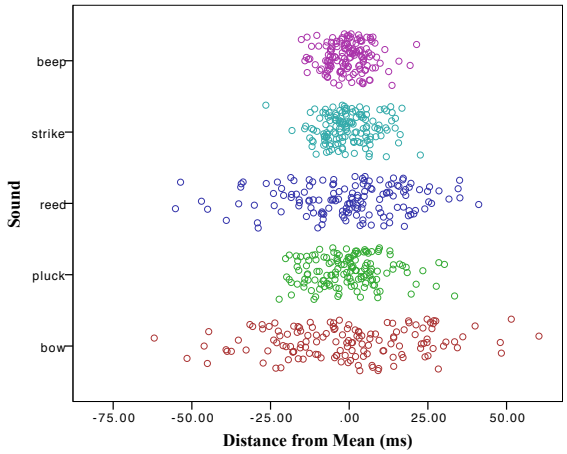


Figure 2. Scatterplot of ΔPAT ground truth data.

To group the results of ΔPAT values across different events within a particular sequence, the mean ΔPAT value was taken for each event and then each ΔPAT value replaced by its distance from that mean. Figure 2 shows scatter plots of these mean-shifted ΔPAT values (with vertical jitter to improve visibility). As expected, shorter attacks gave rise to more tightly clustered ΔPAT times, although outliers remain, while the longer attacks produce more widely spread results, as the location of the note is more ambiguous. We also expect smaller variation in the beep sounds since each event is almost identical, differing only in the phase of the sine in each. The plucks show greater spread than the other percussive attacks, again expected due to the more complex articulation: a double attack of the initial plectrum impact on the string followed rapidly by the release of the string creating the note (Figure 3). The time between impact and release was typically between 20ms and 40ms, averaging 23ms. The audio files were also annotated for PhOT for comparison with ΔPAT and onset detector times. For the percussive attacks this was straightforward as in each case there were discontinuities in the signal at the point where each new event began and which could be found through visual inspection. In the case of the pluck sounds, both the impact and string release times were noted. For the reed sound, onsets starting from silence were similarly clear, while others were estimated from the inflection point in amplitude between the decay of one note to the beginning of the next. The bow sound was particularly difficult and required inspection of the sonogram in addition to the time domain signal and PhOT was estimated from disruption to the harmonic structure as one event ends and the other begins. Table 1 shows the mean and standard deviation offsets from ΔPAT to PhOT for each sound, where the pluck sound is using the string release time. All apart from pluck are positive values as expected, where ΔPAT is later than PhOT. As can be seen from the table, ΔPAT appears very close to PhOT for the short attacks, although with some variation as reflected in Figures 2 and 4. Interestingly pluck is very close to the string release point, in fact slightly earlier, suggesting an effect of the preceding impact bringing the PAT forward. Given the close agreement between mean ΔPAT and PhOT for the short attacks, this is indicates that onset detectors which measure PhOT should provide timing data close to ΔPAT. For the non-percussive attacks ΔPAT is significantly later than PhOT, and so the utility of onset detectors will depend on whether they remain close to PhOT or are similarly delayed.

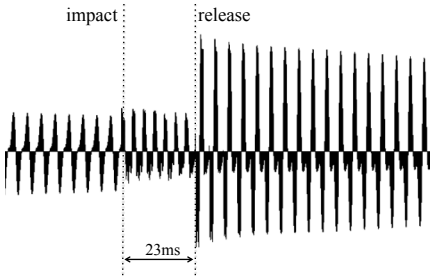


Figure 3. Section of pluck waveform showing decay of previous event followed by initial plectrum impact and release of string.

Sound	*Bow*	*Pluck*	*Reed*	*Strike*	*Beep*
Mean (ms)	23.52	-0.46	51.21	3.48	2.08
σ (ms)	22.83	10.64	18.46	7.42	5.91

Table 1. Mean and standard deviation for ΔPAT-PhOT distance.

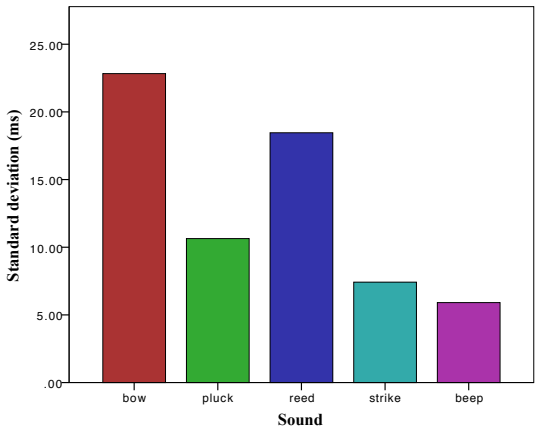


Figure 4. Mean ΔPAT standard deviations.

Figure 4 shows the σ across events for ΔPAT, indicating how consistently human listeners can determine ΔPAT for each sound (against the reference tone). Thus for bowed sounds, ±σ gives a spread of ~42ms and for the sine beep ~11.0ms. While only bow and pluck passed Shapiro Wilks normality tests, over 70% of data for each sound were within ±σ of the mean. The limit of discrimination of temporal events is typically considered to be ~10ms [4]. Wright logically proposed a system for automatic mark-up of audio using onset detection followed by

a PAT model to correct for the difference between PhOT and PAT [15]. However, if the time differences between the ground truth and onset times reported by onset detectors are within similar limits to human listeners it indicates that these may be used directly to provide PAT data without adding a specific PAT-PhOT model.

3. ONSET DETECTION

3.1 Onset Detection Algorithms

Onset detection algorithms are typically based on PhOT or POT, with a time tolerance to decide successful detections. The task usually comprises three main steps: (optional) pre-processing; generation of an onset detection function (ODF) that indicates the probability of an onset at each moment in time; and peak selection across the ODF. While some methods are psychoacoustically motivated, differences between PhOT, POT and PAT are usually ignored. Here those differences are important if an onset detector is to provide ΔPAT estimates.

Several comparative studies of the performance of onset detection algorithms have been published, while the MIREX event compares a number of new algorithms annually. Studies, including [1, 3, 5], compare the rates of false positives and false negatives against a selection of test sounds. Collins [3] compared 16 onset detection algorithms with NPP (non-pitched percussive) and PNP (pitched non-percussive) monophonic sounds, finding that for the NPP case, a spectral difference function based on work by Klapuri [8] was most effective, while for the PNP case all algorithms performed less well, with a phase deviation method being the most successful [1]. While comparing algorithms against PhOT rather than PAT, Collins used detection tolerances of 50ms for PNP sounds and 25ms for NPP, which compare well with the figures shown in Figure 4 [3].

3.2 Onset Measurement

3.2.1 Implementation

A Max patch was developed to run a number of onset detection algorithms against the test audio. This displays the ODF for each detector as well as the detection hits. Initially 8 algorithms were tested, including two widely known Max objects *bonk~* and *sigmund~*, both later rejected as unable to provide sufficiently accurate results. To compare results with a commercial onset detection system, the audio software Melodyne 3.2[1] was also included in the experiment. For each sound Melodyne's percussive mode detection was used, as this outperformed the other options, even on non-percussive sounds, and it should be noted that no detection parameters were user adjusted in this case.

A modified[2] version of the *aubioonset~* MSP object by Andrew Robertson [11], itself a port of algorithms implemented by Paul Brossier [2] was used for high frequency content (HFC), energy based, modified Kullback-Leibler (MKL), complex, spectral difference (SD) and phase deviation (PD) functions, the equations for which can be found in the literature [2]. In each case the FFT size was 2048 with a hop of 128 samples. While there are more recent algorithms, these were chosen as being widely available and frequently referred to in the literature as the basis for other algorithms or tests. Due to difficulties with non-percussive attacks, two adaptations were implemented as Max patches – weighted phase deviation (WPD) and spectral flux (SF) following Dixon [5], the latter rectifying the difference between frames in SD, important in distinguishing between onsets and offsets. Peak-picking for WPD and SF involved taking the difference between the outputs of two moving average filters (using *average~*) and passing the result to a Schmitt trigger (*thresh~*). One filter was coarse providing an adaptive threshold (averaging over ~130ms), the other fine to smooth the ODF (typically ~20ms).

3.2.2 Comparison with Ground Truth

Detection function parameters were adjusted to achieve as close as possible to 100% success rate, i.e. 0 false positives (FP) or false negatives (FN). This was achieved for all the percussive attacks with all detectors, but the reed and bow sounds proved more problematic. Figure 5 shows the mean distance of each algorithm from the ground truth for the percussive attacks. The error bars indicate one standard deviation above and below the mean, the first set being those of the ground truth.

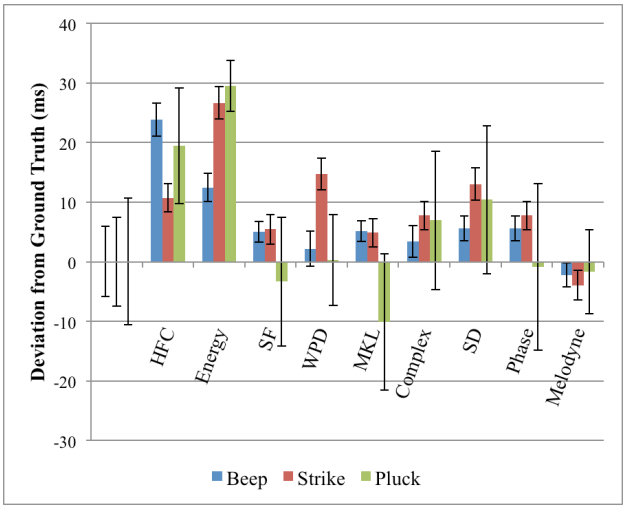


Figure 5. Mean distance from ground truth for each algorithm, percussive attacks (positive values are later).

As can be seen in the figure, HFC and Energy are typically late detectors, and fall outside the standard deviation (σ) ranges for the ground truth, while Melodyne performed very well across all three percussive attacks, preempting the ΔPAT values (PhOT earlier than PAT) as expected. In fact comparison with the PhOT data shows that Melodyne very accurately tracked PhOT (e.g. -0.1ms average distance for the beep), performing slightly worse

[1] http://www.celemony.com/

[2] Modified to include audio rate output of the onset detections as 1-sample delta functions, rather than Max *bangs*, to improve timing accuracy.

with pluck as it marked the impact time rather than string release for two events. Strike and beep across all the detectors show relatively consistent offsets from the ground truth, albeit varying by sound and detector, the σ values are all < 3ms. For pluck, HFC, Energy, WPD and Melodyne achieved a σ less than the ground truth.

To decide whether an onset detector can be used as a ΔPAT measurement tool we must define how closely the outputs of the detector must correspond to the ground truth. Table 2 shows a summary of each detector against each percussive attack for three simple tests. The first test (a) is simply whether the standard deviation of the detector output is less than that of the ground truth – not in itself sufficient, but indicative of relative stability. The second (b) and third (c) state whether combinations of the detector mean and standard deviation lie within limits of the 1 or 2 standard deviations of the ground truth:

$$(|\mu_D| + \sigma_D) < \sigma_{GT} \quad (1)$$

$$(|\mu_D| + (2 \times \sigma_D)) < (2 \times \sigma_{GT}) \quad (2)$$

where μ_D is the detector mean deviation from ground truth, σ_D and σ_{GT} the detector and ground truth standard deviations respectively. Equation 1 implies (for normal distributions) that we expect ~64% of detector values to lie within one standard deviation of the ground truth mean, changing to ~95% of detector outputs within two standard deviations of ground truth mean in equation 2.

Sound	*Beep*			*Strike*			*Pluck*		
Detector	*a*	*b*	*c*	*a*	*b*	*c*	*a*	*b*	*c*
HFC	Y	-	-	Y	-	-	Y	-	-
Energy	Y	-	-	Y	-	-	Y	-	-
SF	Y	-	Y	Y	-	Y	-	-	-
WPD	Y	Y	Y	Y	-	-	Y	Y	Y
MKL	Y	-	Y	Y	Y	Y	-	-	-
Complex	Y	-	Y	Y	-	Y	-	-	-
SD	Y	-	Y	Y	-	-	-	-	-
Phase	Y	-	Y	Y	-	Y	-	-	-
Melodyne	Y	Y	Y	Y	Y	Y	Y	Y	Y

Table 2. Onset detector summary for percussive attacks.

As can be seen in Table 2, Melodyne passed each test for each percussive attack, indicating that it is likely to provide a useful equivalent to ΔPAT data, while WPD is effective for beep and pluck, and MKL for strike.

With the reed sounds several algorithms conflated onset and offsets, with Complex, SD, MKL and PD often showing stronger peaks, sometimes double peaks, on offsets in the detection function (see Figure 6), making their FN rate unacceptably large[1]. Melodyne also suffered from offset conflation with the reed sound, although the detection function could not be examined. As expected, the half-wave rectification introduced in the SF algorithm eliminated this problem, with offset peaks significantly lowered, as did WPD by shaping the response by amplitude. The remaining detectors were able to provide zero FN and FP rates, and for HFC, SF and WPD with σ less than the ground truth. All means were delayed with respect to the ΔPAT ground truth and none were contained within ±σ of the ground truth (see Figure 7).

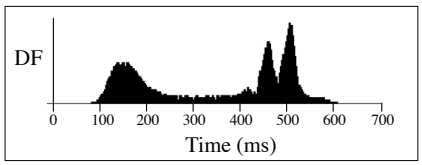


Figure 6. *Complex* onset detection function over the duration of a single reed event, showing large offset peaks.

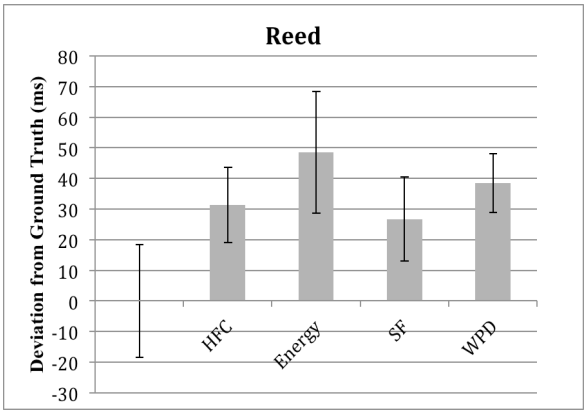


Figure 7. Mean distance from ground truth for selected algorithms, reed sound.

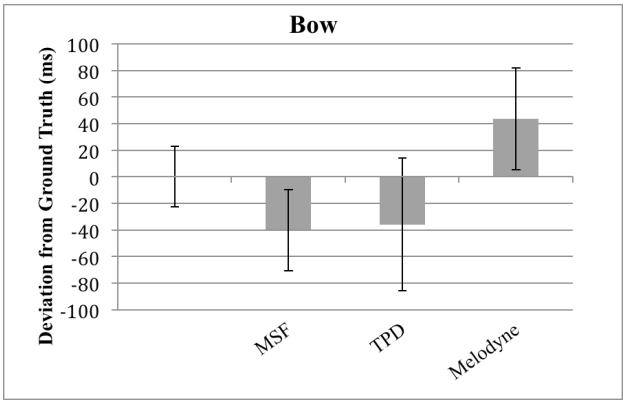


Figure 8. Mean distance from ground truth for selected algorithms, bow sound.

The bowed sound was most problematic (Figure 8), with only Melodyne achieving 0 FN and FP rates, while others (e.g. HFC) resulted in an FP rate of ~33% if adjusted to zero FN rates. SF had one FP with rectification off - i.e. as SD but with the dual-filter peak picker (labeled MSF). MKL only identified a single onset, while WPD achieved zero FN and FP, when rather than weighting each phase contribution by the magnitude from the FFT frequency

[1] Testing with a single clarinet sample indicated that these algorithms suffer offset conflation there also, rather than this being a product of the physical model synthesis.

bins, a threshold was used (TPD). None of the three best detectors were within the ground truth range or had σ lower than the ground truth.

4. DISCUSSION AND FUTURE WORK

The aim of this work was to assess whether automatic onset detection methods might be used to provide metrics for measuring performance accuracy, where the phrases to be assessed would be monophonic but the sounds potentially complex. This required testing since performance timing is considered to be PAT based, while onset detection PhOT or POT based. Further, the reported performance of onset detectors is often reduced to type I and II errors, rather than distances from ground truth.

Ground truth data captured via rhythm adjustment with synchronous presentation indicated levels of agreement of approximately 12-20ms for percussive attacks and 42ms for non-percussive (within a single standard deviation). Given the likelihood that there is indeed a span of time offset over which two sounds may be said to remain in time rhythmically, it would seem useful to develop a new method of ΔPAT measurement that does not force the participant to select a single time value, but rather supports identification of a range. Such a method could make the task easier for participants, speeding up the annotation process and increasing accuracy.

All of the onset detectors managed zero type I and type II errors with the percussive attacks, but only some produced results close enough to the ground truth to be regarded as PAT equivalent data. For the non-percussive sounds, achieving 100% detection even in these short sequences proved challenging, and the timing did not match the ground truth closely enough, requiring some form of PAT model to correct for this. Future work should investigate existing models, such as those tested in [4].

The algorithms used are well known and therefore results may usefully be compared with other studies, but it would be helpful to test more recent algorithms for performance improvements. Recent work has explored the influence of peak-picking algorithms on the performance of onset detection and it would be useful to test alternative methods in this context, particularly as the temporal location of the peak is so critical here [12]. Similarly, preprocessing could be explored.

It would be useful if the MIREX onset detection test data were additionally annotated for PAT so that algorithms could be assessed against PAT as well as PhoT/POT data and against a large data set.

5. REFERENCES

[1] J. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler: "A Tutorial on Onset Detection in Music Signals", *IEEE Trans. On Speech And Audio Proc.,* Vol 13, No 5, pp.1035-1047, 2005.

[2] P. Brossier: *Automatic Annotation of Musical Audio for Interactive Applications*, Ph.D. Thesis, Queen Mary University of London, UK, 2006.

[3] N. Collins: "A Comparison of Sound Onset Detection Algorithms with Emphasis on Psychoacoustically Motivated Detection Functions", *Proceedings of AES118 Convention*, 2005.

[4] N. Collins: "Investigating computational models of perceptual attack time", *Proceedings of the 9th International Conference on Music Perception & Cognition (ICMPC9)*, pp. 923-929, 2006.

[5] S. Dixon: "Onset Detection Revisited", *Proceedings of the 9th Int. Conference on Digital Audio Effects (DAFx'06)*, pp. 133-137, 2006.

[6] N. Ellis, J. Bensoam and R. Caussé: "Modalys Demonstration", *Proceedings of the International Computer Music Conference (ICMC),* 2005.

[7] J. W. Gordon: "The perceptual attack time of musical tones" *J. Acoust. Soc. Am.,* Vol. 82, No. 1, pp. 88–105, 1987.

[8] A. Klapuri: "Sound onset detection by applying psychoacoustic knowledge", *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Proc. (ICASSP),* pp. 3089–92, 1999.

[9] S. O'Modhrain: "A Framework for the Evaluation of Digital Musical Instruments", *Computer Music Journal*, Vol. 35, No. 1, pp. 28–42, 2011.

[10] R. Polfreman: "Multi-Modal Instrument: Towards a Platform for Comparative Controller Evaluation", *Proceedings of the International Computer Music Conference (ICMC),* pp. 147-150, 2011.

[11] A. Robinson: "Queen Mary University of London: Andrew Robertson – Software", http://www.eecs.qmul.ac.uk/~andrewr/software.htm, accessed July 2013.

[12] C. Rosão, R. Ribeiro, and D. Martin de Matos: "Influence Of Peak Selection Methods On Onset Detection", *Proceedings of the 13th International Society for Music Information Retrieval Conference* (ISMIR 2012), pp. 517-12, 2012.

[13] S. Scott: "The point of p-centres", *Psychological Research*, Vol. 61, pp. 4–11, 1998.

[14] R. Villing: *Hearing the Moment: Measures and Models of the Perceptual Centre*, Ph.D. Thesis, National University of Ireland Maynooth, 2010.

[15] M. Wright: *The Shape Of An Instant: Measuring And Modeling Perceptual Attack Time With Probability Density Functions,* Ph.D. Thesis, Stanford University, Stanford, CA, 2008.

K-POP GENRES: A CROSS-CULTURAL EXPLORATION

Jin Ha Lee
University of Washington
jinhalee@uw.edu

Kahyun Choi
University of Illinois
ckahyu2@illinois.edu

Xiao Hu
University of Hong Kong
xiaoxhu@hku.hk

J. Stephen Downie
University of Illinois
jdownie@illinois.edu

ABSTRACT

Current music genre research tends to focus heavily on classical and popular music from Western cultures. Few studies discuss the particular challenges and issues related to non-Western music. The objective of this study is to improve our understanding of how genres are used and perceived in different cultures. In particular, this study attempts to fill gaps in our understanding by examining K-pop music genres used in Korea and comparing them with genres used in North America. We provide background information on K-pop genres by analyzing 602 genre-related labels collected from eight major music distribution websites in Korea. In addition, we report upon a user study in which American and Korean users annotated genre information for 1894 K-pop songs in order to understand how their perceptions might differ or agree. The results show higher consistency among Korean users than American users demonstrated by the difference in Fleiss' Kappa values and proportion of agreed genre labels. Asymmetric disagreements between Americans and Koreans on specific genres reveal some interesting differences in the perception of genres. Our findings provide some insights into challenges developers may face in creating global music services.

1. INTRODUCTION

The overemphasis on Western music and context has been a long-standing issue in the Music Information Retrieval (MIR) domain. To date, there are only a small number of studies that deal with the organization of, and access to, non-Western music. This is a critical issue considering the trend of increasing global distribution and appreciation of music [10].

There are a wide variety of different types of metadata which can describe music, but music genres in particular are considered one of the primary methods for organizing and retrieving music ([13], [4]). However, we currently have a limited understanding of the genres of popular music in non-Western cultures. What kinds of genres are used in these cultures and how similar or different are they to genres used in Western cultures? What kinds of issues or challenges exist in categorizing non-Western popular music by genre? How are the genres used in music-related resources? How are they perceived by average music users from Western vs. non-Western culture?

This work attempts to bridge the gap in our knowledge of music genre in non-Western cultures by analyzing the genres used for K-pop (Korean Pop), and explore how K-pop genres are perceived by users cross-culturally. Korea is a particularly interesting case in which to study cross-cultural issues in music genre, as the country was heavily influenced by American pop culture from the 1950s through the early 21st century, but is now exporting cultural objects and music which are appreciated by foreigners and actively sought by people outside of Korea [11]. This suggests that a lot of cross-cultural music seeking is happening in this space from both directions. Genre will be extremely important for those users as it can serve as useful metadata to discover new music they want to listen to. The main objectives of our study are: 1) to improve our understanding of how genre labels are used in the Korean context and, 2) to explore cross-cultural perception of K-pop genres. By doing so, we hope to obtain insights into what kinds of challenges we may encounter when we start building a music collection targeted for a global user base where cross-cultural music seeking is unavoidable.

2. BACKGROUND AND PRIOR WORK

Despite the increasing number of user studies in the MIR domain, there are still only a handful of cross-cultural studies that investigated issues in music information needs, seeking, organization, management, and consumption. One such study on non-Western music, in particular Korean music, was [5]. Lee et al. [5] collected music queries from Google Answers and Naver Knowledge-iN (지식 iN), and did a comparative analysis. They identified several challenges that Korean users experienced when trying to find Western music, including: 1) common failures in providing traditional bibliographic information such as title or name of the performer(s), 2) difficulty in understanding and using Western genre labels, and 3) difficulty in using lyrics as they often consist of common words and lack discriminating power. The findings support the necessity of establishing new access points for accommodating cross-cultural music searching such as associative metadata (e.g., source of music).

The issue of genre is also raised in [6] by McEnnis and Cunningham as they discussed how music can be "interpreted in terms of how it expresses local issues and concerns, often quite removed from the circumstances that inspired the music's creation." They conclude that attempting to define universal meanings for music across cultural boundaries is destined to fail. This strongly supports the need to investigate how these genres are actually used in different cultures in order to provide better contextual information to potential users.

Norowi et al. [8] and Doraisamy et al. [1] investigated using MARSYAS for automatic genre classification given a mix of Western and traditional Malay genres and demonstrated that it is possible to classify non-Western music with such systems. However, Doraisamy et al. [1] did note that adapting digital music library systems for the retrieval of Malaysian music was challenging due to differences in genres and musical structures.

One theme emerging from a number of these studies is the issue of music genre. Genre is the primary means by which listeners search and browse music [4], [13], yet studies on genre classifications show that there is hardly any convergence as demonstrated in this quote from [9]:

"Easy listening" in one classification is called "Variety" in another and worse, taxonomy structures do not match: "Rock" for instance denotes different songs in different classifications.

Navigating through these genre classification systems is very likely to get more complicated as we move across different cultures. This is already happening quite frequently with the emergence of new tools and technologies for music distribution targeted for global music markets and users. For instance, K-pop which used to be a relatively obscure music category outside of Korea is now appreciated in numerous countries around the world due to the fact that users can easily access K-pop songs through media like YouTube[1]. As users get exposed to music from unfamiliar cultures and in unknown languages, it will become increasingly important to provide them with intelligent access to music resources, and despite its flaws, genre is still a widely used descriptor for organizing and accessing music.

3. STUDY DESIGN

3.1 Analysis of Korean Genre Labels

In order to understand the different types of Korean music genres, we collected several hundred genre labels from the following list of popular Korean commercial websites (sorted alphabetically) for music distribution.

- 벅스뮤직 (Bugs Music): http://www.bugs.co.kr/
- 싸이뮤직 (CyMusic): http://music.cyworld.com/
- 다음뮤직 (Daum Music): http://music.daum.net/
- 엠넷 (M.net): http://www.mnet.com/
- 멜론 (Melon): http://www.melon.com
- 네이버뮤직 (Naver Music): http://music.naver.com/
- 올레뮤직 (Olleh Music): http://www.ollehmusic.com/
- 소리바다 (Soribada): http://www.soribada.com/

We examined the different types, organizational structures, and examples of music that are categorized into each genre category used in these websites.

3.2 User Annotation of K-pop Music

Three American and three Korean annotators were hired to provide genre labels for the 1894 songs found in a collection of K-Pop music collected as part of a new K-MIREX (Music Information Retrieval Evaluation eXchange) data set. There is always a resource tradeoff between the number of annotators and the number of songs to be annotated. Choosing six annotators is consistent with established practice in the inter-indexer consistency literature [6] while allowing us to collect labels on a wide variety of songs. The annotators generated a total set of 11,352 song-label pairs. Annotators chose from eight genre labels: *Ballad*, *Dance/Electronic*, *Folk*, *Hip-hop/Rap*, *Rock*, *R&B/Soul*, *Trot*, and *Other*. These genres were selected based on the analysis of genre labels used in Korean music websites (Further discussed in Section 4.1). A brief training manual providing short descriptions of each genre along with exemplar audio samples was created to train the annotators. This manual was prepared in both Korean and English.

4. DATA AND DISCUSSION

4.1 Overview of the Korean Genre Labels

From eight major Korean music distribution websites, we collected all the genre labels used for organizing their music. 307 unique labels were distilled from the aggregate collection of 602 labels after removing duplicates (including different transliterations of the same term). We present the top-level genre categories used in Table 1. A solid dot in Table 1 represents that the specific genre category was used on the website. An open dot signifies that a genre was used in conjunction with another genre (e.g., Jazz/Blues). The genres are ordered by the frequency of occurrences across multiple websites and only ones used in more than one website are presented in Table 1. The genre labels that only appeared once at the top level include: 7080, Music used in commercials, Prenatal education, Trot (Bugs); Chinese music (Melon); Country/Folk, Easy listening, Punk; Reggae (M.net); Funk (Soribada); and Rap (Cy).

Many of the genres used in Korean Websites were the same or very similar (e.g., Classic for Classical music) to the ones used in North America, most likely due to the strong American influence on K-pop music. For instance, genres such as Hip-hop, Rap, and R&B were introduced to South Korea by Korean American musicians or students who studied abroad in the US in the mid-1990s, and they continue to act as leading voices in those genres today [10]. In addition to this cultural proximity created by influences of popular culture or religion, the regional proximity also plays a role in which genres are prominently represented in a particular culture. This is evidenced by J-pop which was used in 7 out of 8 websites as a top-level category. There are other examples: for instance, on All Music Guide (http://www.allmusic.com) which is based in United States, Latin is one of the top categories although it did not appear anywhere in the top categories of Korean websites. In some sense, it is natural for users to gravitate toward music that is from places near them since there is a better chance for them to be exposed to such music.

[1] YouTube launches exclusive K-Pop Channel. Dec, 2011. (http://www.soompi.com/news/youtube-launches-exclusive-kpop-channel)

	Bugs	Cy	Daum	M.net	Melon	Naver	Olleh	Soribada
OST/O.S.T./Soundtrack	●	●	●	●	●		●	●
J-POP/Japanese music	●	●	●	●	●		●	●
CCM	●	●			●		●	●
Jazz	●	●	○ blues	○ blues	●		●	●
Gayo (가요)/K-pop	●		●	●	●		●	●
Pop/Pop song	●		●	○ soul	●		●	●
Classic (클래식)/Classical	●	●	○ new age	●	●			●
New Age	●	●	○ classic		●			●
Gukak (국악)	●			●	●		●	
Religious music			●	●	●		●	
World music			●	●	●			●
Hip-hop	○ R&B	○ rap		●				●
Dongyo (동요)/Children's	●			●	●			
Electronic/Electronica		●		●				●
Rock		●		○ metal				●
R&B	○ hip-hop	○ soul						●
Indie	●							●
Korean music (국내음악)		●				●		
Foreign music (국외음악)		●				●		
Metal				○ rock				●
Soul		○ R&B		○ pop				
Blues			○ jazz	○ jazz				
Other/Etc.			●			●	●	●

Table 1. Comparison of the top-level genres of eight Korean music websites

Another interesting observation of these data is that OST (Original Soundtrack) is the most commonly used top-level genre label. OST is used to refer to songs that were used in TV shows, dramas, movies, etc. The prevalence of this label confirms the findings in [4] that information on "associated use" (e.g., movie, ad) was the most commonly used feature in music identification queries. In Korea, it is fairly common for new or relatively unknown artists to become extremely popular "overnight" due to the exposure of their songs in TV commercials or dramas (e.g. *Americano* by 10cm; *Honey Honey Baby* by Yozoh).

Broad genres encompassing a wide range of Korean music styles were present in all websites. The *Gayo* (가요) (the term referring to all Korean popular music, sometimes used interchangeably with K-pop) and Korean music (국내 음악) genres contained subdivisons in five of the websites. Table 2 shows the subdivisions of the "*Gayo*/ Korean Music" category from the five websites. The most commonly used sub-genres are Ballad, Dance, Hip-hop, and Rock. The sub-genres that appeared only once include: Club (Bugs); Jazz/Blues, Idol, 7080 (Daum); Urban, and Rap (M.net).

	Bugs	Daum	M.net	Melon	Naver
Ballad	○ R&B	●	●	●	●
Dance	○ club	●	●	●	●
Hip-hop	●	●	○ rap	○ rap	●
Rock	●	●	●	●	○ folk
Trot		●	●		●
Folk	●	●	●		
R&B	○ ballad	○ soul	○ urban		
Electronic(a)		●	●	●	
Indie		●	●		

Table 2. Subdivision of K-pop in 5 Korean websites

Based on our analysis, we decided to select the most commonly used eight genre labels for our annotation experiment: *Ballad, Dance/Electronic, Folk, Hip-hop/Rap, Rock, R&B/Soul, Trot,* and *Other*. We grouped two genres into one label when the boundaries seemed fuzzy. This was evidenced by a number of songs being categorized into both genres across multiple websites.

4.2 Results of the Annotation Experiment

4.2.1 Agreements among Users

Fleiss' kappa is a standard measure of inter-rater reliability when there are more than two annotators, and when the rated variable is categorical [2]. As there are three annotators in each cultural group and the genre annotations are categorical, we used Fleiss' kappa to measure the inter-rater reliability among the annotators. The results showed that the Koreans reached a higher agreement level ($\kappa = 0.664$) than the Americans ($\kappa = 0.413$). According to [2], these κ values indicate substantial and moderate agreement, respectively. The agreement level among all six annotators was also moderate ($\kappa = 0.477$). This indicates that while there exists a common understanding of K-Pop genres across cultural boundaries, it is more consistent among Korean listeners than American listeners. While this result is not unexpected as the music stimuli were K-Pop and the genre labels were developed from analyzing Korean websites, it indeed demonstrates the challenge in designing cross-cultural MIR systems using genre as access points to serve users from different cultural backgrounds.

Figure 1 illustrates the number of songs that received three, two, or zero "agreed" (i.e., identical) labels within the Korean and American annotator groups. Among the Koreans, 63% (1189/1894) songs scored three agreed la-

bels as opposed to 36% (688/1894) for the Americans. If we loosen the agreement criterion to majority vote (at least two annotators agree), Korean consistency was 96% (1820/1894) while American consistency was 87% (1654/1894). As there are eight genre choices for each song, the probabilities for annotators in each group to reach agreement by chance or idiosyncratic ratings are extremely low: $(1/8)^3 = 0.19\%$ for unanimous agreement and $(1/8)^2 = 1.56\%$ for agreement by two. Therefore, comparing songs with genre labels agreed by even a small number of annotators can still show the differences influenced by the cultural background of the annotators.

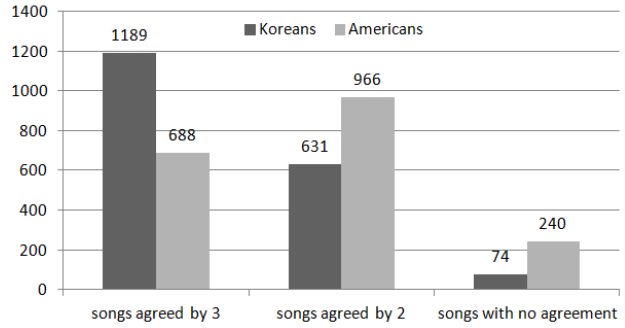


Figure 1. Number of songs that received three, two, or zero agreed labels

To investigate possible cultural differences between the Koreans and Americans, we looked at the distribution of songs with unanimous labeling within each annotator group. Table 3 presents the distribution of genre labels for the overlap of 505 songs unanimously labeled within each group. A chi-square test of independence on these distributions revealed that the culture factor and genre factor were independent ($\chi^2 = 2.16$, df = 6, $p = 0.90$). This indicates that cultural background did not affect genre judgments for the songs with unanimous agreement.

	Ballad	Dance	Folk	Hip-hop	R&B	Rock	Trot	Total
KO	236	93	4	25	63	53	31	505
AM	227	87	3	24	78	54	32	505

Table 3. Distribution of songs with unanimous labels

However, when we consider songs with two or three agreed labels, the results are different. With this relaxed criterion, we can find the overlap of 1,602 songs between the Koreans and Americans. The distribution across genres is shown in Table 4. A chi-square test of independence on this distribution indicates that culture and genre factors were *not* independent ($\chi^2 = 56.70$, df = 6, $p < 0.001$). In other words, the judgments on genre were related to users' cultural background.

The discrepancy on these two test results suggests that some songs are "exemplary" songs that have strong musical characteristics representing a particular genre. These songs will receive unanimous judgments regardless of cultural background. However, for songs with genres that are less obvious, cross-cultural difference is significant. In other words, cultural background *does* seem to affect how different judgments are made. This is informative for designing cross-cultural MIR systems as in most cases genres are not highly agreed upon even within a single culture, due to the issues of variations in definition and/or hybrid genres (further discussed in Section 4.3).

	Ballad	Dance	Folk	Hip-hop	R&B	Rock	Trot	Total
KO	759	257	60	59	153	174	140	1602
AM	642	208	57	56	281	212	146	1602

Table 4. Distribution of songs with two or three agreed labels

4.2.2 Disagreement between Genres

We also examined which genres showed the highest cross-cultural differences. As there was a statistically significant relationship between culture and genre when considering the 1,602 songs with at least two agreed labels, we now examine the disagreement between each pair of genres on this data set. In Table 5, each cell shows the number of songs labeled as one genre by Koreans (column) and another by Americans (row). The cells on the diagonal (highlighted cells) are numbers of songs agreed by the two groups, while other cells represent the disagreement between the two groups. The matrix shows asymmetric disagreements between Americans and Koreans on many genres.

The bolded numbers in Table 5 indicate major disagreements between Korean and American annotators. The largest discrepancy was the 121 songs labeled as Ballad by Koreans but R&B by Americans. Considering the two groups only agreed on a total of 122 R&B songs, this discrepancy is very substantial. Also note that of the 57 songs labeled as Folk by Americans, 18 songs were labeled as Trot by Koreans. This may be due to the disagreement resulted by same genre labels referring to different types of music in Western vs. Korean cultures. Historically, Ballad and Folk have evolved somewhat differently in K-pop culture than the Western culture and both genres are closely associated with music from a particular time period (i.e., Ballad from the 80s-90s, and Folk from the 60s-80s). Therefore a song that sounds like Ballad but is from the 00s may not be categorized as Ballad by Koreans [see Section 4.3.2 for more discussion].

We also observed that among the 212 Rock songs annotated by Americans, 42 (18.92%) were labeled as Ballad by Koreans. This disagreement might be attributed to the hybrid genre Rock Ballad. Similarly, the disagreement between Ballad and Folk may be attributed to the hybrid genre Folk Ballad.

AM \ KO	Ballad	Dance	Folk	Hip-hop	R&B	Rock	Trot	Total
Ballad	561	1	13	0	**26**	**25**	**16**	642
Dance	7	184	1	6	2	4	4	208
Folk	**10**	0	27	0	0	2	**18**	57
Hiphop	0	**15**	0	39	1	1	0	56
R&B	**121**	**20**	1	8	122	7	2	281
Rock	**42**	**25**	6	6	0	128	5	212
Trot	**18**	12	12	0	2	7	95	146
Total	759	257	60	59	153	174	140	1602

Table 5. Cross-tabulation between genre annotations (majority votes) by Korean and American users

In summary, it is easier for Korean annotators to reach an agreement on the genres of K-Pop music than Americans. Difference on genre definitions and historical context in the two cultures might have caused the discrepancies between Koreans and Americans, especially for genres such as Ballad and Folk. Hybrid genres might also be a reason for disagreement. The comparison between annotations from two groups confirmed that it is challenging to identify genres across cultural boundaries.

4.3 Discussions

4.3.1 Variations in Genre Criteria

One reason why genre labels can be difficult to interpret, understand, or compare is because they are based on a range of different criteria. Among the genres we examined, the most common defining characteristic was indeed musical style. However, this is not always the case. We found that the following other dimensions were frequently used to define genres:

- **Associated use**: e.g., OST (Original Soundtrack), Soundtrack (사운드트랙), Commercial ads (광고뮤직)
- **Region**: e.g., J-pop (Japanese pop), Chinese music (중국음악), World music (월드뮤직)
- **Purpose**: e.g., Prenatal education (태교), Ringtones (벨소리), Meditation (명상), MR (Music Recorded: used for "Instrumental")
- **Era**: e.g., 7080 (70s and 80s music), 00' Dance

This issue is not unique to Korean music. Some of these genres can be observed in Western music genre classifications as well (e.g., World music). However, a number of examples appear to be unique to Korean music (e.g., 7080, MR), which can pose challenges when mapping genres across cultural boundaries. For example, the genre "MR" could probably be mapped to "instrumental" but what about 7080? For someone who is not familiar with the historical context of Korean popular music, such label is essentially meaningless.

Categorization of music based on regions, although commonly used, seems especially problematic and can pose a problem for mapping Korean genres to Western genres. K-pop is most likely categorized under World music in North America (if it is categorized at all) as well as J-pop, etc., but clearly that will not be the case in Korea and Japan. Also many Korean artists now release albums in other countries making it difficult to categorize their music based on the region. For instance, when the Korean musician Boa releases albums in Japan sung in Japanese, should that be categorized as K-pop or J-pop?

4.3.2 Discrepancies in Genre Usage

As noted in previous research [5], in Korea, the genre label Pop can be used to denote any Western popular music from outside of Korea as opposed to popular music originated from Korea (*gayo* or K-pop). The fact that you can use the label Pop as a broader term that encompasses all Western music and at the same time use it as a sibling category of Rock, Electronic, etc. can lead to confusion as to what these labels actually refer to.

Another example of a Korean music genre with fuzzy boundaries is Ballad. The popularity of Ballad peaked in the 1980s, and it is still one of the most popular genres in Korea as demonstrated in Table 2. Bugs Music's genre/style dictionary defines Ballad as (translated):

> *Ballad is not a genre but a style. In other words, ballad is not a specific form of music that exists separately, but is used in conjunction with other genres to indicate a particular mood of the music. (e.g., Rock ballad, Pop ballad, Jazz ballad). In popular music, when the term "Ballad music" is used, it typically indicates a sentimental song with a slow tempo and sad content (usually about a breakup).*

Here, "Ballad" is defined as a style which is dependent on a genre, yet the term is used as a genre in a number of websites, including Bugs Music itself. Websites such as Bugs or Melon provide a list of styles to complement the genre classification, but some of these styles are essentially combinations of two genre terms (e.g., Dance Pop, Pop Rock, Rap-Metal, Club/Dance) which can also be confusing to users.

4.3.3 Unique Genres

Several genres we observed are unique to Korean music. Korean Traditional music such as *minyo* (민요) or *pansori* (판소리) were categorized under the term *gukak* (국악). Another example of a unique genre from popular music is Trot which refers to a distinctive style of popular music that does not exist in Western culture. The term Trot (트로트) is used to describe a "South Korean sentimental love song style performed with an abundance of vocal inflections [12]". This unique style of music has existed in Korea since the early 20th century and is typically enjoyed by older people. Mapping this music to a Western genre based on musical characteristics is simply not possible because there is not a good counterpart to this style of music. We conjecture that it is very likely that this type of music will end up being categorized as International or World music. However, in other non-Western cultures, we can in fact find similar styles of music to Trot. For instance, Trot and Japanese Enka music do share some musical similarities, but in the websites we analyzed, these genres were never grouped together under the same category. Although Trot is probably the most unfamiliar genre of the eight genres to Americans, there were much more disagreements among Ballad, R&B, and Rock than Trot. When the genre is unique and not similar to any of the genres listeners are accustomed to, it may be easier to identify than a genre that is adopted slightly differently in multiple cultures.

4.3.4 Composite and Hybrid Genres

Examining Table 1 and 2 shows that several genres are often used together; for example, Jazz and Blues, Ballad and R&B, Soul and Urban; Club and Dance, Hip-hop and Rap, etc. From an outsider perspective, some of these combinations of genres may appear baffling: what are the differences among R&B/Ballad, Hip-hop/R&B and R&B/Soul? Composite genres may reflect culturally significant distinctions. For instance, on Bugs music, Ballad/R&B is a sub-genre of *Gayo* (K-pop), but R&B/Soul

is a sub-genre of Pop songs (Western popular music). This is because in Korea, Ballad is generally used for any slow and sentimental songs, thus fitting well with R&B. This may be why there was such high disagreement between these genres (see Table 5).

Similarly, we observed many K-pop songs that are difficult to categorize in one genre; rather, they are better described as a hybrid, mixing components of several different styles of music (e.g., rap + dance + rock). This trend can be explained by two reasons: the influence of Seo Taiji and the popularity of idol groups in Korea.

The K-pop artist Seo Taiji had a major influence on Korean hybrid music styles. He is considered a central K-pop figure, and was the first to combine elements from multiple genres including Dance, Rock, Rap, Hip-hop, Ballad, and even Korean Traditional music, *gukak*. His influence is still found in many K-pop songs today. In fact, hybridity is now regarded as one of the most significant aspects of contemporary K-pop culture [11].

Another reason for the hybridity of K-pop music is the popularity of idol groups. The "idol" culture is quite prominent in South Korea. Major entertainment companies select young teenagers, have them go through years of training, and debut them in groups [3]. These groups are strategically created and often consist of members who have different strengths (e.g., singer, dancer, rapper). Therefore the music they present is also designed to integrate and accentuate the multiple roles and styles of the group's members. In this way, Idol as a genre, is a hybrid fusion of multiple music styles and influences.

Websites	Genre/Style Labels for Album *Hayeoga*
Melon	Genre: **Rock**; Style: **90's Ballad, Alternative Rock, Rap-Metal, 90's Dance**
Bugs	Genre: **Pop**; Style: **Dance Pop**
Naver	Genre: **Dance, Rock**
Olleh	Genre/Style: **Gayo (Kpop)/All**
Daum	Genre: **Gayo (Kpop) > Dance**
M.net	Gnere: **Gayo (Kpop) > Dance, Rock**

Table 6. Genre/Style Labels assigned for Seo Taiji's Album *Hayeoga* (하여가)

Although this creative approach of mixing different styles of music led to many popular K-pop songs, it also poses challenges for classifying these songs into a particular genre. For instance, Table 6 shows how one of Seo Taiji's albums was categorized in multiple Korean Websites. As you can see, it is categorized in multiple genres/styles including Rock, Ballad, Dance, Rap-Metal, and K-pop. Our 6 participants annotated the genre of the title track *Hayeoga* as: Rock (3), Ballad (2), and Dance (1).

5. CONCLUSION AND FUTURE WORK

As we investigated genres used in Korean music distribution websites and analyzed the annotation results, we discovered a number of issues, some that are unique to Korean culture and music, and some that are common issues of musical genres in any context. While the Korean websites we examined shared many genres with the ones used in Western culture, certain labels (i.e., Pop, Ballad) refer to different styles of music and there were also a few unique genres that reflect the context of K-pop culture. Annotation results show more disagreement for these genres as well. We also noted that genres are constructed based on multiple different criteria in addition to similarities in musical styles. In the Korean context, genres based on associated uses, purposes, and regions seem to be important. Mapping genre labels from multiple cultures will be challenging due to the different structure that values certain characteristic of division over the others in addition to genres that are unique to particular cultures.

In our future work, we plan to expand this study by collecting genres from other cultures for comparison. In addition, we want to conduct in-depth interviews of users from different cultures and ask how they understand, determine, and organize musical genres.

6. ACKNOWLEDGEMENTS

Funding support: Korean Ministry of Trade, Industry and Energy (Grant #100372) & the A.W. Mellon Foundation.

7. REFERENCES

[1] S. Doraisamy, H. Adnan, and N. M. Norowi: "Towards a MIR system for Malaysian music," *Proc. of the ISMIR*, pp. 342-343, 2006.

[2] K. L. Gwet: *Handbook of Inter-Rater Reliability*, Gaithersburg: Advanced Analytics, LLC, 2010.

[3] E-Y. Jung: "Articulating youth culture through global popular music styles: Seo Taiji's use of rap and metal," *Korean Pop Music: Riding the Wave*, Folkestone: Global Oriental, 2006.

[4] J. H. Lee and J. S. Downie: "Survey of music information needs, uses, and seeking behaviours: preliminary findings," *Proc. of the ISMIR*, pp. 441-446, 2004.

[5] J. H. Lee, J. S. Downie, and S. J. Cunningham: "Challenges in cross-cultural/multilingual music information seeking," *Proc. of the ISMIR*, pp.1-7, 2005.

[6] K. Markey: "Inter-indexer consistency tests," *Lib. and Info. Sci. Research*, Vol. 6, pp. 155–177, 1984.

[7] D. McEnnis and S. J. Cunningham: "Sociology and music recommendation systems," *Proc. of the ISMIR*, pp. 185-186, 2007.

[8] N. M. Norowi, S. Doraisamy, and R. Wirza: "Factors affecting automatic genre classification: An investigation incorporating non-Western Musical form," *Proc. of the ISMIR*, pp. 13-20, 2005.

[9] F. Pachet: Content management for electronic music distribution, *Communications of the ACM*, Vol. 46, No. 4, pp. 71-75, 2003.

[10] J-S. Park: "Korean American youth and trans-national flows of popular culture across the Pacific," *Amerasia Journal*, Vol.30, No.1, pp.147-169, 2004.

[11] D. Shim: "Hybridity and the rise of Korean popular culture in Asia," *Media Culture Society*, Vol.28, no.1, pp. 25-44, 2006.

[12] M-J. Son: "Regulating and negotiating in T'ûrot'û, a Korean popular song style," *Asian Music*, Vol.37, No.1, pp. 51-74, 2006.

[13] F. Vignoli: "Digital music interaction concepts: A user study," *Proc. of the ISMIR,* pp. 415-421, 2004.

COUPLING SOCIAL NETWORK SERVICES AND SUPPORT FOR ONLINE COMMUNITIES IN CODES ENVIRONMENT

Felipe M. Scheeren
Marcelo S. Pimenta
Institute of Informatics
UFRGS, Brazil
{felipems,mpimenta}
@inf.ufrgs.br

Damián Keller
Amazon Center for Music Research
UFAC, Brazil
dkeller@ccrma.stanford.edu

Victor Lazzarini
National Univ. of Ireland, Maynooth, Ireland
victor.lazzarini
@nuim.ie

ABSTRACT

In recent years, our research group has been investigating the use of computing technology to support novice-oriented computer-based musical activities. CODES (Cooperative Music Prototyping Design) is a Web-based environment designed to allow novice users to create musical prototypes through combining basic sound patterns.

This paper shows how CODES has been changed to provide support to some concepts originally from of Social Networks and also to Online Communities having Music Creation as intrinsic motivation.

1. INTRODUCTION

During the last few years, our research group has been investigating the use of computing technology to support novice-oriented computer-based musical activities. The development of this support has followed an interdisciplinary approach, and involves a multidisciplinary team of experts in Computer Music, Human-Computer Interaction (HCI) and Computer Supported Cooperative Work (CSCW). We are particularly interested in the convergence of technological support for cooperative creative activities, part of the field called "networked music"[20]. Network music allows people to explore the implications of interconnecting their computers for musical purposes. Because networked music works result from the convergence of social and technological aspects of Internet, this area has attracted the interest of the music technology community and the existing applications have evolved towards sophisticated projects and concepts including, for example, real-time distance performance systems, and various systems for multi-user interaction and collaboration. CODES is also a networked music system - CODES is a Web-based environment designed to support music creation by means a process called Cooperative Music Prototyping (CMP) - but with special focus on music novices.

Similarly to other Rich Internet Applications such as YouTube, MySpace, and Flickr – that have turned the passive user into an active producer of content, bringing into the picture new purposes, like engagement, entertainment and self-expression – CODES makes possible a novice be actor if their own musical experiences in music The main motivation of our work is the belief that no previous musical knowledge should be required for participating in creative musical activities.

The goal of this paper is to present and discuss several concepts developed by our research group concerning how CODES has been changed for coupling social networks services and for providing support to an online community for cooperative music making. Our environment started out as a website that people could use to create their music interactively and cooperatively, but it has grown into a more general online community of people allowing to build an audience around music experimentation, music creation, (music) knowledge sharing, and entertainment.

2. CODES ENVIRONMENT

Differently from YouTube, Flickr, and even MySpace, where people only publish their "already ready" content, CODES is a system for music creation, instead of a system just for publishing music. CODES offers a high level music representation and user interface features to foster easy direct manipulation (drag-and-drop) of icons representing sound patterns (predefined MP3 samples with 4 seconds of duration), combining them to create (new) simple musical pieces – called Music Prototypes (MPs.)

Using adequate support features, CODES users can create, edit, share and publish MPs in their group or on the Web. These shared MPs can be repeatedly tested, listened to, and modified by the partners, who cooperate on MP refinement. Users can start a new MP by choosing the name and the musical style they want. The selection of a musical style allows CODES to filter the sound patterns offered to the user. However, since all styles are available from the sound library, mixing sound patterns from different styles within the same musical prototype is still possible.

Edition in CODES includes actions like "drag-and-drop" sound patterns from the sound library to the editing area, "move," "organize," "delete," "expand" the duration,

and "collapse" sound patterns to listen to the final result – a MP. When sharing a MP, the "owner" user can invite CODES users to use a search engine or may send explicit invitations via e-mail to non-members asking them for cooperation. When someone accepts such an invitation, the user becomes a prototype partner and can edit the MP like the owner does.

The prototypical nature of CODES is designed and built to provide a novice-oriented perspective. At any time users can listen to the musical prototype and link arguments to their decisions. Thus, all prototype partners can discuss and change ideas about each step of the prototype refinement, in order to understand each other's decisions. When someone considers that the resulting sounds are good, a "publication request" can be triggered and the group may discuss and deliberate about the publication of this musical prototype in the CODES home page. This activity is called musical prototype publishing. As an alternative to publishing their music, users may export their musical prototype, and share it at will. Thus, a novice may experiment with music by combining, listening and rearranging pre-defined sound patterns to create the MPs. Furthermore, CODES users may cooperate with partners in a cyclical and collaborative process of prototype creation - the CMP - through customized awareness, argumentation, and negotiation mechanisms until a final consensual prototype stage is reached.

2.1 Awareness Mechanisms in CODES

Through CODES, anyone can draft, test, modify, and listen to MPs. These actions can be done by the first author and their partners that cooperate in the refinement of the MP. This implies a focus not only on community management (i.e., discovering, building, or maintaining virtual communities) but also on experimenting and participating in specific design practices using a suitable interaction vocabulary. This process suggests the existence of noteworthy distinct kinds of cooperation activities. Systems aiming to provide effective support for these different activities have to meet specific requirements. Awareness and conflict resolution are already considered critical issues in general CSCW systems. However, mechanisms existing in other systems need some adaptation to take into account the idiosyncrasies of the CMP context. The ultimate goal is to provide actual cooperation, social knowledge construction, argumentation, and negotiation among the actors of the MP design activities. This type of cooperation is supported by a set of mechanisms borrowed from the Software Engineering and HCI areas and specially adapted for CODES, namely awareness, music prototyping rationale, authorship, version control, and conflict resolution [19].

2.2 Related Work

The most representative systems found in the literature of collective musical creation or music experimentation on the Web concerned with musical experimentation by novices are: Daisyphone [4], PitchWeb [9], Web- Drum [5], Public Sound Objects – PSOs [2], and JamSpace [11]. A brief description of each system and a comparative analysis according to various criteria (categorized as technological and architectural, computer music related, HCI related, CSCW related) may be found at [15]. In summary, the main drawback of existing networked music systems comes from the lack of focus on effective cooperation: in our viewpoint, effective collaboration is directly obtained through adequate adoption of techniques such as argumentation, authorship (allowing users to know their original contributions at anytime), interaction trace, awareness, and group memory. When these techniques are not adequately considered or explored within interaction design (taking into account real needs and tasks of the novice users for cooperative musical interactions), networked music systems do not provide support for effective novice usage [16].

2.3 CODES Principles

CSCW concepts and tools have made more obvious the existence of a more connective, cooperative, and collective nature of creativity rather than the prevailing focus on the individual. The creative processes being highly fuzzy, the programming of cooperative tools for creativity and innovation should be adaptive and flexible.

Creativity models in music have been heavily influenced by general models of creativity such as [10-23]. While some creativity models for music emphasize the cognitive dimensions of music creation [6-7], others consider its material dimension as an integral part of music creation [3-8]. In fact, to assess musical creativity is a very difficult task because social context, physical context, and personal factors shape the creative act and may determine the function and the dynamic of the creative processes involved [14]. Given this complex scenario, to put into practice collaboration, flexibility and multiculturality [18], we have adopted two principles which have been confirmed by findings obtained during CODES development and usage: (a) Music creation by novices should be prototypical; and (b) Music creation by novices should be cooperative [16].

Within prototypical music creation, novices can draft an initial musical sketch (a simple MP) which can be tested, modified and listened to, applying a cyclical refinement process until a final stage is reached. In the music literature, "draft" is the term commonly applied to initial creative products. However, here the emphasis is on the cyclical prototyping process and not on the product itself. Consequently, in this chapter "prototype" and "draft" are equivalent. The prototyping process clearly resembles the incremental software development cycles adopted in the industry. Since music creation can be considered a design activity, it seems natural and straight-

forward to adopt a prototypical process to model this activity.

In cooperative music creation, the refinement of an initial musical idea is a consequence of the collaboration of the author with her partners. In this context, what matters is not necessarily the musical quality of the finished work, but giving the possibility of a creative experience to a larger community of participants. The members build a social network by explicit invitation to cooperate until a final consensual MP stage is reached.

3. FROM CODES TO SOCIAL CODES

CODES was conceived originally to be a Computer Supported Cooperative Work (CSCW) system with a design based on cooperation and interaction concepts, so the evolution towards SNS and communityware is straightforward. Even though support systems for both group types — teams and communities — have developed independently, both areas have something in common: the contact facilitation with unknown and known collaborators [13]. While community support systems concentrated mostly on the building process, i.e. finding people with similar interests, CSCW focused on the collaboration process, i.e. the synchronization and exchange of information in the context of a specific team task. Like [21], we are convinced that awareness can be a common base for community support systems to improve contact building as well as for CSCW to maintain group work at a high performance level. Therefore, we need to address how we extend CODES from traditional CMP support and CSCW perspective towards online human community support.

Successful online communities motivate online participation. An online community provides people a place to come together using the Internet: it is always on and is a more accessible way to keep in touch with people who are geographically far or with those who have conflicting schedules.

Every online community has (sometimes implicit) rules that may specify how to participate and to engage with the community — from peripheral participation ("lurker") to explicitly recognized participation ("leader"). CSCW technologies can provide tools for supporting these roles. In the case of communities, these tools are used in combination, including text-based posts and chat rooms and forums that use voice, video, or avatars.

The community metaphor can create several different functions for encouraging social interaction in communities [12]: 1. Knowing each other; 2. Sharing preference and knowledge; 3. Generating consensus; 4. Supporting everyday life; 5. Assisting social events.

A virtual community is a social network of individuals who interact through specific media, potentially crossing geographical and political boundaries in order to pursue mutual interests or goals. One of the most pervasive types of virtual community includes Social Networking Services (SNS), which consist of various online communities. Nowadays, there are lots of SNS with focus on music listening and sharing, rather than music creation. YouTube and Vimeo are video sites that feature musical contents. Some SNS, as Myspace, have a social character, merging social networks aspects with content distribution. Facebook includes several musical pages and bands profiles. Even Apple tried to foster a musical social network, known as Ping.

Traditionally, a SNS essentially consists of a representation of each user (often a profile), his/her social links, and a variety of additional services. CODES provides a distinct vision of a SNS having music as its intrinsic motivation, combining the traditional features (profiling, social links) to different and specific features related to CMP. Thus, CODES has three different levels of viewing and interaction:

a. *Level 1*: Public level, like an broadcasting channel, as Myspace, where posts are posted on to a "bulletin board" for everyone, without resorting to messaging users individually. Thus, people who do not know each other can check some data (personal information and prototypes) that the users select to publish, making this information available to any visitor;

b. *Level 2*: Restricted one-way level, as on Twitter or Google+, when users may subscribe to other users' posts — this is known as following and subscribers are known as followers. If a user follows someone, this is a non-mutual relationship, where who is being followed does not need to follow back. The follower can see someone else's timeline as an RSS feed and also access MPs and even modify it (if this option is allowed in the MP), but these modifications are just local and temporary. It is also possible to repost a post from another user, and share it with one's own followers;

c. *Level 3*: Partnership level, when two people follow each other, it establishes a collaborative relation, where they can, not only suggest, but also edit the prototypes together. In this case, they are named 'partners.'

CMP is an activity that involves people creating groups and working together on an MP as a shared workspace. In CODES, a cooperative musical prototype is initiated by someone that creates a new prototype, elaborates an initial contribution, and asks for the collaboration of other "partners" by sending explicit invitations.

Partners who accept the invitation can participate in the collaborative musical manipulation and refinement of the prototype. The group can publish the final or partial results of their CMP in the public spaces (level 1 above), in which interested users could discover it and follow

(level 2 above) or join the collaboration as new partners (level 3 above).

In order to avoid undesired dependencies, inconsistencies and conflicts between contributions, preserving authorship among the several contributions of a community, CODES implements a particular layer-oriented version management mechanism. In this approach, each layer represents one partner's view, and the union of partners' contributions (a combination of layers) results in a cooperative MP version. Any partner can browse between the contributions, independently of the creator, keeping the creator's original ideas and authorship. It is also possible to edit another user's contribution, by issuing an explicit "modification request" to a partner. The interested reader can find more details in [19].

The basic idea of our CMP process is that members cooperate not only by means of explicit conversation and explicit actions on a shared object space, but also by interpreting the messages and actions of other actors in accordance with the model of their thinking and acting, which has been built up in the course of their interaction. A shared objects space involves prototype-oriented information, which comprises all information about musical prototypes, including their composition (combination of sound patterns, versions formed by layers) and social-oriented information (including interactions between actors during the process).

3.1 CODES Social: Features and UI Description

In our social networking services version of CODES, we choose to apply several well known HCI interface guidelines to create a dynamic and creative environment, providing SNS for the emergence of communities and enabling knowledge sharing by means of rich interaction and argumentation mechanisms associated with the MPs evolution. Several basic CODES features are similar to conventional features of Social Networking sites. Most often, individual users are encouraged to create profiles containing various information about themselves. To protect user privacy, social networks usually have controls that allow users to choose who can view their profile, contact them, add them to their list of contacts, and so on. Users can upload pictures of themselves to their profiles, post blog entries for others to read, search for users with similar interests, and compile and share lists of contacts. In addition, user profiles have a section dedicated to comments from friends and other members.

CMP is a simple cyclical process including the following activities: (a) MP creation, (b) MP edition, (c) MP sharing, and (d) MP publishing. Through a Music Prototyping Rationale (MPR) mechanism — based on the Design Rationale concept from HCI — each user may associate comments (i.e. an idea or an observation) and arguments (pro or cons) to any action on any MP. The arguments can be addressed to a specific partner or to the whole group. Due to the exploratory nature of CODES usage, MPR is one of its most important characteristics, allowing users to perceive and analyze group members' actions on music prototypes ("to understand WHAT my partners are doing") and the reasoning behind these actions ("to understand WHY my partners are doing it").

The CODES interface was designed to strike a balance between user interfaces that are so easy-to-use that they end up depleting their expressiveness, and others that are so complicated that they discourage beginners. The CODES user interface has three levels of interaction for different user profiles: (a) Public Level, (b) MP Editing Level, and (c) Sound Pattern Editing Level. The lowest level of CODES — sound pattern editing — is a kind of "piano roll" editor, having no social-oriented features. Therefore, we will only discuss the other two levels.

At the public level, anyone (including non-members) can access and explore musical prototypes, by searching and listening. One of the goals at this level is to encourage the potential audience to become CODES members, and encourage members to publish their musical prototypes to foster the formation of a virtual community focusing on music.

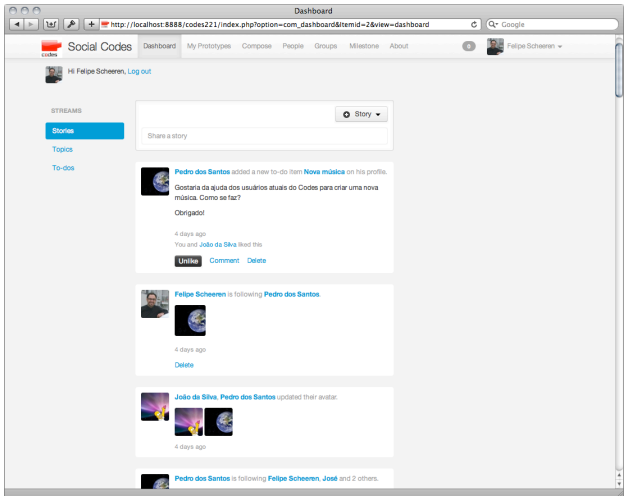


Figure 1. CODES main page

The main page at the public level is divided into four areas (see Figure 1):

• (A) Top Pane: System logo, Dashboard (the main page, with user's timeline, which shows followings updates), My Prototypes (user's MP and the ones user is collaborating with), Compose (opens the CODES MP Editing Level interface), People (shows a list of CODES users and a search box), Groups (where users can choose to follow discussion groups or create new ones), Milestone (where users can create a set of to-dos to each MP), About page, Notification area and a dropdown menu with user settings.

• (B) Left Pane: User's name and picture with logout and three kinds of streams: Stories (the main timeline

view), Topics (users and followings discussions about it's prototypes) and users MP's To-dos.

• (C) Main Pane: where the actions happen. All the things that you can choose in the other panes will appear on this one. Here, users will see and edit their prototypes, following or follower lists, the search results, etc.

• (D) Bottom Pane: Systems' information – not shown.

The user's main view is different of the user's profile view. The left pane has a bigger image of the person, with links for his music gallery and friends list. And the content pane shows new updates, such as new music, collaborations, followings, or followers.

The MP Editing Level is the most important level of the system. At this level, users can create and edit their MPs cooperatively (see Figure 2). The edition of a MP in fact is a simple task. The sound patterns are dragged from the sound library — a region having icons representing music instruments organized in folders named 'rock,' 'funk,' 'jazz,' etc. — and dropped into the MP editing area — the biggest region above. The sound patterns displayed in the editing area are played from left to right. At any time, the user can play the MP existing in the MP editing area using the execution control buttons: Play, Rewind. The basic action at this level is to add or remove sound patterns within the editing area, as well as to change their sequence, size, combination, and position. Each author's contribution in the shared workspace is identified by color: the edges of icons of sound patterns are colorful, with the same color chosen by the user at the registration. A detailed description of all features related to MP edition can be found in [15].

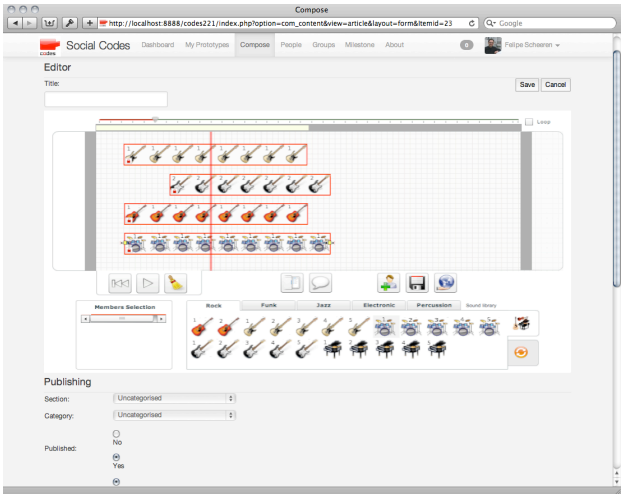


Figure 2. Musical Prototyping editing page

Every new MP is *a priori* private, so without authorization no one is able to see/listen it. However, in "My Prototypes", the user can see all MPs it has created and all the MPs it is collaborating with (as a partner). At the preferences panel for each MP, the user can select the musical style of each composition (multiple tags are allowed), as well as one of the following distribution options: *Private* (No one but the owner will be able to edit/listen this MP), *Collaborative* (People will be able to listen and edit the parts of the MP that the owner sets as allowed to edit), *Public* (Everybody can listen, Collaborators can edit it and followers too), *Closed* (Everybody can listen, but only the author can edit).

Several difficulties were addressed in this work for bridging the gap between groups of novice users and the possibility of creating online communities for music making over the Web. Indeed, one of the main challenges was the combination of cooperation-oriented mechanisms (for effective collaboration in a CMP) with community-oriented mechanisms (for effective community management).

4. CONCLUSION

Starting from an environment for Cooperative Activities for musical creation, this work has discussed two main challenges of networked music systems for novices:

a) how to provide support for social network services in CODES environment and;

b) how to create Online Communities having music as intrinsic motivation.

The first challenge is specially relevant with respect to provide different mechanisms — well-known in social networks sites but not known by most of users of networked music systems at all — in CODES environment. The second challenge is important for providing new features and motivations for social relationships. Since people have always found listening, performing, or creating music significant in their lives, whether for enjoyment or for social cohesion, we think such online community related to Music creation by novices is an interesting endeavour because music making in modern life tends to be left in the hands of the professional artists, musicians, and singers.

CODES users does not need to have previous musical knowledge for participating in musically creative activities. Obviously, providing support for non-musicians or for musicians are not the same thing [17]. Musician-oriented systems usually include full and complex information, concepts, and interface functionalities that are part of the "musician's world" and usually not understood by ordinary users. In our work such knowledge-intensive skills are replaced by strong commitment with communication, cooperation and sharing by members of a community.

Through CODES, novices may have the opportunity to be – like experienced musicians are – the actors of their own musical experiences. This means they can draft simple musical pieces that can be modified, and repeatedly listened to, both by the first authors and by their partners that will be cooperating in the refinement of the MP. This implies a focus not only on community management (i.e.,

discovering, building or maintaining virtual communities) but also on experimenting and participating in specific design practices using a suitable interaction vocabulary.

Moreover, this suggests the existence of noteworthy distinct kinds of cooperation activities, and that systems aiming to provide effective support for these different activities should also meet different requirements.

5. REFERENCES

[1] Adobe: "Flex," Retrivied May 10, 2013 from http://www.adobe.com/products/flex.html

[2] A. Barbosa: "Public sound objects: A shared environment for networked music practice on the web," Organised Sound, Vol. 10, No. 3, pp. 233–242, 2005

[3] S. Bennett: "The process of musical creation: Interview with eight composers," Journal of Research in Music Education, No. 24, pp. 3–13, 1976.

[4] N. Bryan-Kinns and P. G. T. Healey: "Daisyphone: Support for remote music collaboration," Proceedings of the 2004 Conference on New Interfaces for Musical Expression, pp. 27-30, 2004.

[5] P. Burk: "WebDrum," Retrieved May 02, 2013, from http://www.transjam.com/webdrum

[6] C. W. Chen: "The creative process of computer-assisted composition and multimedia composition: Visual images and music," (PhD Thesis). Royal Melbourne Institute of Technology, Australia, 2006

[7] D. Collins: "A synthesis process model of creative thinking in music composition" Psychology of Music, Vol. 33, No. 2, pp. 193–216, 2005.

[8] C. Dingwall: "Rational and intuitive approaches to music composition: The impact of individual differences in thinking/learning styles on compositional processes," (Bachelor Dissertation Thesis). University of Sydney, Australia, 2008.

[9] W. Duckworth: "Making music on the web," Leonardo Music Journal, Vol. 1, No. 9, pp. 13–17, 1999.

[10] H. Gardner: "Creating minds," BasicBooks, New York, NY, 1993.

[11] M. Gurevich: "Jamspace: Designing a collaborative networked music space for novices," Proceedings of NIME, pp. 118–123, 2006.

[12] T. Ishida: "Community computing: Collaboration over global information networks," John Wiley and Sons, New York, NY, 1998.

[13] T. Ishida (ed): "Community computing and support systems: social interaction in network communities," Lecture Notes in Computer Science, Vol. 1519, 1998

[14] D. Keller, M. H. Lima, M. S. Pimenta and M. Queiroz: "Assessing musical creativity: Material, procedural, and contextual dimensions," Proceedings of the 21st Conference of the Brazilian National Association of Research and Post-Graduation in Music (ANPPOM), 2011

[15] E. M. Miletto: "CODES: An interactive novice-oriented web-based environment for cooperative musical prototyping," (PhD Thesis), Federal University of Rio Grande do Sul, Brazil. 2009.(available at http://hdl.handle.net/10183/2281)

[16] E. M. Miletto, M. S. Pimenta, F. Bouchet, J.-P. Sansonnet and D. Keller: "Principles for music creation by novices in networked music environments," Journal of New Music Research, Vol. 40, No.3, 2011

[17] E. M. Miletto, L. V. Flores, M. S. Pimenta, J. Rutily and L. Santagada: "Interfaces for musical activities and interfaces for musicians are not the same: The case for codes, a web-based environment for cooperative music prototyping," Proceedings of the 9th International Conference on Multimodal Interfaces, pp. 201-207, 2007.

[18] M. Pimenta, D. Keller, E. Miletto, L. V. Flores and G. G. Testa: "Technological Support for Online Communities Focusing on Music Creation: Adopting Collaboration, Flexibility and Multiculturality from Brazilian Creativity Styles," pp. 283-312, In: N. A. Azab (Org.): Cases on Web 2.0 in Developing Countries, IGI Global, Hershey, 2013.

[19] M. S. Pimenta, E. M. Miletto and L. V. Flores: "Cooperative mechanisms for networked music," Future Generation Computer Systems, Vol. 27, No. 1, pp. 100–108, 2011

[20] M. Schedel and J. P. Young: "Editorial," Organised Sound, Vol. 10, No. 3, pp. 181–183, 2005.

[21] J. Schlichter, M. Koch and C. Xu: "Awareness - The common link between groupware and community support systems." Proceedings of CCSS, pp. 78-94, 1998.

[22] W3C: "Web standards," Retrieved May 10, 2013 from http://www.w3.org/standards/

[23] G. Wallas: "The art of thought," Harcourt Brace and World, New York, NY, 1926.

BASIC EVALUATION OF AUDITORY TEMPORAL STABILITY (BEATS): A NOVEL RATIONALE AND IMPLEMENTATION

Zhuohong Cai[1], Robert J. Ellis[1], Zhiyan Duan[1], Hong Lu[2], and Ye Wang[1]

[1] School of Computing
National University of Singapore
{a0109706,ellis,zhiyan,wangye}@comp.nus.edu

[2] School of Computer Science
Fudan University
honglu@fudan.edu.cn

ABSTRACT

The accurate detection of pulse-level temporal stability has important practical applications; for example, the creation of fixed-tempo playlists for recreational exercise (e.g., jogging), rehabilitation therapy (e.g., rhythmic gait training), or disc jockeying (e.g., dance mixes). Although there are numerous software algorithms which return simple point estimate statistics of "overall" tempo, none has operationalized the beat-to-beat *stability* of an inter-beat interval series. We propose such a method here, along with several novel summary statistics. We illustrate this approach using a public data set (the 10,000-item subset of the Million Song Dataset) and outline a series of future steps for this project.

1. INTRODUCTION

Motor synchronization with an auditory beat has been deemed a human cultural universal [20] and a "diagnostic trait of our species" [16]. Even infants show perceptual sensitivity to and motor coordination with musical rhythms [26,28]. A temporally stable beat facilitates rhythmic human movement during leisure activities such as exercise (for recent reviews, see [10,11]). It also serves as the basis for a class of gait rehabilitation therapies known as "Rhythmic Auditory Stimulation" or "Rhythmic Auditory Cueing" for Parkinson's disease (for reviews, see [12,22]), stroke [25], and others [27].

Numerous beat tracking algorithms have been developed which return a time series of detected beats for a given audio input (for reviews, see [5,18]), returning a simple "beats per minute" point estimate of tempo. None of these algorithms, however, has attempted to operationalize the beat-to-beat *stability* of that tempo over time, other than occasional efforts to note whether multiple excerpts taken from the same audio file have the same approximate tempo. Such a coarse estimate of tempo stability does not have the necessary precision for the type of clinical applications cited above applications, which not only need to know if a given audio file is stable, but the precise time indices at which it is stable (so as to preserve that information in the playlist).

To address these issues, we present a novel analysis tool: "Basic Evaluation of Auditory Temporal Stability" (BEATS) for Matlab (version ≥ 7.8). BEATS is *not* a beat tracking algorithm; instead, it uses the output of an existing beat tracking algorithm (i.e., beat and barline onset timestamps) to provide a full set of outcome statistics. Here, we focus on the "Million Song Subset" of 10,000 metadata files selected from the Million Song Dataset [1] (http://labrosa.ee.columbia.edu/millionsong/), with all audio files processed using the proprietary "Analyze" algorithm [9] developed by The Echo Nest (www.echonest.com). Compatibility with this data source has long-term advantages, as the full Echo Nest library contains over 34 million analyzed audio files.

2. METHODS

2.1 Data inputs

BEATS pulls four Echo Nest fields from each metadata file: `beats_start` and `bars_start` (the estimated onsets of successive beats and barlines, respectively); and `tempo` and `time_signature`. Next, the `beats_start` and `bars_start` vectors are transformed into an inter-beat interval (IBeI) series and an inter-bar interval (IBaI) series, respectively, by taking the first-order difference of each vector.

2.2 Initialization Thresholds

BEATS requires the user to specify three Initialization Thresholds:

(1) "Local Stability Threshold", θ_{Local}: a percentage value (default = 5.0%) used to define temporal stability at the level of individual and successive IBeIs (detailed below).

(2) "Run Duration Threshold", θ_{Run}: the minimum duration (default = 10 s) of a set of consecutive IBeIs (i.e., a "Run") that fall below θ_{Local}.

(3) "Gap Duration Threshold", θ_{Gap}: the maximum duration (default = 2 s) between the last timestamp of Run_j and the first timestamp of Run_{j+1}.

2.3 Internal Calculations

The first statistic calculated by BEATS is the central tendency, λ (for "location") of the IBeI series: the single value which best characterizes the predominant IBeI. Obtaining an "optimal" value for λ can more challenging than simply taking the median or mode of a series. Consider the hypothetical IBeI series **S** shown in Figure 1, which exhibits two tempo changes (at the 21st and 41st IBeIs). In Matlab, `median`(**S**) = 0.869 and `mode`(**S**) = 0.477. (`mode` is known to be problematic for both non-discrete and non-quantized data.) Neither statistic effectively captures the central tendency of **S**.

To address this, we define an iterative loop in which the range of **S** is divided into an increasing odd number of bins *k* from 1 to 15. The loop stops at the largest value of *k* in which the most-frequent bin contains just over one-third of the data (or quits when *k* = 15). λ is then defined as the median value within the most-frequent bin. Under this definition, **S** has a λ = 0.993, which better captures its central tendency.

Having derived λ, the longest "Stable Segment" within an IBeI series can be identified. The first step in this process is to quantify local temporal stability in two ways: local *deviations* (from λ) and local *differences* (adjacent IBeIs). Local deviations are quantified by an absolute deviation from λ (ADL), calculated for each element *i* of IBeI series **S**:

$$S_{\mathrm{ADL},\boldsymbol{i}} = 100 \times \frac{|S_i - \lambda|}{\lambda}. \qquad (1)$$

Local differences are quantified by a first-order absolute successive difference (ASD), calculated for each element *i* of **S**:

$$S_{\mathrm{ASD},\boldsymbol{i}} = 100 \times \frac{|S_i - S_{i-1}|}{0.5 \times (S_i + S_{i-1})}, \qquad (2)$$

where $S_{\mathrm{ASD},1} = 0$ to preserve the series indexing. Both $\mathbf{S}_{\mathrm{ADL}}$ and $\mathbf{S}_{\mathrm{ASD}}$ are expressed relatively (i.e., as percentages) to facilitate comparisons across IBeI sequences in different tempo ranges.

Next, a binarized version of **S** ($\mathbf{S}_{\mathrm{Bin}}$) is created:

$$S_{\mathrm{Bin},i} = \begin{cases} 1, & \begin{cases} S_{\mathrm{ADL},i} \le \theta_{\mathrm{Local}} \\ S_{\mathrm{ASD},i} \le \theta_{\mathrm{Local}} \end{cases} \\ 0, & \text{otherwise} \end{cases}. \qquad (3)$$

$\mathbf{S}_{\mathrm{Bin}}$ identifies the locations of Runs (i.e., strings of 1s) and Gaps (strings of 0s) within the IBeI series itself. Finally, the Stable Segment is defined as the longest sequence of {Run_j, Gap_j, Run_{j+1}, … Gap_{n-1}, Run_n}, where each Run has a duration ≥ θ_{Run}, each intervening Gap has a duration ≤ θ_{Gap}, and the median IBeI value of each pair of neighboring Runs has a percent difference of ≤ θ_{Local}.

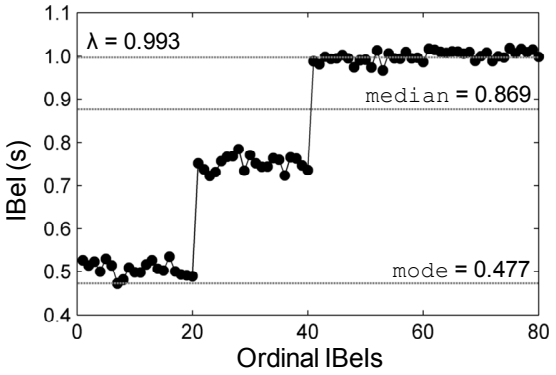


Figure 1. A hypothetical IBeI series, with IBeI values (*y*-axis) plotted ordinally (*x*-axis). The usual estimators of median and mode are both non-optimal. The newly-proposed λ statistic provides a better match.

2.4 Outcome Statistics

BEATS calculates six statistics from each Stable Segment:

(1) "Stable Duration" (in seconds): the time between the first and last time stamps of the longest run.

(2) "Stable Percentage": Stable Duration relative to the duration of the entire IBeI series.

(3) "Estimated Tempo" (in beats per minute, BPM): the median IBeI value within the Stable Segment, multiplied by 60.

(4) "Estimated Meter": a more precise definition than the typical beats-per-bar value. Specifically, for a Stable Segment with a bar timestamp series {r_i, r_{i+1}, ...} and beat timestamp series {b_j, b_{j+1}, ...}, let B_i be the number of beat timestamps for which $r_i \le b_j < r_{i+1}$. Estimated Meter is then taken as the mean of all B_i. Only in the case when all B_i have the same value will an integer value result (e.g., 4.00), providing a simple way to identify the presence of a changing meter within the Stable Segment.

(5) "Percentile of Absolute Deviations from λ" ($PADL_P$): a statistically robust alternative to a percentage-based coefficient of variation (CV), used extensively in the gait literature (for a review, see [6]). For an IBeI series **S**, CV is defined as the standard deviation of *S* divided by the mean of *S*, and multiplied by 100. Because it makes use of the standard deviation, CV is susceptible to inflation by high-value outliers (e.g., a beat that is "dropped" by the beat tracking algorithm). By contrast, $PADL_P$ offers a more robust formulation:

$$\mathrm{PADM}_P = \mathtt{prc}(\mathbf{S}_{\mathrm{ADL}}, P), \qquad (4)$$

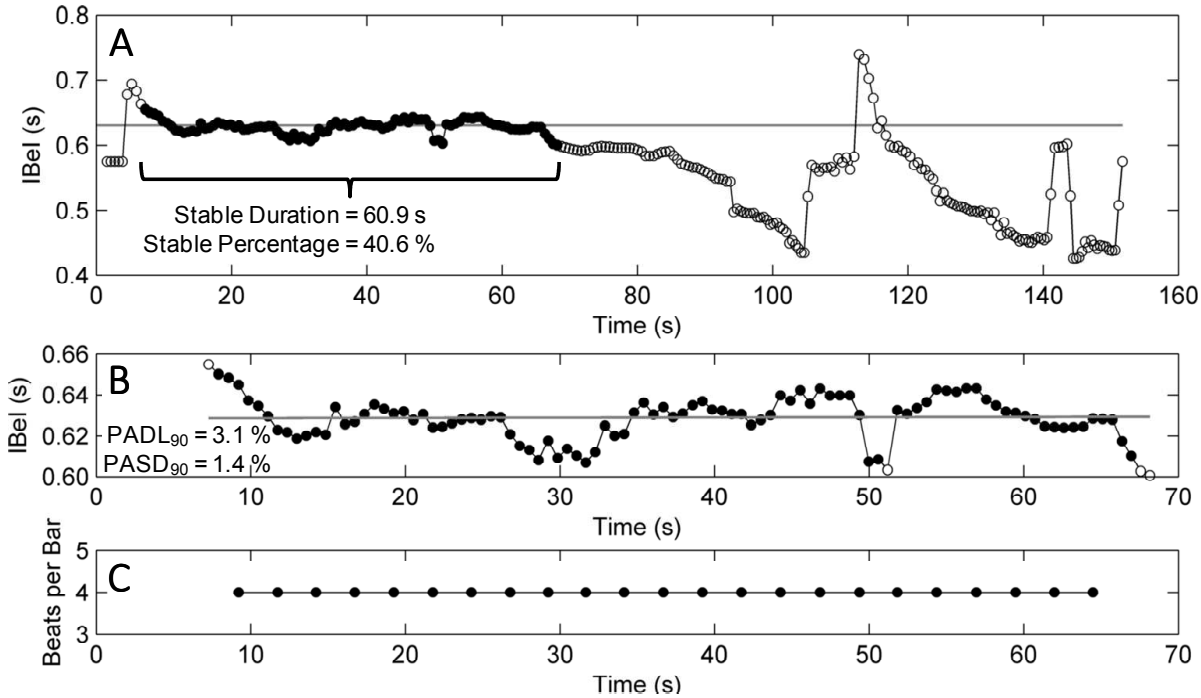


Figure 2. Visual illustration of the Outcome Statistics calculated by BEATS. Panel A shows the IBeI series (*y*-axis) as a function of real time (*x*-axis), with λ shown as a horizontal gray line (at $y = .631$) and the Stable Segment highlighted in filled circles. Panel B shows the Stable Segment in isolation; the best-fitting line (gray line) reveals a lack of temporal drift. Panel C shows all the number of beats per bar for all detected downbeats; Estimated Meter is consistent at four beats per bar throughout the Stable Segment.

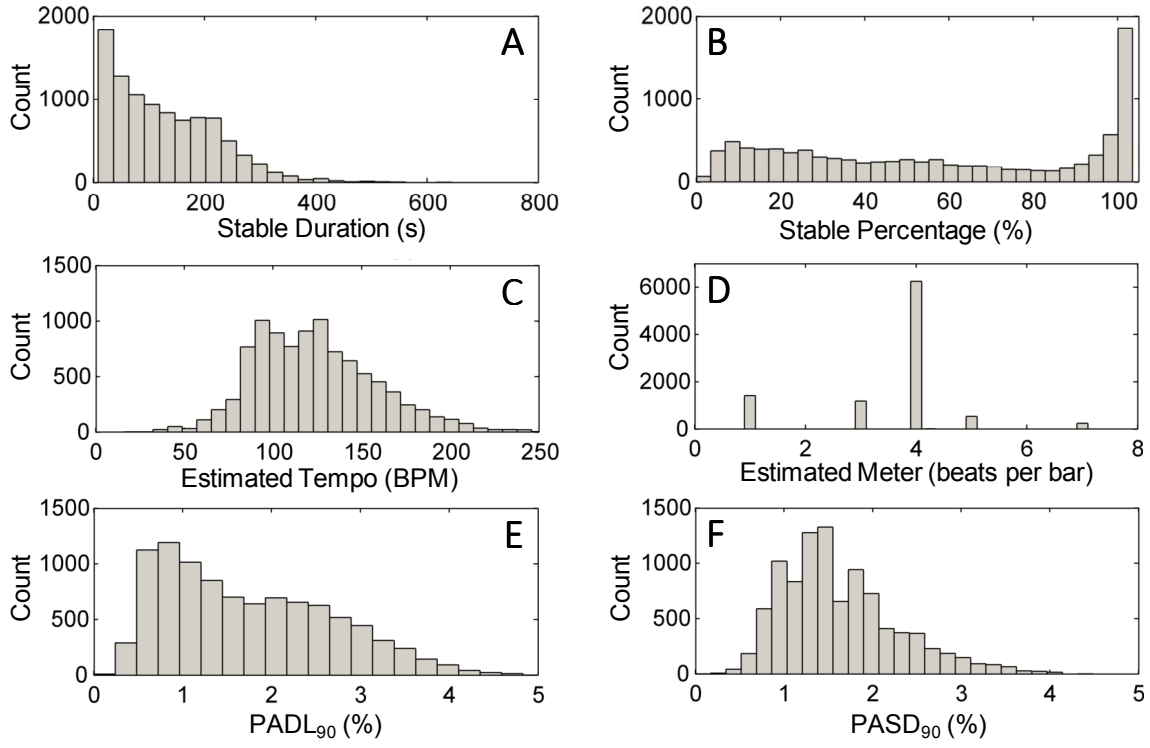


Figure 3. Histogram summaries of the six Outcome Statistics across the full 10,000-item dataset.

where $\texttt{prc}(\mathbf{S}_{ADL},P)$ is the Pth percentile of the $\mathbf{S}_{ADL}$ vector (Eq. 1).

(6) "Percentile of Absolute Successive Differences" ($PASD_P$): a robust alternative to the root-mean-square of successive differences (rMSSD), widely used in studies of heart rate variability to quantify beat-to-beat fluctuations (for a review, see [24]. However, just as squaring deviations from the mean make the standard deviation susceptible to outliers, squaring successive differences yields a potentially inflated RMSSD. By contrast, $PASD_P$ is defined as:

$$PASD_P = \texttt{prc}(|\mathbf{S}_{ASD}|, P) . \quad (5)$$

For both $PADL_P$ and $PASD_P$, BEATS uses $P = 90$ as its default. In practice, however, any value of P between 0 and 100 may be used.

2.5 Implementation

BEATS was run on the 10,000-item dataset using its default Initialization Thresholds (Section 2.2). (The rationale behind $\theta_{Local} = 5.0\%$ is explained in Section 4.1.)

3. RESULTS AND DISCUSSION

Figure 2 presents a visual illustration of the six Outcome Statistics calculated by BEATS, in a single file from the Million Song Subset (Track TRAHNHL128F14A4DDD: "In the Hall of the Mountain King" by Edvard Grieg, performed by the Staatskapelle Dresden; available at http://open.spotify.com/track/2cTXwtIFEeCNa0ZtbI97zh). This work is famous for its *accelerando*, which can be seen in the IBeI plot of Figure 2A (albeit with some confusion on the part of the Echo Nest "Analyze" algorithm [9], a point discussed further in Section 4.1). Such a recording would be of limited use for a constant-tempo exercise paradigm. A temporally stable segment, however (using $\theta_{Local} = 5.0\%$), can in fact be found between the 0′08″ and 1′09″, which can be more clearly appreciated in Figure 2B. The identified Stable Segment has a $PADL_{90} = 3.1\%$ and a $PASD_{90} = 1.4\%$, markedly different than if those statistics are calculated from the *entire* IBeI series ($PADL_{90} = 23.3\%$ and a $PASD_{90} = 3.1\%$). Finally, Figure 2C shows the number of beats per bar within the Stable Segment; this yields an Estimated Meter = 4.

Figure 3 presents a histogram for each of the six BEATS Output Statistics across the full 10,000-file data set. Of particular note is Figure 2B, which indicates that Stable Percentage varied widely across the data set. Indeed, only 18.6% of files were deemed temporally stable (i.e., as defined by $\theta_{Local} = 5.0\%$) over their entire duration (i.e., Stable Percent = 100). In other words, the probability that a song randomly selected from the MSD can be played it in its entirety as part of a rhythmic movement paradigm (i.e., has a moderately stable perceptual tempo with less than 5.0% local tempo variability) is < 20%.

Figure 4 presents a slightly different picture, plotting the percentage of files (y-axis) with a Stable Duration ≥ the x-axis value. Allowing BEATS to identify the Stable Segment within each audio file (if present) yields a higher percentage of files available for exercise playlists; for example, 55.7% of files are temporally stable over a duration of ≥ 90 s within the file—three times the number of files that are stable over their entire duration.

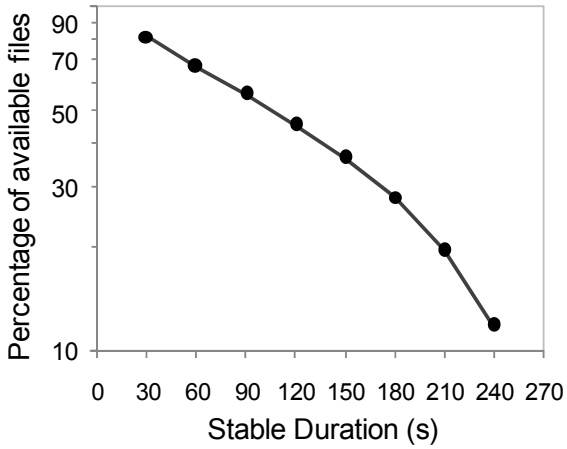


Figure 4. The percentage of files in the 10,000-item dataset which have a Stable Duration ≥ the corresponding x-axis value.

The power of BEATS lies in the flexible way its Outcome Statistics may be combined to deliver a stimulus set optimized to a user's specific needs. For example, a gait training paradigm requiring highly stable music might use the following set of inclusion criteria: Stable Duration ≥ 120 seconds, Tempo between 50 and 150 BPM, Estimated Meter = 4.0, $PADL_{90} \leq 2.5$, and $PASD_{90} \leq 2.5$. This combination of inclusion criteria retains 24.2% of the 10,000-file dataset (again, using $\theta_{Local} = 5.0\%$). Although this percentage may seem low, it is *scalable*. That is, assuming that the remainder of the Million Song Dataset yields similar distributions for the six Outcome Statistics, nearly 250,000 candidate songs could be made available for rhythmic synchronization paradigms (using these same inclusion criteria), and more still if the entire 34-million-item Echo Nest library were leveraged.

4. CONCLUSIONS, CAVEATS, AND FUTURE DIRECTIONS

We present a novel tool to evaluate auditory temporal stability (BEATS). An important departure that BEATS makes from previous methods is that it seeks to identify the most *temporally stable* segment within an inter-beat interval (IBeI) series for an entire audio file, rather than

derive a point estimate of tempo for the entire IBeI series. This increased flexibility enables BEATS to identify a greater number of candidate pieces of music that satisfy the requirements of rhythmic exercise applications.

4.1 Caveats

For ease of illustration in the present report, a single Local Stability Threshold (θ_{Local} = 5.0%) was used to instantiate BEATS and generate the associated figures. This value was chosen based on prior studies which have explored just-noticeable differences (JNDs) for changes in tempo (e.g., [3,8,19,23]), with reported values ranging from 10% (for single pairs of intervals) to 2% (for longer sequences). The stimuli in each of these cited studies, however, were all (1) isochronous (i.e., all intervals equally spaced in time), and (2) contained no more than 10 temporal intervals per sequence. Both factors limit the generalizability of these studies to actual extended excerpts of music, which is frequently (1) non-isochronous and (2) of a longer duration (enabling a stronger reference tempo to be formed, and thus a more finely tuned ability to detect change). θ_{Local} = 5.0% was chosen as a compromise, but warrants further experimental validation. That is, determining a threshold for "perceptually stable" in a non-isochronous IBeI series with varying degrees of local and global variability across trials (and across different tempo ranges) would greatly increase the utility of BEATS.

Another issue, highlighted by Figure 2, concerns the accuracy of the beat tracking algorithm itself. That is, BEATS is ignorant of the fidelity of the algorithm used to derive an inter-beat and inter-bar interval series. In the case of Figure 2, the derived IBeI series (as derived by the Echo Nest "Analyze" algorithm [9]) does not match the steady acceleration of tempo present within the audio file. Furthermore, preliminary exploration of the 10,000-item dataset suggests that highly complex or multi-layered rhythm loops that have an underlying perceptual pulse may nevertheless flummox a beat tracking algorithm.

Although this may mean that BEATS is conservative (in that it will classify some pieces of music as "temporally unstable" when they in fact may not be), such conservativeness may be beneficial in practice, as it will rule out pieces of music that may in fact be too challenging for listeners to synchronize with.

Alternatively, research from another sub-domain of audio content analysis, *score–performance matching* (e.g., [7,21]), may provide techniques to more robustly quantify changes in tempo over time, enhancing the ability of BEATS to detect excerpts of tempo stability.

4.2 Future Directions

By summarizing temporal stability using simple summary statistics, the output of BEATS can become the input to search engines for which tempo is a key feature (e.g., [4,14,15]). In its current state, however, BEATS is a work in progress. Our own future goals for this project include (1) implementing BEATS on much larger datasets (such as the entire Million Song Dataset, or even larger Echo Nest datasets), and (2) developing a high-quality web-based user interface ("iBEATS") that will offer visualizations (box plots, scatter plots) and flexible parameter settings (buttons and sliders) to efficiently sort and sift through large amounts of metadata (including artist, release date, and genre tags) to create customized playlists for clinical (e.g., gait rehabilitation) or commercial (e.g., rhythmic exercise) applications.

5. ACKNOWLEDGMENT

We thank three anonymous reviewers for helpful comments. This research is supported by the Singapore National Research Foundation under its *International Research Centre @ Singapore* funding initiative, and administered by the Interactive Digital Media Programme Office.

6. REFERENCES

[1] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," *in ISMIR 2011: Proceedings of the 12th International Society for Music Information Retrieval Conference*, October 24-28, 2011, Miami, Florida, 2011, pp. 591–596.

[2] S. Dixon, "An interactive beat tracking and visualisation system," in *Proceedings of the International Computer Music Conference*, Havana, Cuba, 2001, pp. 215–218.

[3] C. Drake and M. C. Botte, "Tempo sensitivity in auditory sequences: evidence for a multiple-look model," *Percept. Psychophys.*, vol. 54, no. 3, pp. 277–286, Sep. 1993.

[4] F. Gouyon, "Dance music classification: A tempo-based approach," in *Proceedings of the International Conference on Music Information Retrieval*, Barcelona, 2004.

[5] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *Audio Speech Lang. Process. Ieee Trans.*, vol. 14, no. 5, pp. 1832–1844, 2006.

[6] J. M. Hausdorff, “Gait dynamics in Parkinson’s disease: common and distinct behavior among stride length, gait variability, and fractal-like scaling,” *Chaos Woodbury N*, vol. 19, no. 2, p. 026113, Jun. 2009.

[7] H. Heijink, P. Desain, H. Honing, and L. Windsor, “Make me a match: An evaluation of different approaches to score—performance matching,” *Comput. Music J.*, vol. 24, no. 1, pp. 43–56, 2000.

[8] R. B. Ivry and R. E. Hazeltine, “Perception and production of temporal intervals across a range of durations: Evidence for a common timing mechanism,” *J. Exp. Psychol. Hum. Percept. Perform.*, vol. 21, no. 1, pp. 3–18, 1995.

[9] T. Jehan, “Analyzer Documentation.” 2011. http://developer.echonest.com/docs/v4/_static/AnalyzeDocumentation.pdf

[10] C. I. Karageorghis and D.-L. Priest, “Music in the exercise domain: a review and synthesis (Part I),” *Int. Rev. Sport Exerc. Psychol.*, vol. 5, no. 1, pp. 44–66, Mar. 2012.

[11] C. I. Karageorghis and D.-L. Priest, “Music in the exercise domain: a review and synthesis (Part II),” *Int. Rev. Sport Exerc. Psychol.*, vol. 5, no. 1, pp. 67–84, Mar. 2012.

[12] I. Lim, E. Van Wegen, C. De Goede, M. Deutekom, A. Nieuwboer, A. Willems, D. Jones, L. Rochester, and G. Kwakkel, “Effects of external rhythmical cueing on gait in patients with Parkinson’s disease: a systematic review,” *Clin. Rehabil.*, vol. 19, no. 7, pp. 695–713, 2005.

[13] J. Langner and W. Goebl, “Visualizing expressive performance in tempo-loudness space,” *Comput. Music J.*, vol. 27, no. 4, pp. 69–83, 2003.

[14] Z. Li, Q. Xiang, J. Hockman, J. Yang, Y. Yi, I. Fujinaga, and Y. Wang, “A music search engine for therapeutic gait training,” in *Proceedings of the international conference on Multimedia*, 2010, pp. 627–630.

[15] Z. Li and Y. Wang, “A domain-specific music search engine for gait training,” in *Proceedings of the 20th ACM international conference on Multimedia*, New York, NY, USA, 2012, pp. 1311–1312.

[16] B. H. Merker, G. S. Madison, and P. Eckerdal, “On the role and origin of isochrony in human rhythmic entrainment,” *Cortex*, vol. 45, no. 1, pp. 4–17, 2009.

[17] M. F. McKinney and D. Moelants, “Ambiguity in tempo perception: What draws listeners to different metrical levels?,” *Music Percept.*, vol. 24, no. 2, pp. 155–166, 2006.

[18] M. F. McKinney, D. Moelants, M. E. P. Davies, and A. Klapuri, “Evaluation of audio beat tracking and music tempo extraction algorithms,” *J. New Music Res.*, vol. 36, no. 1, pp. 1–16, 2007.

[19] N. S. Miller and J. D. McAuley, “Tempo sensitivity in isochronous tone sequences: the multiple-look model revisited,” *Percept. Psychophys.*, vol. 67, no. 7, pp. 1150–1160, 2005.

[20] B. Nettl, “An ethnomusicologist contemplates universals in musical sound and musical culture,” in *The origins of music*, B. Wallin, B. Merker, and S. Brown, Eds. Cambridge, MA: MIT Press, 2000, pp. 463–472.

[21] A. Robertson, “Decoding Tempo and Timing Variations in Music Recordings from Beat Annotations.,” in *ISMIR*, 2012, pp. 475–480.

[22] T. C. Rubinstein, N. Giladi, and J. M. Hausdorff, “The power of cueing to circumvent dopamine deficits: a review of physical therapy treatment of gait disturbances in Parkinson’s disease,” *Mov. Disord.*, vol. 17, no. 6, pp. 1148–1160, 2002.

[23] H. H. Schulze, “The perception of temporal deviations in isochronic patterns,” *Percept. Psychophys.*, vol. 45, no. 4, pp. 291–296, 1989.

[24] Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology, “Heart Rate Variability: Standards of Measurement, Physiological Interpretation, and Clinical Use,” *Circulation*, vol. 93, no. 5, pp. 1043–1065, Mar. 1996.

[25] M. H. Thaut, G. C. McIntosh, and R. R. Rice, “Rhythmic facilitation of gait training in hemiparetic stroke rehabilitation.,” *J. Neurol. Sci.*, vol. 151, no. 2, pp. 207–212, Oct. 1997.

[26] I. Winkler, G. P. Háden, O. Ladinig, I. Sziller, and H. Honing, “Newborn infants detect the beat in music,” *Proc. Natl. Acad. Sci.*, vol. 106, no. 7, pp. 2468–2471, 2009.

[27] J. E. Wittwer, K. E. Webster, and K. Hill, “Rhythmic auditory cueing to improve walking in patients with neurological conditions other than Parkinson’s disease-what is the evidence?,” *Disabil. Rehabil.*, vol. 35, no. 2, pp. 164–176, 2013.

[28] M. Zentner and T. Eerola, “Rhythmic engagement with music in infancy,” *Proc. Natl. Acad. Sci.* U.S.A., Mar. 2010.

Oral Session 7:

Symbolic Data Processing

SIARCT-CFP: IMPROVING PRECISION AND THE DISCOVERY OF INEXACT MUSICAL PATTERNS IN POINT-SET REPRESENTATIONS

Tom Collins, Andreas Arzt, Sebastian Flossmann, and Gerhard Widmer
Department of Computational Perception, Johannes Kepler University Linz
{tom.collins, andreas.arzt, sebastian.flossmann, gerhard.widmer}@jku.at

ABSTRACT

The geometric approach to intra-opus pattern discovery (in which notes are represented as points in pitch-time space in order to discover repeated patterns within a piece of music) shows promise particularly for polyphonic music, but has attracted some criticism because: (1) the approach extends to a limited number of inexact repetition types only; (2) typically geometric pattern discovery algorithms have poor precision, returning many false positives. This paper describes and evaluates a solution to the *inexactness problem* where algorithms for pattern discovery and inexact pattern matching are integrated for the first time. Two complementary solutions are proposed and assessed for the *precision problem*, one involving categorisation (hence reduction) of output patterns, and the second involving a new algorithm that calculates the difference between consecutive point pairs, rather than all point pairs.

1. INTRODUCTION

The discovery of repeated patterns within a piece of music is an activity that manifests itself in a range of disciplines. In music psychology, for example, listeners' emotional responses to a piece exhibit distinctive behaviour at the beginning of repeated sections [11]. In music analysis, an awareness of the locations of motifs, themes, and sections, and their relation to one another, is a prerequisite for writing about the construction of a piece [3]. Last but not least, in music computing, algorithmic pattern discovery can be used to define compressed representations [13] (e.g., the numeric pitch sequence 67, 68, 67, 69, 69, 66, 67, 66, 68, 68 can be encoded as 67, 68, 67, 69, 69, and a translation operation "-1") and can act as a guide for the algorithmic generation of new music [9]. In the interests of supporting these multiple manifestations, it is important that the field of music information retrieval continues to develop and refine algorithms for the discovery of repeated patterns, and continues to evaluate these against each other and human-annotated ground truths.

There are two main representations in use for discovering repeated patterns within a piece of music (hereafter *intra-opus discovery* [8]): (1) *viewpoints* [9] involve encoding multiple aspects of the music as strings of symbols (such as the numeric pitches mentioned above, or durations, intervals between notes, etc.). This approach has been applied mainly to monophonic music; (2) the *geometric approach* [14] involves converting each note to a point in pitch-time space (see the pitch-time pairs in Figures 1A and B). Higher-dimensional spaces are also possible (e.g., including dimensions for duration or staff number). The geometric approach is well-suited to handling polyphonic music, where few attempts have been made to apply viewpoints. This paper focuses on the geometric approach; specifically, ontime and *morphetic pitch number* [14] (C4 = C♯4 = 60, D♭4 = D♮4 = D♯4 = 61, E♭4 = E4 = 62, etc.).

Before getting into more details of related work, it is helpful to distinguish the terms *pattern matching* and *pattern discovery*. Typically in pattern matching, there is a short musical query and a longer piece (or pieces) of music, and the aim is to match the query to more or less exact instances in the piece(s) [2, 17]. In intra-opus pattern discovery there is no query, just a single piece of music, and the requirement to discover motifs, themes, and sections that are repeated within the piece [8, 14]. (One could say that the purpose of a pattern discovery algorithm is to *create* analytically interesting but hitherto unknown queries.) Pattern discovery and pattern matching have been discussed in the same papers [13], but nobody to our knowledge has integrated discovery and *inexact* matching components in one algorithm before. This full integration is one of the contributions of the current work, and the other consists of two complementary methods for improving the precision of pattern discovery algorithms. The paper is organised around describing and evaluating components of a new algorithm called SIARCT-CFP, beginning at the end of the acronym with "FP" for fingerprinting, then "C" for categorisation, and finally SIARCT, which stands for Structure Induction Algorithm for r superdiagonals and Compactness Trawler, which has been defined before [5] and for which a Matlab implementation has been released. [1]

2. THE INEXACTNESS PROBLEM

In reviewing the Structure Induction Algorithm (SIA) and other geometric pattern discovery algorithms (see [14] or [7] for details), Lartillot and Toiviainen noted that "this ge-

[1] http://www.tomcollinsresearch.net

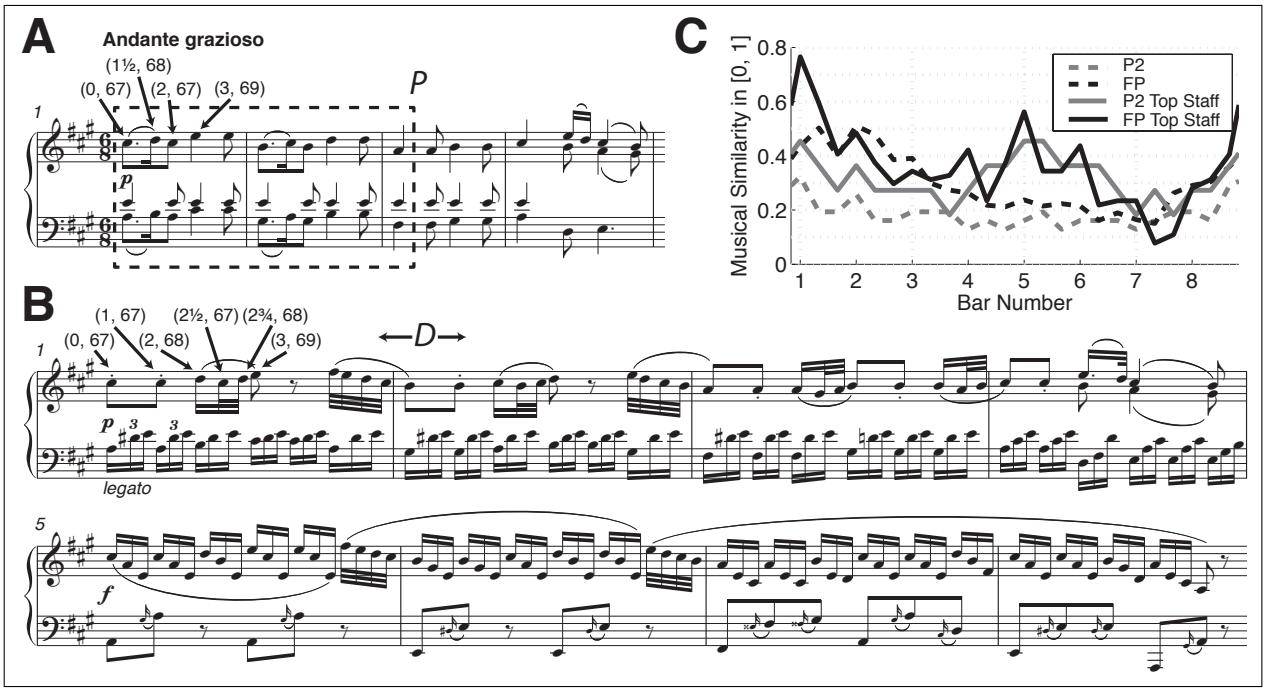


Figure 1. (A) Bars 1-4 of the Theme from the first movement of Piano Sonata no.11 in A major K331 by Wolfgang Amadeus Mozart (1756–1791). Labels give the ontime and morphetic pitch of the indicated note, and the box contains the top-rated pattern output by SIARCT; (B) Bars 1-8 of Variation II from the same movement; (C) Symbolic musical similarity of the pattern in (A) to the passage in (B), for two algorithms applied separately to the full texture and top staff only.

ometrical strategy did not apply to melodic repetitions that presented rhythmic variations" [10, pp. 290-291]. To illustrate this problem we use a theme by Mozart, from "one of the most overanalyzed pieces in the history of music theory" [15, p. 160]. We are not particularly interested in adding to discussions of the structure of the theme itself, rather in the relation of the theme to a subsequent variation. If the passage in Figure 1B were appended to the passage in Figure 1A and SIA applied to the single resulting point set, there would be little in the output to suggest that the first two bars of Figure 1B contain a variation on the bounded pattern P in Figure 1A. The points $\{(0, 67), (3, 69), (6, 66), (9, 68), (12, 65)\}$ would appear in the same output maximal translatable pattern (MTP, [14]), as they occur under the same translation in Figure 1B, but intervening points in the bounded pattern do not.

The pattern matching algorithm P2 [17] struggles with rhythmic variation also: for a given pattern P and a larger point set D, it returns all vector-frequency pairs $(\mathbf{w}, m)$ such that $m \geq 1$ points of P occur translated by $\mathbf{w}$ in D. We implemented P2 and used it to match P (from Figure 1A) to partial occurrences in D (Figure 1B). A summary of the output is plotted in Figure 1C, for both full-texture versions of P and D and a restriction to the right hand only (dashed and solid lines respectively). The maximal frequency M for pairs $(\mathbf{w}_1, m_1)_{i \in \{1,2,\ldots,s\}}$ corresponding to each crotchet-note ontime in D is plotted, normalised by the number of points in P, to give a measure of the symbolic musical similarity of P to D over time. While there are local maxima in the grey lines at bars 1, 2, and 5 (in the second case because P2 is transposition-invariant and there is a transposed pattern within P), in general they have a relatively small range, reflecting P2's struggle to distinguish genuine rhythmic variation from less related material.

Subsequent work on geometric pattern matching improves upon P2 in terms of capturing rhythmic variation, by representing durations as line segments [12, 17], by using the Hausdorff metric [16], or by converting to a tonal space representation [1]. A recent *fingerprinting* (FP) approach [2] has the advantage of not relying on durational information, and has options for transposition, time-shift, and scale-factor invariance, as well as tolerance for the amount by which the inter-onset interval of a pair of notes is permitted to differ, compared to a corresponding note pair in the original. The output of FP is a time series $S = S_t : t \in T$, where the set T of successive time points may or may not be uniformly spaced. The magnitude of S_t, called the *matching score*, indicates the extent to which an occurrence of the query begins at time t. In the transposition-invariant version, calculation of the matching score time series begins by creating fingerprint tokens

$$[y_j - y_i,\ x_j - x_i],\ \ t, \tag{1}$$

for locally constrained combinations of successive ontime-pitch pairs $(x_i, y_i), (x_j, y_j)$, in both a query pattern P and the larger point set D. The pair in brackets in (1) is the hash key, and $t = x_i$ is a time stamp. A scatter plot of the time stamps of matching hash keys for P and D can be used to identify regions of high similarity, which appear as approximately diagonal lines. The matching score is calculated by applying an affine transformation to the scatter plot and binning (for details, see [2, 18]).

An implementation of the FP algorithm was used to match exact/inexact occurrences of P from Figure 1A to D in Figure 1B, and the results are plotted in Figure 1C as black lines. It can be seen that FP outperforms P2 at distinguishing the rhythmic variation in bars 1-2 of Figure 1B. The use of locally constrained combinations of ontime-pitch pairs, rather than one candidate translation vector applied to all points in P, is what enables the FP algorithm to find a stronger match than P2.

Progress has been made in geometric pattern *matching* techniques, but Lartillot and Toiviainen's [10] criticism of the *discovery* approach still stands, as nobody to our knowledge has integrated an inexact matching technique within a pattern discovery approach. We do so now, according to the following steps, which define the "FP" part of SIARCT-CFP:

1. Let $P_1, P_2, \ldots, P_M$ be the output of a pattern discovery algorithm, each P_i having at least one translationally exact repetition (two occurrences) in D;

2. For $i = 1, 2, \ldots, M$, run the FP algorithm [2] on P_i and D, returning time points $t_1^{P_i}, t_2^{P_i}, \ldots, t_m^{P_i}$ at which there may be further exact/inexact occurrences of P_i, according to whether the value at $t_j^{P_i}$ is greater than some *similarity threshold* $c \in [0, 1)$.

Underlying this integration of pattern discovery and pattern matching is the following assumption, which we call the *translationally exact once* (TEO) hypothesis:

> If a piece of music contains multiple inexact occurrences of a perceptually salient or analytically interesting pattern, then for some majority subset of the pattern (i.e., a subset containing at least half of the points), there exists at least one translationally exact repetition (i.e., at least two occurrences).

If the discovery algorithm outputs such a majority subset, then the matching algorithm may be relied upon to output further exact/inexact occurrences of the pattern.

As a case study, the new algorithm SIARCT-CFP was run on the Nocturne in E major op.62 no.2 by Frédéric Chopin (1810–1849). [2] This is a sensible choice of piece, as it contains multiple variations of the opening theme (c.f. Figures 2B and D for instance). Fourteen patterns were output in total, one of which Q is bounded in Figure 2A, and occurs translated three times (bars 27–28, 58–59, and 60–61). These occurrences are rated as very similar to Q, with normalised matching scores close or equal to 1. The time series output by the FP has mean .264 and standard deviation .173, suggesting that the occurrence in Figure 2C is not distinguishable from other unrelated material. This makes sense, as although the contour and rhythm of the melody are as in Q, the pitch intervals are different (see arrows) and so is the accompaniment. We note, however, that

[2] The first part of the algorithm, SIAR, ran with parameter $r = 1$. Second, the compactness trawler (CT) ran with compactness threshold $a = 4/5$, cardinality threshold 10, and lexicographic region type [7]. Third, the categorising and fingerprinting (CFP) ran with similarity threshold $c = 1/2$.

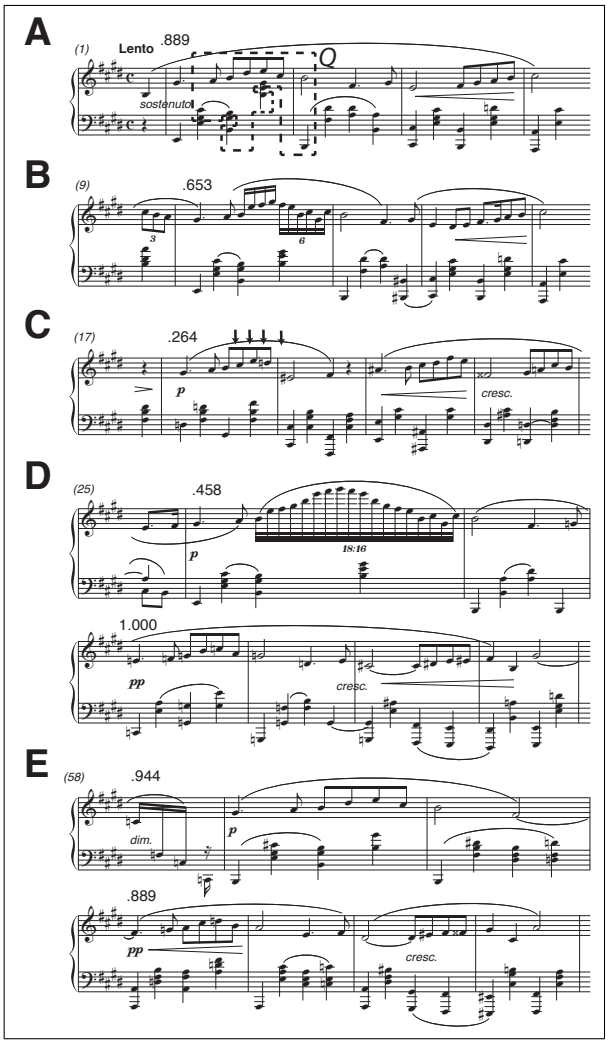


Figure 2. Excerpts from the Nocturne in E major op.62 no.2 by Chopin. Dashed lines in (A) bound a pattern Q discovered by SIARCT, which is used to match other inexact occurrences, with degree of exactness indicated in the figure by numbers in [0, 1]. Pedalling omitted for clarity.

the FP algorithm could be extended further to incorporate contour (up, down, same), as well as other viewpoints [9], because of its use of locally constrained comparisons.

3. THE PRECISION PROBLEM

3.1 Categorisation by Pattern Matching

Now that we have integrated some inexact pattern matching techniques into our pattern discovery approach, it is possible to employ them for the purposes of categorisation, based on the idea that P2 [17] or FP [2] can be used to compare two discovered patterns P_i and P_j in exactly the same way as if $P_i = P$ was a query and $P_j = D$ was a point set (or vice versa, as the measures are symmetric).

The second "C" in SIARCT-CFP stands for a **c**ategorisation process, which will be described now. The purpose of categorisation is to reduce an overwhelming amount of information (e.g., output patterns) to a more manageable number of exemplars. Here *categorisation* does not mean

classifying patterns into an accepted/interesting category versus a rejected/uninteresting category; rather it means grouping similar patterns and representing each group with one exemplar pattern. Our motivation for categorising the output of SIARCT is to improve its precision: while the precision and recall of pattern discovery algorithms has been shown to benefit from compactness trawling, the precision is still quite poor [7]. For example, SIARCT outputs 76 patterns when run on Chopin's op.62 no.2, which can be reduced to fourteen patterns by using the following categorisation process:

1. Let $P_1, P_2, \ldots, P_M$ be the output of a pattern discovery algorithm, sorted descending by a rating of perceived pattern importance [6], or some other ordering. Let $J = \{1, 2, \ldots, M\}$ index the patterns that are uncategorised currently;
2. For the most important uncategorised pattern, index $i = \min(J)$, calculate the maximum normalised matching scores $s(P_i, P_j)$ for each $j \in J$, $j \neq i$;
3. For each similarity score $s(P_i, P_j)$ that is greater than some specifiable similarity threshold $c \in [0, 1)$, place pattern P_j in the category for which P_i is the exemplar, and remove j from J;
4. Repeat steps 2 and 3 until either J has one element k, in which case define P_k to be an exemplar with category membership P_k, or otherwise J is empty;
5. For the purposes of algorithm evaluation, return only the exemplars $P_{i(1)}, P_{i(2)}, \ldots, P_{i(m)}$.

Depending on the choice of c, $m \ll M$. The categorisation process can be visualised with two similarity matrices (Figure 3). The matrix in Figure 3A contains the maximum normalised matching scores for each pair of 76 output patterns for Chopin's op.62 no.2, ordered as in step 1 above. The matrix in Figure 3B is a permutation of 3A, showing the categorised patterns ($c = .5$) in their fourteen categories, bounded by white squares. The fourth square from top-left in Figure 3B represents the category for which Q in Figure 2A is the exemplar. The fivefold ($5.43 \approx 76/14$) reduction in output achieved by pattern-matching categorisation may well improve precision: as discussed, the theme annotated in Figure 2A survives the categorisation process, and so do all of the repetitions in this piece lasting four or more bars (results not shown). Pattern-matching categorisation also constitutes a novel and interesting use of the FP algorithm [2]. It should be noted that choosing too low a value for c could lead to over-reduction and filtering out of analytically interesting patterns. For instance, the first two squares in Figure 3B show considerable variegation, suggesting that some interesting subcategories may be overlooked.

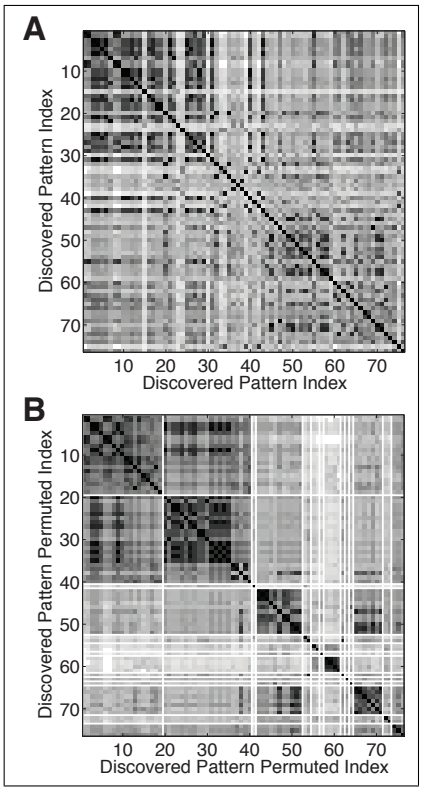


Figure 3. (A) Pairwise symbolic musical similarities (ranging from white for dissimilar to black for identical) for 76 patterns discovered by SIARCT in Chopin's op.62 no.2, ordered by a rating formula for perceived salience; (B) Permutation of the above matrix, with white lines indicating the results of categorising into fourteen groups.

3.2 Consecutive Points and Conjugate Patterns

The final novel contribution of this paper is to evaluate the SIARCT pattern discovery algorithm [5] against a collection of music containing repeated sections, and to compare its performance (especially precision) to SIA [14] and SIAR [5]. SIA outputs thousands of patterns for Chopin's op.62 no.2 (and other pieces of music [7]), so it is necessary to develop a more parsimonious pattern discovery algorithm for use as input to the categorisation and fingerprinting components described above (e.g., SIARCT outputs only 76 patterns for Chopin's op.62 no.2).

It has long been thought that in order to discover repeated patterns within a geometric representation D of a piece, it is necessary to calculate the difference between each pair of n points ($n[n-1]/2$ calculations in total), as in SIA [14]. Unlike SIA, the first step of SIARCT is to calculate the difference between consecutive pairs of points only ($n - 1$ calculations). Some exhaustive pairwise comparisons are still made in the second step, but for small, non-overlapping subsets of D, meaning that the total number of difference calculations performed by SIARCT is far less than $n[n-1]/2$, in all but one degenerate case. [3] The third step of SIARCT makes use of a concept known as *conjugate patterns* [5]: if a pattern containing l points occurs m times in a point set, then there exists in the same point set a pattern consisting of m points that occurs l times. The fourth step calculates MTPs for each vector in a list L. As a consequence of manipulating conjugate patterns, the vectors corresponding to repeated sections should be at or near

[3] Please see [5] for the algorithmic details.

the top of L. So for this step we could: (1) distribute each MTP calculation to parallel processors, and/or; (2) output MTPs dynamically for the user to browse, whilst calculation of the remaining MTPs continues. The main claim is that SIARCT will have much smaller output than SIA, with minimal or no negative impact on its performance as measured by precision, recall, and robust versions of these metrics [4]. The compactness trawler (CT) part of SIARCT is exactly as in [7], so is not addressed again here.

SIA, SIAR, and SIARCT were run on point-set representations of movements by Ludwig van Beethoven (1770–1827) and Chopin listed in Figure 4A. SIARCT ran with compactness threshold $a = 1$, and points threshold $b = 50$. This means that only patterns containing 50 points or more were returned, and they had to have maximal compactness of 1. The parameter values make sense in terms of trying to discover repeated sections. To make the evaluation fair, we also filtered the results of SIA and SIAR, returning only those patterns that contained 50 points or more. In the results, these versions of SIA and SIAR are referred to as SIA (50+) and SIAR (50+).

3.3 Evaluation Results

Figure 4B shows the log of the total number of patterns output by each algorithm for each movement/piece. It supports the claim that SIAR has a much smaller output than SIA. It is difficult to see from Figure 4B, but the same observation applies to the filtered versions of each algorithm, SIAR (50+) and SIA (50+). The number of patterns output by SIARCT is several orders of magnitude less than that of any other algorithm. Figure 4C and Figure 4E show that compared with SIA's performance, SIAR is not negatively impacted by restricting calculations to consecutive pairs of points. The establishment precision and establishment recall for SIAR and SIA are comparable across all pieces.

Overall, the most effective algorithm is SIARCT (see Figure 4C and Figure 4E). For half of the pieces, it discovers all ground truth patterns exactly (Figure 4F). When SIARCT fails to discover a ground truth pattern exactly, often this is due to a difference between the repeated section as written in the score, and the repeated pattern as heard in a performance. For instance, in the fourth movement of Beethoven's op.7, bars 65–70 are marked as a repeated section, and this is included in the ground truth. The repeated notes extend beyond these bars in both directions, however, creating a longer repeated pattern in a performance. SIARCT discovers the latter, *performed* pattern, which reduces exact precision and recall. The more robust *establishment* metrics are not much reduced (e.g., see Figure 4E), and arguably discovering the performed pattern is preferable from a music-perceptual point of view.

4. DISCUSSION AND FUTURE WORK

This paper identifies two valid reasons why the geometric approach to intra-opus pattern discovery has attracted some criticism—namely (1) the approach extends to a limited number of inexact repetition types only, and (2) typically geometric pattern discovery algorithms are imprecise, returning many false positives results. A new algorithm called SIARCT-CFP has been described and evaluated component-wise, in an attempt to address these criticisms. It is the first geometric pattern discovery algorithm to fully integrate an inexact pattern matching component (the fingerprinting algorithm of [2]), and this matching component was shown to be effective for retrieving inexact occurrences of themes in pieces by Mozart and Chopin. The comparison of the FP algorithm [2] to a baseline pattern matching algorithm P2 [17] demonstrated that the former was superior for a particular example. In general it may be preferable to have two or more pattern matchers at one's disposal, however, as the number of variation techniques is large, and trying to account for them all with one algorithm will likely produce false positive matches.

The precision metrics were of particular interest to us in the comparative evaluation of SIARCT [5], SIAR [5], and SIA [14], as we claimed that SIARCT could achieve levels of precision comparable to SIA and SIAR, without harming recall. This claim was supported by the evaluation results, although in future work it will be necessary to see if similar results are achieved for ground truths containing shorter patterns than repeated sections.

Our *translationally exact once* (TEO) hypothesis (see Section 2) was borne out in the case study of Chopin's op.62 no.2, where Q (Figure 2A) occurred exactly under translation (bars 27–28, 58–59, and 60–61), and its contents were sufficient for use as a query to retrieve less exact versions such as in bars 9 (Figure 2B) and 25 (Figure 2D). For the case study of the Theme section and Variation II from Mozart's K331, SIARCT was able to discover perceptually salient patterns such as P in Figure 1A, which recurs in bars 5-7 of the Theme section (not shown). As the TEO hypothesis holds in both cases, future work should focus on finding counterexample pieces, as this will help to refine and improve our underlying assumptions and ensuing algorithms. Future work will also attempt to show users/developers the differences between themes and partial matches, and to identify variation techniques (triplets, minore, etc.) automatically.

5. ACKNOWLEDGEMENTS

This paper benefited from the use of Kern Scores, and helpful discussions with David Meredith. We would like to thank four anonymous reviewers for their comments. This work is supported by the Austrian Science Fund (FWF), grants Z159 and TRP 109.

6. REFERENCES

[1] T. Ahonen, K. Lemström, and S. Linkola: "Compression-based similarity measures in symbolic, polyphonic music," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 91–96, 2011.

[2] A. Arzt, S. Böck, and G. Widmer: "Fast identification of piece and score position via symbolic fingerprint-

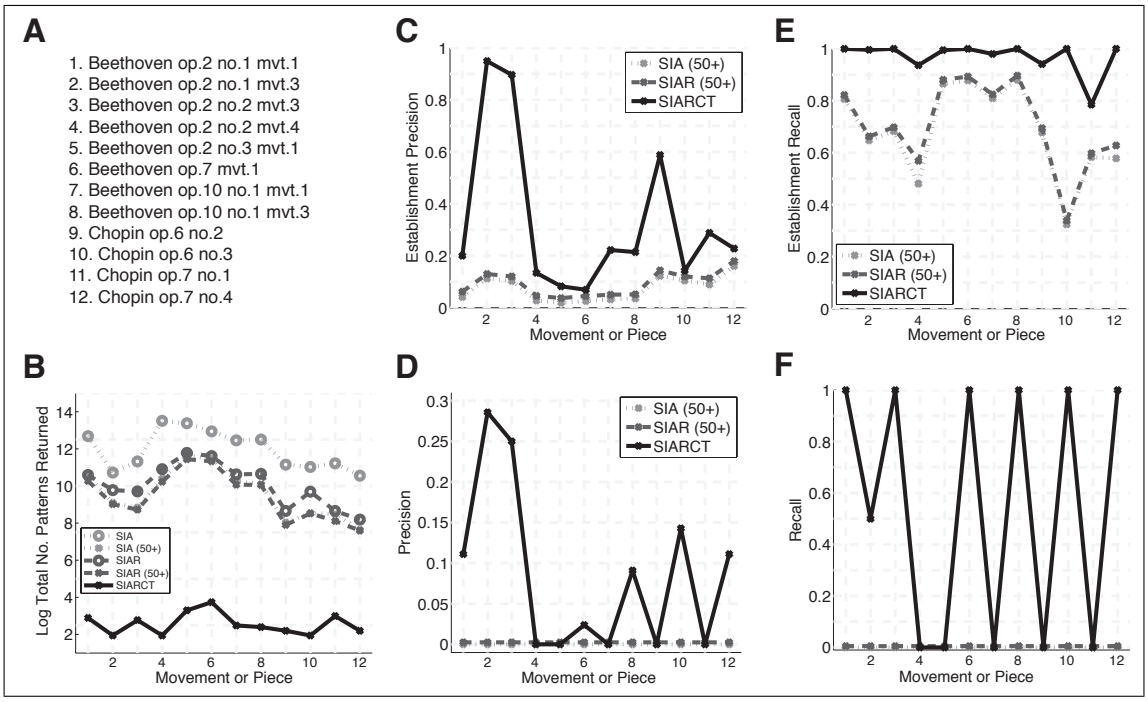


Figure 4. Evaluation metrics for three algorithms, run on eight movements by Beethoven and four pieces by Chopin.

ing," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 433–438, 2012.

[3] I. Bent and A. Pople: "Analysis," *The new Grove dictionary of music and musicians*, Macmillan, London, 2001.

[4] T. Collins: "Discovery of repeated themes and sections," Retrieved 4th May 2013, from http://www.music-ir.org/mirex/wiki/2013: Discovery_of_Repeated_Themes_%26_Sections

[5] T. Collins: *Improved methods for pattern discovery in music, with applications in automated stylistic composition*, PhD thesis, Faculty of Mathematics, Computing and Technology, The Open University, 2011.

[6] T. Collins, R. Laney, A. Willis, and P. Garthwaite: "Modeling pattern importance in Chopin's mazurkas," *Music Perception*, Vol. 28, No. 4, pp. 387–414, 2011.

[7] T. Collins, J. Thurlow, R. Laney, A. Willis, and P. Garthwaite: "A Comparative Evaluation of Algorithms for Discovering Translational Patterns in Baroque Keyboard Works," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 3–8, 2010.

[8] D. Conklin and C. Anagnostopoulou: "Representation and discovery of multiple viewpoint patterns," *Proceedings of the International Computer Music Conference*, pp. 479–485, 2001.

[9] D. Conklin and I. Witten: "Multiple viewpoint systems for music prediction," *Journal of New Music Research*, Vol. 24, No. 1, pp. 51-73, 1995.

[10] O. Lartillot and P. Toiviainen: "Motivic matching strategies for automated pattern extraction," *Musicae Scientiae*, Discussion Forum 4A, pp. 281–314, 2007.

[11] S. Livingstone, C. Palmer, and E. Schubert: "Emotional response to musical repetition," *Emotion*, Vol. 12, No. 3, pp. 552–567, 2012.

[12] A. Lubiw and L. Tanur: "Pattern matching in polyphonic music as a weighted geometric translation problem," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 154–161, 2004.

[13] D. Meredith: "Point-set algorithms for pattern discovery and pattern matching in music," *Proceedings of the Dagstuhl Seminar on Content-Based Retrieval*, 2006.

[14] D. Meredith, K. Lemström, and G. Wiggins: "Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music," *Journal of New Music Research*, Vol. 31, No. 4, pp. 321–345, 2002.

[15] E. Narmour: "Some major theoretical problems concerning the concept of hierarchy in the analysis of tonal music," *Music Perception*, Vol. 1, No. 1, pp. 129–199, 1983.

[16] C. Romming and E. Selfridge-Field: "Algorithms for polyphonic music retrieval: the Hausdorff metric and geometric hashing," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 457–462, 2007.

[17] E. Ukkonen, K. Lemström, and V. Mäkinen: "Geometric algorithms for transposition invariant content-based music retrieval," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 193–199, 2003.

[18] A. Wang: "An industrial strength audio search algorithm," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 7–13, 2003.

A MACHINE LEARNING APPROACH TO VOICE SEPARATION IN LUTE TABLATURE

Reinier de Valk **Tillman Weyde** **Emmanouil Benetos**

Music Informatics Research Group
Department of Computer Science
City University London
{r.f.de.valk,t.e.weyde,Emmanouil.Benetos.1}@city.ac.uk

ABSTRACT

In this paper, we propose a machine learning model for voice separation in lute tablature. Lute tablature is a practical notation that reveals only very limited information about polyphonic structure. This has complicated research into the large surviving corpus of lute music, notated exclusively in tablature. A solution may be found in automatic transcription, of which voice separation is a necessary step. During the last decade, several methods for separating voices in symbolic polyphonic music formats have been developed. However, all but two of these methods adopt a rule-based approach; moreover, none of them is designed for tablature. Our method differs on both these points. First, rather than using fixed rules, we use a model that learns from data: a neural network that predicts voice assignments for notes. Second, our method is specifically designed for tablature—tablature information is included in the features used as input for the models—but it can also be applied to other music corpora. We have experimented on a dataset containing tablature pieces of different polyphonic textures, and compare the results against those obtained from a baseline hidden Markov model (HMM) model. Additionally, we have performed a preliminary comparison of the neural network model with several existing methods for voice separation on a small dataset. We have found that the neural network model performs clearly better than the baseline model, and competitively with the existing methods.

1. INTRODUCTION

The lute, an instrument widely used from the early sixteenth to the mid-eighteenth century, has left us with a considerable corpus of instrumental polyphonic music: over 860 print and manuscript sources survive, containing approximately 60,000 pieces [12]. This music is notated exclusively in lute tablature. Lute tablature is a practical notation that provides no direct pitch information and only limited rhythmic information, but instead instructs the player where to place the fingers on the fretboard and which strings to pluck (see Figure 1). It reveals very little about the polyphonic structure of the music it encodes, since it specifies neither to which polyphonic voice the tablature notes belong, nor what their individual durations are. Lute tablature's "alien nature" [5] is the principal reason why, apart from a number of specialist studies, this large and important corpus has so far escaped systematic musicological research.

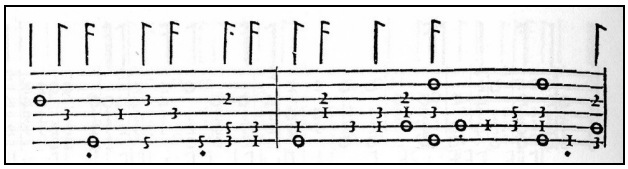


Figure 1. Excerpt of lute tablature in Italian style.

Transcription into modern music notation—a format much more familiar to the twenty-first-century scholar or musician—will increase the accessibility of the corpus, and, in fact, is the current modus operandi among those studying lute music. Transcribing tablature, however, is a time-consuming and specialist enterprise. Automatic transcription into modern music notation may provide a solution. An important step in the process of (automatic) transcription of polyphonic music is voice separation, i.e., the separation of the individual melodic lines ('voices') that together constitute the polyphonic fabric. Using machine learning techniques, we have developed two models for voice separation in lute tablature—a neural network model and a baseline hidden Markov model (HMM) model—which, with some modifications, can also be applied to other music corpora.

The outline of this paper is as follows: in Section 2, the existing methods for voice separation are discussed. In Section 3 the proposed models are introduced, and in Section 4 the dataset is presented. Section 5 is dedicated to the evaluation of the models; in Section 6 the results are discussed; and in Section 7 the performance of the neural network model is compared with that of several existing methods. Concluding thoughts are presented in Section 8.

2. RELATED WORK

During the last decade, several methods for separating voices in symbolic polyphonic music formats have been developed.[1] Except for two, described further below, all of these methods are rule-based. More concretely, they are based on at least one of two fundamental perceptual principles that group notes into voices, which have been

[1] In addition, a number of methods for voice separation in music in audio format exist—these, however, are left out of consideration here.

labelled the Pitch Proximity Principle and the Temporal Continuity Principle by Huron [6]. These principles imply that the closer notes are to one another in terms of pitch or time, respectively, the more likely they are perceived as belonging to the same voice. In addition, some of the methods include supplementary perceptual principles. Although these methods vary considerably in their approach, in each of them, the perceptual principles it is based on guide the voice assignment procedure.

Temperley [17] adopts an approach based on four 'preference rules,' i.e., criteria to evaluate a possible analysis. Two of these match the abovementioned principles; the other two prescribe to minimise the number of voices (New Stream Rule) and to avoid shared notes (Collision Rule). Cambouropoulos [1] briefly describes an elementary version of a voice separation algorithm based on the (Gestalt) principle of pitch proximity only. Chew and Wu [4] use a 'contig' approach, in which the music is divided into segments where a constant number of voices is active (the contigs). The voice fragments in the segments are then connected on the basis of pitch proximity; voice crossings are forbidden. Szeto and Wong [16] consider voices to be clusters containing events proximal in the pitch and time dimensions, and model voice separation as a clustering problem. The aim of their research, however, is to design a system for pattern matching, and not one for voice separation. In their method, voice separation is only a pre-processing step that prevents "perceptually insignificant stream-crossing patterns" from being returned by the system. Kilian and Hoos [9] present an algorithm that is not intended primarily for correct voice separation, but rather for creating "reasonable and flexible score notation." Their method allows for complete chords in a single voice. In the method presented by Karydis *et. al* [8], too, a 'voice' is not necessarily a "monophonic sequence of successive non-overlapping notes" [2]. Rather, they prefer to use the term 'stream,' which they define as "a perceptually independent voice consisting of single or multi-note sonorities." Hence, in addition to the 'horizontal' pitch and time proximity principles, they include two 'vertical integration' principles into their method: the Synchronous Note Principle (based on Huron's Onset Synchrony Principle) and the Principle of Tonal Fusion (based on Huron's Tonal Fusion Principle). A new version of this algorithm is described in Rafailidis et al. [14]. Madsen and Widmer [11], lastly, present an algorithm based primarily on the pitch proximity principle, with some heuristics added to handle unsolved situations.

In the remaining two methods, then, machine learning techniques are used. Kirlin and Utgoff [10] describe a system that consists of two components: a 'predicate,' implemented as a learned decision tree, that determines whether or not two notes belong to the same voice, and a hard-coded algorithm that then maps notes to voices. Jordanous [7] adopts a probabilistic approach based on a Markov model, and presents a system that learns the probability of each note belonging to each voice, as well as the probability of successive note pairs belonging to the same voice.

In addition to these more recent methods, another rule-based method—one designed specifically for automatic transcription of German lute tablature—was developed as early as the 1980s by Charnassé and Stepien [3]. In their research an approach was followed that combines expert knowledge encoded as rules with simpler heuristics. Although the results appear to be promising, the research seems to have ended prematurely.

3. PROPOSED MODELS

We have implemented two models for voice separation in tablature. The first uses a discrete hidden Markov model [13] to predict voice assignments for complete chords; the second uses a neural network (NN) to predict voice assignments for individual notes. The HMM model, in which the tablature chords are the only observations, is straightforward and functions as a baseline model to compare the neural network model against.

In our method, as in most existing methods, we use the notion of voice as a monophonic sequence of notes. In contrast to most rule-based methods, however, we allow voice crossings and shared notes (notes where two voices meet at the unison), both of which are perceptually problematic, but encountered frequently in polyphonic lute music. (This goes in particular for shared notes, which, especially in denser polyphony, are difficult to realise technically on the lute. Although actual unisons are sometimes used, a more idiomatic solution is to finger only one note of the unison—a technique also witnessed in keyboard music. Such notes shall henceforth be referred to as 'shared single notes.') Furthermore, unlike most existing methods, we assume in advance a maximum number of possible voices (five).[2]

3.1 HMM Model

We have used an HMM model in which the observations are the tablature chords, and the hidden states are the voice assignments. Each chord c is represented by a vector of pitches (MIDI numbers), depending on the number of notes in the chord ranging in length from 1 to 4; each voice assignment q_t for a given time frame t is represented by a vector of length 4. Here, each vector index represents a voice and can take the values -1, … , 3, where -1 denotes inactivity of the voice, and one of the other numbers the sequence number in the chord of the pitch that is assigned to that voice.

For each training set used in cross-validation, we have created a transition probability matrix $P(q_{t+1}|q_t)$, denoting the probability of having transitions between various voice assignments, an observation probability matrix $P(c_t|q_t)$, denoting the probability of encountering chord c_t given voice assignment q_t, and an initial state distribution $P(q_1)$. Since a training set might contain no instances of certain chord-voice assignment combinations, we have modified $P(c_t|q_t)$ by including a small non-zero

[2] Because of technical and physical limitations of lute and lutenist, more voices are rare in lute music.

probability for all cases where the number of pitches in a chord is the same as the number of assigned pitches in a voice assignment. This way, we discourage the prediction of voice assignments in which too few or too many pitches are assigned. Finally, the optimal voice assignment sequence is computed using the Viterbi algorithm [13].

It should be noted here that our HMM model is similar to Jordanous's system, as described in [7]. Firstly, both are probabilistic approaches, and second, only pitch-related observations from the training data are used. The main difference between her system and our HMM model is that in the former, a Markov chain with an ad-hoc cost function based on learned transition probabilities is used. Jordanous herself notes that "[i]t would be interesting to apply Hidden Markov Models . . . so that more of the previously allocated notes can be used to assist in voice allocation."

3.2 Neural Network Model

In the neural network model, the task of voice separation is modelled as a classification problem where every tablature note is assigned to a voice—or, in the case of a shared single note, to two voices. We used a standard feed-forward neural network with resilient backpropagation (Rprop) [15] and sigmoid activation function, which provides a proven fast and robust learning model.[3] The network consists of an input layer of 32 neurons, one hidden layer of 8 neurons, and an output layer of five neurons, each of which represents a voice. Having five output neurons enables us to use the network for five-voice lute music; however, because we are currently using a four-voice dataset, at the moment the fifth neuron is never activated. Using the sigmoid function, the individual output neurons all have activation values between 0 and 1; the neuron that gives the highest activation value determines the voice assignment decision. Prior to the actual training and testing, we have optimised the regularisation parameter λ (0.00003) and the number of hidden neurons (8) using a cross-validated grid search.

Using cross-validation and regularisation, we have trained in three runs, where each run consisted of 200 training epochs and the network weights were re-initialised randomly at the start of each run. The model from the training run in which the lowest error rate (see Section 5) was obtained, was selected for the validation stage.

In the validation stage, the model traverses the tablature note by note, from left to right (always starting with the lowest note in a chord), and assigns the notes to voices. The test process is linear, and previous voice assignments are not revised—except when an assignment conflict arises within a chord, i.e., when a note is assigned to a voice that was already assigned a note in the chord. Because we do not allow two notes within a chord to be assigned to the same voice, conflicts are solved using a heuristic that reassigns the current note to a yet unassigned voice. Since we have encountered only two conflicts in our experiments, we will not go into further details on this heuristic here. We assume that the low number of conflicts is due to the fact that the voices already assigned in the chord are given as a feature to the network (see next section).

3.2.1 Features

A 32-dimensional feature vector is generated for each tablature note, which contains two types of information (see Table 1). Features 1-12 contain only tablature information, and consist of (a) features encoding instrument-technical properties of the note (1-8), and (b) features encoding information about the position of the note within the chord (9-12). Features 13-32 contain information about the note's polyphonic embedding: (c) pitch and time proximities of the note to the previous note in each voice at the current onset time (13-27), and (d) the voices that have already been assigned to previous notes in the chord (28-32).

Three things should be noted here. First, features 13-27 encode, in essence, the principles that were labelled Pitch Proximity- and Temporal Continuity Principle by Huron [6]. Second, for the calculation of features 13-32, in addition to tablature information, voice assignment information is needed. Third, the time window within which the information is extracted that is used for the voice assignment decision, is presently still rather limited as it reaches back only one note per voice.

Tablature information	
Note information	7. isOrnamentation
1. pitch	8. isOpenCourse
2. course	**Chord information**
3. fret	9. numberOfNotesBelow
4. minDuration	10. numberOfNotesAbove
5. maxDuration	11. pitchDistanceToNoteBelow
6. chordSize	12. pitchDistanceToNoteAbove
Polyphonic embedding information	
Pitch/time proximities	23-27. offsetOnsetProx
13-17. pitchProx	**Voices already assigned**
18-22. interOnsetProx	28-32. voicesAlreadyAssigned

Table 1. Features for the NN model.

4. DATASET

At the moment, we are focusing on sixteenth-century lute music—more specifically, on intabulations, lute arrangements of polyphonic vocal pieces. There are three reasons for this choice. First, intabulations are highly representative of the entire sixteenth-century corpus since they then formed the predominant lute genre. Second, since the densest polyphonic structures in lute music are found in intabulations, they constitute a sub-corpus that is challenging for our research. Third, the use of intabulations provides an objective way of devising a ground truth by polyphonically aligning the tablature and the vocal pieces, whose voices are always notated separately. We have thus transcribed a number of carefully selected intabulations into modern music

[3] We use the implementation provided by the Encog framework. See http://www.heatonresearch.com/encog (accessed May 2013).

notation, and then converted these to MIDI, storing each voice in a separate file. The tablature encoding (in .txt format), together with the MIDI representation of the ground truth, are given as input to the model.

The dataset currently consists of nine intabulations, all for four voices (the most common intabulation format), and contains pieces of different polyphonic texture: three imitative pieces, three 'semi-imitative' pieces (pieces that contain points of imitation, but whose structure is not governed by them), and three free pieces. It comprises a total of 8892 notes divided over 5156 chords, single-note chords included (Table 2).

Piece	Texture	Notes	Chords
Ochsenkun 1558, *Absolon fili mi*	imitative	1184	727
Ochsenkun 1558, *In exitu Israel de Egipto*	imitative	1974	1296
Ochsenkun 1558, *Qui habitat*	imitative	2238	1443
Rotta 1546, *Bramo morir*	free	708	322
Phalèse 1547, *Tant que uiuray*	free	457	228
Ochsenkun1558, *Herr Gott laß dich erbarmen*	free	371	195
Abondante1548, *Mais mamignone*	semi-imitative	705	316
Phalèse 1563, *Las on peult*	semi-imitative	777	395
Barbetta 1582, *Il nest plaisir*	semi-imitative	478	234
Totals		8892	5156

Table 2. The dataset used for the experiments.

5. EVALUATION

5.1 Evaluation Metrics

Our main evaluation metric is the error rate, which is the percentage of notes assigned to an incorrect voice. The error rate is calculated by comparing, for each note, the predicted voice assignment with the ground truth voice assignment. For the NN model, we use two modes of evaluation. In *test* mode, we calculate the feature vectors with which the model is evaluated using the ground truth voice assignments. In *application* mode, which corresponds to the 'real-world situation' where the ground truth voice assignments are not provided, we calculate the feature vectors using the voice assignments predicted by the model. In application mode errors can propagate—once a note has been assigned to the wrong voice(s), this will influence the decision process for the assignment of the following notes or chords—typically resulting in higher error values. We thus distinguish between the *test error*, which is the error rate in test mode, and the *application error*, the error rate in application mode. For the HMM model, we evaluate using only a single metric that corresponds to the application error in the NN model.

Furthermore, for both models we use a tolerant and a strict approach for calculating errors—a distinction that applies to how shared single notes are handled. We distinguish between fully correct assignments (C), fully incorrect assignments (I) and three additional mixed categories: one voice assigned correctly but the other overlooked (O); one voice assigned correctly but another assigned superfluously (S); and one voice assigned correctly but the other assigned incorrectly (CI). All possibilities are listed in Table 3. In the tolerant evaluation approach, then, O, S, and CI are not counted as errors; in the strict approach they are counted as 0.5 errors.

P(*n*)	G(*n*)	Possibility	Error category				
			C	O	S	CI	I
1	1	P is G	X				
		P is not G					X
1	2	P is one of G		X			
		P is none of G					X
2	1	one of P is G			X		
		none of P is G					X
2	2	both P are G	X				
		one of P is G				X	
		none of P are G					X

Table 3. Error categories (P(*n*) = predicted voice(s) for note *n*; G(*n*) = ground truth voice(s) for note *n*).

5.2 Results

We have trained and evaluated both models on the complete dataset using nine-fold cross-validation, where the folds correspond to the individual pieces in the dataset and each piece serves as test set once. The results are given in Table 4.

Tolerant approach		Strict approach	
Error (%)	**Std. dev.**	**Error (%)**	**Std. dev.**
NN model, test error			
11.52	3.41	12.87	3.63
NN model, application error			
19.37	5.43	20.67	5.61
HMM model, application error			
24.95	6.59	25.64	6.69

Table 4. Averaged error rates (weighted) and standard deviation in cross-validation.

6. DISCUSSION

The performance of the models is compared by means of the application error rates. We see that the NN model outperforms the HMM model by about 5 percentage points—both when the tolerant and when the strict approach is applied. While the application error gives a realistic idea of how well the NN model actually performs, it is also interesting to have a look at the test error, which reflects the performance of the model when 'perfect' context information—context information derived directly from the ground truth voice assignments—is provided. A comparison of the test and application mode informs us about error propagation in the application mode. On the individual pieces, the test

errors are approximately between one half and two-thirds the size of the application errors, meaning that each misassigned note propagates 0.5-1.0 times. The high application errors might be explained at least partly by the observation that the pieces with high application errors contain many longer ornamental runs consisting of single notes, which are highly characteristic for lute music. Thus, when the first note of such a run is assigned to an incorrect voice, the following notes are very likely to be assigned to that voice as well. Because in such cases all notes are considered incorrect, single errors can propagate dramatically. However, the run as a whole will be assigned to a single voice, which is still a musically reasonable choice. This can be reflected using different evaluation metrics such as soundness and completeness (see Section 7).

We also observe that both models have problems handling shared single notes. In the NN model, 118 of the 129 shared single notes in the ground truth are assigned to only a single voice in test mode, and 114 in application mode. Moreover, 120 notes are superfluously assigned to a second voice in test mode, and 117 in application mode. We are currently using a simple heuristic to determine whether a note should be assigned to two voices: if the second highest activation value in the network output does not deviate more than 5.0% (the 'deviation threshold') from the highest activation value, the note is assigned to both corresponding voices. Although the current threshold leads to balanced results (118/114 shared single notes assigned erroneously to a single voice, versus 120/117 non-shared notes assigned superfluously to two), the method for determining shared single notes could be improved. In the HMM model, then, the number of shared single notes assigned erroneously to a single voice is in the same range (95); the number of notes assigned superfluously to two voices, however, is much lower (27). With respect to handling shared single notes, the HMM model overall thus performs better.

Voice crossings constitute another problem. An informal inspection shows that, in both models, most voice crossings are not detected. In the NN model, the main reason for this is that our features by design provide little support for voice crossings. This might be improved by including a 'melodic Gestalt criterion' in the form of features that represent melodic shape in the model. The inclusion of such features goes hand in hand with an increase of the information extraction window.

7. COMPARISON

We have compared our NN model with several of the existing methods for voice separation for which results and evaluation metrics are documented [4, 7, 10, 11, 14]. Using the same cross-validated procedure as above, but now excluding tablature-specific features such as course and fret, we have trained and tested the NN model on a small dataset that is comparable to those used in the above methods, and then evaluated the results using the different evaluation metrics proposed. It must be noted that the results of the comparison are only indicative, as the datasets used are similar but not identical and not all evaluation metrics are defined in detail.

Our dataset consists of the first five three-voice and the first five four-voice fugues of book I of Johann Sebastian Bach's *Wohltemperirtes Clavier*. [4] This collection of 48 preludes and fugues has been used, in total or in part, as the test set in most other methods we compare with—the only exception being the one described in [10], where the model is trained and tested on excerpts of the (stylistically comparable) chaconne from Bach's second violin partita (BWV 1004).

To enable a comparison we use five evaluation metrics: precision and recall, defined in [7] as "the percentage of notes allocated to a voice that correctly belong to that voice" (precision) and "the percentage of notes in the voice that are successfully allocated to that voice" (recall); soundness and completeness, defined in [10] as the percentage of adjacent note pairs in a predicted voice of which both notes belong to the same ground truth voice (soundness) and, conversely, the percentage of adjacent note pairs in a ground truth voice of which both notes have been assigned to the same predicted voice (completeness); and Average Voice Consistency (AVC) as used by [4], which measures, "on average, the proportion of notes from the same voice that have been assigned . . . to the same voice."

	Dataset	**Evaluation metric (%)**				
		P	R	S	C	A
NN	10 fugues (3-4vv)	83.12	83.12	94.07	93.42	82.67
[4]	48 fugues (3-5vv)					84.39
[7]	45 fugues (3-4vv)	80.88	80.85			
[10]	Bach chaconne			88.65	65.57	
[11]	30 Bach *Inventions* (2-3vv); 48 fugues (3-5vv)			95.94	70.11	
[14]	4 fugues (3-4vv)		92.50			

Table 5. Comparison of the NN model with other methods (P = precision; R = recall; S = soundness; C = completeness; A = Average Voice Consistency).[5]

As can be seen in Table 5, the results obtained by our NN model are in a similar range as those reported for the other models, and at times better. Moreover, with an application error of 16.87% (and a test error of 4.00%), the NN model performs better than on tablature (cf. Table 4).

[4] The dataset (in the form of MIDI files) was retrieved from www.musedata.org (accessed July 2013).

[5] In [11] it is stated that soundness and completeness "as suggested by Kirlin [and Utgoff]" were used as evaluation metrics; however, the textual definitions given differ. We have not yet been able to clarify this inconsistency, so we present the numbers and metrics exactly as in [11]. [14] use 'accuracy' as metric, whose definition matches that of recall.

8. CONCLUSIONS AND FUTURE WORK

In this paper we propose a neural network model for voice separation in lute tablature. This model is more flexible than the existing rule-based models in that it adapts to the data, and thus is less restricted with regard to what needs to be fixed as a priori rules. The model clearly outperforms the baseline HMM model and also seems to be more robust. In addition, it performs apparently competitively with the existing voice separation methods we have compared it with; however, extended tests will be needed for a systematic comparison. Although there is still room for improvement, the results are sufficiently promising to continue experimenting—not only with NN models, but also with different HMM models. Issues that need to be solved in particular are the high error propagation in the NN model's application mode, which currently complicates a real-world application, the handling of shared single notes, and the detection of voice crossings.

In future work, we will therefore extend the current NN model by including more features and by expanding the information extraction window. Additionally, we have started working on an approach that does not assign individual notes, but rather complete chords, to voices. With regard to the HMM model, we will experiment with more complex models using Gaussian mixture HMMs and factorial HMMs. Lastly, we are planning to work towards a more comprehensive and rigorous comparison of voice separation methods.

9. ACKNOWLEDGEMENTS

Reinier de Valk is supported by a City University London PhD Studentship and Emmanouil Benetos is supported by a City University London Research Fellowship.

10. REFERENCES

[1] E. Cambouropoulos: "From MIDI to Traditional Musical Notation," *Proceedings of the AAAI Workshop on Artificial Intelligence and Music*, n.p., 2000.

[2] E. Cambouropoulos: "'Voice' Separation: Theoretical, Perceptual and Computational Perspectives," *Proceedings of the 9th International Conference on Music Perception and Cognition*, pp. 987-997, 2006.

[3] H. Charnassé and B. Stepien: "Automatic Transcription of German Lute Tablatures: An Artificial Intelligence Application," *Computer Representations and Models in Music*, Ed. A. Marsden and A. Pople, Academic Press, London, pp. 144-70, 1992.

[4] E. Chew and X. Wu: "Separating Voices in Polyphonic Music: A Contig Mapping Approach," *Computer Music Modeling and Retrieval: Second International Symposium, Revised Papers*, Ed. U. K. Wiil, Springer, Berlin, pp. 1-20, 2004.

[5] J. Griffiths: "The Lute and the Polyphonist," *Studi Musicali*, Vol. 31, No. 1, pp. 89-108, 2002.

[6] D. Huron: "Tone and Voice: A Derivation of the Rules of Voice-Leading from Perceptual Principles," *Music Perception*, Vol. 19, No. 1, pp. 1-64, 2001.

[7] A. Jordanous: "Voice Separation in Polyphonic Music: A Data-Driven Approach," *Proceedings of the International Computer Music Conference*, n.p., 2008.

[8] I. Karydis et al.: "Horizontal and Vertical Integration/Segregation in Auditory Streaming: A Voice Separation Algorithm for Symbolic Musical Data," *Proceedings of the 4th Sound and Music Computing Conference*, pp. 299-306, 2007.

[9] J. Kilian and H. Hoos: "Voice separation—A Local Optimisation Approach," *Proceedings of the 3rd International Conference on Music Information Retrieval*, n.p., 2002.

[10] P. Kirlin and P. Utgoff: "VoiSe: Learning to Segregate Voices in Explicit and Implicit Polyphony," *Proceedings of the 6th International Conference on Music Information Retrieval*, pp. 552-557, 2005.

[11] S. T. Madsen and G. Widmer: "Separating Voices in MIDI," *Proceedings of the 7th International Conference on Music Information Retrieval*, n.p., 2006.

[12] A. J. Ness and C. A. Kolczynski: "Sources of Lute Music," *The New Grove Dictionary of Music and Musicians*, 2nd ed., Ed. S. Sadie, Macmillan, London, pp. 39-63, 2001.

[13] L. R. Rabiner: "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, Vol. 77, No. 2, pp. 257-286, 1989.

[14] D. Rafailidis, E. Cambouropoulos, and Y. Manolopoulos: "Musical Voice Integration/ Segregation: *VISA* Revisited," *Proceedings of the 6th Sound and Music Computing Conference*, pp. 42-47, 2009.

[15] M. Riedmiller and H. Braun: "RPROP—A Fast Adaptive Learning Algorithm," *Proceedings of the International Symposium on Computer and Information Science*, n.p., 1992.

[16] W. M. Szeto and M. H. Wong: "Stream Segregation Algorithm for Pattern Matching in Polyphonic Music Databases," *Multimedia Tools and Applications*, Vol. 30, pp. 109-127, 2006.

[17] D. Temperley: *The Cognition of Basic Musical Structures*, The MIT Press, Cambridge, MA, 2001.

A METHODOLOGY FOR THE COMPARISON OF MELODIC GENERATION MODELS USING META-MELO

Nicolas Gonzalez Thomas, Philippe Pasquier, Arne Eigenfeldt, James B. Maxwell
MAMAS Lab, Simon Fraser University
ngonzale@sfu.ca, pasquier@sfu.ca

ABSTRACT

We investigate Musical Metacreation algorithms by applying Music Information Retrieval techniques for comparing the output of three off-line, corpus-based style imitation models. The first is *Variable Order Markov Chains*, a statistical model; second is the *Factor Oracle*, a pattern matcher; and third, *MusiCOG*, a novel graphical model based on perceptual and cognitive processes. Our focus is on discovering which musical biases are introduced by the models, that is, the characteristics of the output which are shaped directly by the formalism of the models and not by the corpus itself. We describe META-MELO, a system that implements the three models, along with a methodology for the quantitative analysis of model output, when trained on a corpus of melodies in symbolic form. Results show that the models' output are indeed different and suggest that the cognitive approach is more successful at the tasks, although none of them encompass the full creative space of the corpus. We conclude that this methodology is promising for aiding in the informed application and development of generative models for music composition problems.

1. INTRODUCTION

Computational Musicology has generally focused on studying human composed music, however, algorithms for music generation provide a rich and relatively unexplored area for study. As algorithmic and generative models grow in number and complexity, the task of selecting and applying them to specific musical problems still remains an open question; for example, in the development of Computer Aided Composition (CAC) environments.

Stylistic Imitation, a particular Musical Metacreative approach [20], can be described as creativity arising from within a pre-established conceptual space. At times, this is referred to as Exploratory Creativity [5] and in practical musical terms is concerned with generating new and original compositions that roughly cover the same space as the corpus, thus fitting a given musical style [2]. The conceptual space of a style can be defined by observing the musical features which remain invariant across the corpus if, indeed, there are any such features.

The techniques applied for this task can be broadly categorized into two methodological groups: corpus based and non-corpus based methods. In the former, musical knowledge of the style is obtained through empirical induction from existing music compositions (generally in symbolic MIDI format), using machine learning techniques. Whereas in the latter, this knowledge is provided by researchers in the form of theoretical and/or rule-based representations.

We are concerned here with applying Music Information Retrieval (MIR) tools in a controlled setting for the purpose of understanding more completely how these methods behave in real world applications. For this study we have chosen three corpus based models: the statistical *Variable Order Markov Model* (VOMM) [19], the *Factor Oracle* (FO) pattern matcher [8], and MusiCog [15], a novel, cognitively inspired approach used for the suggestion and contextual continuation (reflexive interaction) of musical ideas in the notation-based CAC system Manuscore [14].

The main question that we address is: given three corpus-based style-imitative models, which characteristics of the output are shaped by the underlying models themselves and not by the corpus? That is, we aim to discover the *musical biases* which arise from the formalism of the models. To answer this we investigate how each model's output is different in a *statistically significant* way.

A second question that arises is: what is the appropriate methodology for this research problem? We propose a framework and methodology for generating and evaluating melodies in a controlled setting where all models share the same fundamental conditions (Section 3.1). We use inter-model analysis to compare features from the melodic output of each model to the corpus and to the output of all other models, and intra-model analysis to reveal information about the relationships between the melodies generated by a single model.

Our contributions are: (1) *META-MELO* a MAX/MSP implementation of the three models, used for melodic generation from a corpus (Section 3), (2) a methodology which applies Machine Learning and MIR techniques for model output comparison (Section 5), (3) the results of a study where we apply this methodology (Section 6) and (4) a corpus of classical, popular, and jazz melodies.

Finally, we distinguish the tasks of composition from

interpretation and are concerned here only with the former.

2. EXISTING WORK

We address the problem of music evaluation by using computational techniques to investigate model output in comparison to human-composed corpora, but also in terms of model self-consistency. This analysis is useful for determining which model output is truer to a corpus, and also for discovering more precisely how the models differ. Our hypothesis is that they differ to a degree that is statistically significant, and that this difference has an effect that is perceptible and can be described as a musical bias.

In previous work on computational evaluation, Manaris et. al. [13] use artificial 'music critics' based on power-law metrics trained on a corpus as a fitness function for an evolutionary generative model. This example, as well as others, generally do not consider the comparative analysis of different music generation formalisms. On the other hand, Begleiter et. al. [4] compare the output of different VOMM algorithms in the music domain with the goal of measuring performance in terms of prediction; i.e., how precisely a model is able to reproduce the corpus. In this case the distinction is that the task of prediction is not the same as ours of comparing models that generate *novel pieces* in the same style.

The evaluation provided by Pearce and Wiggins [21] is very relevant to our work; they empirically compare the output of three variants of Markov models trained on chorale melodies using musical judges. The stepwise regression also described provides directions for improving the models by indicating the quantifiable musical features which are most predictive in their failure.

Our approach empirically compares the output of three methodologically distinct corpus-based music generation models (statistical, pattern-matching and cognitive), without the intervention of human listeners. One advantage of this is that, by avoiding listening studies, methodologies may be developed that models can incorporate for introspection in real-time generation. We also provide here a simple and alternative technique for aiding the development of the models using decision trees.

3. META-MELO

The generative system implemented in MAX/MSP, which is independent from the comparative methodology, is available for download together with the training corpus and a more detailed model description than is provided here [22]. We follow with a presentation of this system, other components used are shown in Figure 1.

3.1 Music Representation and MIDI Parsing

The system uses a simple but flexible representation for learning melodic (monophonic) music inspired by the multiviewpoint approach proposed by Conklin and Witten [7]. The symbols used for the Markov model and Factor Oracle systems are derived from the concatenation of pitch interval and onset interval information. Several attributes can be used to train the models, which brings immediate challenges for the evaluation and comparison methodology. The approach we have chosen is to restrict the study and the description of the system, for the purpose of comparing the models in a controlled setting.

If the algorithms of the models are implemented in a simple form, we expect to achieve a more transparent comparison, but with likely less interesting musical output, therefore reducing the value of the analysis. On the other hand, if more sophisticated implementations with musical heuristics are used for improving musical output, we will obtain results which are of poor generalization power with regards to the underlying models. It is worth noting that none of the models contains an explicit model of tonality, this is one of the reasons that we focus here on an analysis of the folk corpus: the relative harmonic and modulatory simplicity mitigates this issue.

For the Markov and Factor Oracle implementations, the set of attributes is indexed and a dictionary is built for the set of symbols corresponding to that attribute. This way, when an event is observed which has not appeared before, a new entry is created in the dictionary. An important function in this initial stage is the *quantization* of temporal values to a reasonable minimum resolution of sixteenth notes. This allows the parsing function to: (1) group and classify similar events as the same element in the dictionary and (2) avoid creating entries with negligible differences since notation uses quantized representations.

We parse the melody by individual note events rather than grouping by beats or bars for the purpose of obtaining an event-level granularity. Therefore there is no preservation of beat or measure information. For example, if there are two eighth-notes in a beat, we create an event for each note (note level) rather than one event with two notes (beat level). This will make evident certain biases that may otherwise be masked by parsing musical segments.

Since MusiCOG is a cognitive model [15], it handles the parsing of MIDI input using principles of music perception and cognition which are not included in the other two models, therefore not requiring some of the preliminary parsing described above.

3.2 Corpus

The corpora consist of monophonic MIDI files from Classical, Jazz, Popular, and Folk songs. These classes were selected for the purpose of investigating model behaviour in contrasting musical settings. We use a Finnish folk song collection that is available for research purposes [10], and manually created the other corpora by extracting melodies from MIDI files freely available on the web. For matters of space we present here an analysis on a subset of the folk corpus alone, the complete collection of corpora is available for download [22]. A subset of 100 pieces of 16 bars in length was selected from this corpus, totalling ~ 6400 notes. It consists of 94 combined pitch-time interval types, therefore a total of ~ 70 samples of transitions, assuming a uniform distribution.

4. OVERVIEW OF THE MODELS

4.1 Markov Models

Markov Chains are a widely used statistical approach. Two well known real-time systems implementing these techniques are Zicarelli's "M" [24] and Pachet's "Continuator" [19]. The theoretical basis lies in the field of *stochastics* which studies the description of random sequences dependent on a time parameter t. In their most basic form Markov Chains describe processes where the probability of a future event X_{t+1} depends on the current state X_t and not on previous events. In this way a sequence of musical notes can be analyzed to obtain a set of probabilities which describe the transitions between states: in this case, transitions between musical events.

As described by Conklin [6], perhaps the most common from of generation from Markov models is the so-called "random walk," in which an event from the distribution is sampled at each time step, given the current state of the model. After each selection the state is updated to reflect the selection. The *memory* or *order* of the model is the number of previous states that is considered, and thus defines the order of the Markov Chain. We implement a Variable Order Markov Model (VOMM) with a variability of 1-4 events.

4.2 Factor Oracle

Since music can be represented in a symbolic and sequential manner, pattern-matching techniques can be useful for the learning and generation of pattern redundancy and variations, respectively. The Factor Oracle [1] is one example of a text and/or biological sequence search algorithm that has been applied to music. It is an acyclic automaton with a linear growth in number of transitions with regards to the input pattern, which has been utilized in string searches [3]. There exists a construction mechanism [1] allowing the alphabet to be grown sequentially and the complete structure incremented online, thus allowing for the search of string factors in real-time.

It is important to note that neither the Markov Model (MM) nor the Factor Oracle (FO) will ever generate a transition that is not in the corpus. Also, it is conceivable that knowledge of the formal properties of each model could be used to evaluate model performance. However, as the corpus grows in size, knowledge of the formal properties of the models alone is not of much aid in predicting their behaviour: hence the need for an empirical evaluation.

4.3 MusiCOG

MusiCOG, created by Maxwell [15], models perceptual and cognitive processes, with a special focus on the formation of structural representations of music in memory. The architecture is designed for the learning and generation of musical material at various levels (pitch, interval and contour), with an emphasis on the interaction between short- and long- term memory systems during listening and composition. As a cognitive model, with a complex hierarchical memory structure, there are many possible ways to generate output from MusiCOG. For this study, in order to reflect a similar systematic approach to the FO and MM, and to avoid music theoretical or compositional heuristics, we selected a relatively simple stochastic approach which attempts to balance the application of both top-down (i.e., structural) and bottom-up (i.e., event transition) information.

MusiCOG (MC) is a feedback system, capable of interpreting its own output and modifying its behaviour accordingly. As an online model, MC will normally learn from its own output, but an option to disable this behaviour has been added and applied to half of the generation examples used for all tests in the current study. This was done in order to bring MC closer in functionality to the MM and FO, without entirely negating important aspects of its design.

5. METHODOLOGY

The analysis first requires extracting features from the input and output melodies, selecting significant features, and calculating similarity measures. Then, k-means and t-tests are used for clustering, calculating confusion matrices, and determining significant differences. Finally, we use Classic Multi-Dimensional Scaling (CMDS) for further investigating and interpreting the differences found. Figure 1 depicts a diagram outlining the methodology proposed.

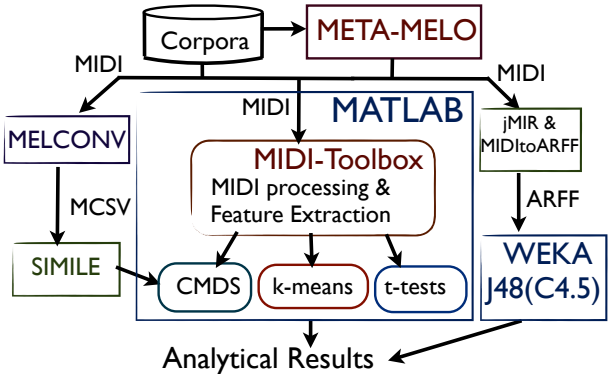


Figure 1. The components used, META-MELO is the generative MAX/MSP system which is independent from the methodology.

MATLAB is used for most data processing scripts, feature extraction, CMDS and t-test calculations. *SIMILE* [11] and *MELCONV* are Windows command line programs developed by Frieler for the conversion and comparison of MIDI monophonic files. The *Matlab MIDI Toolbox* [9] is used for a variety of MIDI processing and feature extraction functions. *WEKA* [12] is used for selecting the most significant features (C4.5 Java clone: J48) and for further data analysis and exploration. *MIDItoARFF* [23] and *jSymbolic* [16], a module of the jMIR toolbox from the same author, are also used for extracting features from MIDI files.

In Section 6.1 we describe the use of decision trees for selecting the features that best describe the differences in the models. These features are then used in Section 6.2

for visualizing the corpus and output of all models. In Section 6.3 and Section 6.4 we describe similarity analysis and CMDS respectively, used for evaluating the closeness of the groups of melodies. Finally, we performed pairwise, one-tailed t-tests for determining statistical significance, described in Section 6.5.

We trained each model with 100, 16 bar pieces from the folk corpus and generated 32 pieces of 16 bars long from each model [22].

6. RESULTS

6.1 Decision Trees

We used WEKA [12] for generating a C4.5 decision tree and reducing our feature space. This method was useful for arriving at an initial result in the search for musical biases by detecting which features, amongst the many extracted, distinguish the different models from one another and from the corpus. Figure 2 shows a tree learned on the output from the models trained on the Folk corpus collection (CC). The three features arrived at by using the C4.5 decision tree are: (1) *Compltrans*, a melodic originality measure which scores melodies based on a 2nd order pitch-class distribution (transitions between pitch classes) obtained from a set of $\sim$ 15 thousand classical themes. The value is scaled between 0 and 10 where a higher number indicates greater originality. (2) *Complebm*, an expectancy-based model of melodic complexity which measures the complexity of melodies based on pitch and rhythm components calibrated with the Essen collection. The mean value is 5 and the standard deviation is 1, the higher the value the greater the complexity of the melody. (3) *Notedensity*, the number of notes per beat. Details of these features can be found in the *MIDI Toolbox* documentation [9].

The first number in the leaf is the count of instances that reach that leaf, the number after the dash, if present, indicates the count of those instances which are misclassified along with the correct class. Three aspects of the tree stand out:

1. Most of MM, 25 melodies, are classified as FO and are therefore not easy to distinguish.
2. The root (*Compltrans*) successfully separates 86% of FO and MM instances from 89% of CC and 100% of MC, an indication of greater similarity between CC and MC.
3. The *Notedensity* feature seems to greatly aid in classifying and distinguishing MC from CC where other features are less successful. This type of analysis provides valuable diagnostic insight on the MC model since we can deduce that an increase in note density on the output would potentially improve the imitation capabilities of the model.

6.2 Originality and Complexity

In Figure 3 the corpus and the output instances for all models are plotted using the *Compltrans* and *Complebm* features described in Section 6.1. The plot shows a clear

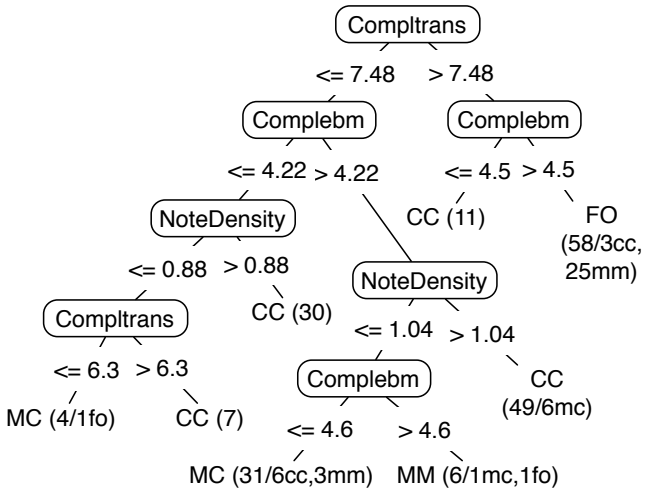


Figure 2. C4.5 decision tree (J48).

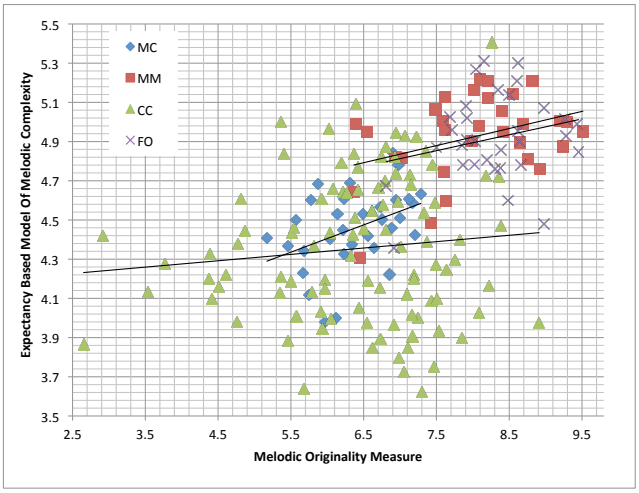


Figure 3. Expectancy Based Complexity and Originality. Folk corpus (CC) and model output.

overlap between the corpus and MC, whereas MM and FO cluster together with higher values on both dimensions.

6.3 Similarity Analysis

It has been noted by Müllensiefen and Frieler [17, 18] that hybrid measures have a greater predictive power than single-attribute measures for estimating melodic similarity. They provide an optimized metric 'Opti3,' which has been shown to be comparable to human similarity judgements [18]. Opti3 is based on a weighted linear combination of interval-based pitch, rhythmic, and harmonic similarity estimates, normalized between 0-1, where a value of 1 indicates that melodies are identical.

The corpus and the three sets of model output were analyzed to establish the similarity between them (inter-corpus analysis), as well as the diversity within the sets (intra-corpus analysis). We use the Opti3 measure and calculate the mean of the distances between each melody from

Intra and Inter-Model Melodic Similarity

Model	MM	MC	FO	CC
Markov (MM)	.206			
MusiCOG (MC)	.183	.208		
Factor Oracle (FO)	.166	.165	.198	
Folk Corpus (CC)	.178	.174	.154	.201

Table 1. Mean melodic similarity of model output and corpora using the "Opti3" similarity measure (1.0 = identity). Intra-model similarity is represented in the diagonal, lower value indicates higher diversity.

the set in the row against all melodies in the column set (cartesian product). Looking at Table 1, we can interpret the diagonal as intra-model dissimilarity, the diversity of each set. Since a low value indicates higher diversity, MC is marginally the least and FO the most diverse of all sets, including the corpus. Furthermore, with this analysis, MM produces the output which is most similar to the corpus. This is the first discrepancy that we observe in the results.

6.4 Classic Multi-dimensional Scaling

CMDS is a multivariate analysis process similar to Principal Component Analysis (PCA), used for visualizing the clustering of N-dimensional data. We calculated dissimilarity matrices from the similarity measures obtained with "Opti3". In Figure 4 we can see that the horizontal axis separates quite generally the corpus from the model output. Although the dimensions are not easy to interpret, it is evident that the models do not explore the full 'creative' range of the corpus. Correlating with the similarity analysis, the diversity of FO output is apparent as it occupies a broader range in the space. It is worth noting that a similar topology was observed when scaling a set of 100 melodies from each model [22].

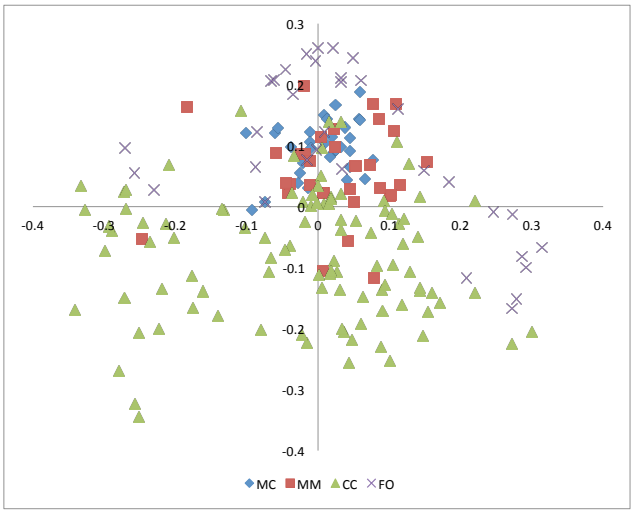


Figure 4. Multi-dimensional scaling for all models and corpus, using the optimized distance metric 'Opti3'. The corpus collection is marked as 'CC'.

t-test Results

	MM	MC	FO
MusiCOG	IVdist	-	-
Factor Oracle	No	PCdist	-
Folk Corpus	PCdist	No	PCdist

Table 2. T-test results for the folk corpus and model output, 'No' is indicated where no significance was found, otherwise the dimension where significance exists. In these cases the resulting p values are all < 0.001.

6.5 Significance Tests

Table 2 shows the pairwise tailed t-tests that were performed on the 6 pairs of groupings of corpus and models for determining difference across four dimensions: Pitch-class distributions (PCdist), interval distribution (IVdist), contour and rhythm (*meldistance* function [9]). First, as in the similarity analysis, the mean distance between all melodies in one set is measured. Second, the distance from each melody in this set is measured against all the melodies in the set with which it is being compared. Finally, the t-test is run on these two sets of measurements (further details are available [22]). Where we found significance, it was at most in one dimension, either PCdist or IVdist. In those cases the p value is < 0.001, Bonferroni corrected. Results show that (1) MC is significantly different from both other models but not from the corpus, and (2) both MM and the FO are different from the corpus and undifferentiated between themselves. Although it is unclear how much these results are affected, the assumption of independence is violated in this study and for this reason further exploration for a suitable analysis is required.

7. CONCLUSION AND FUTURE WORK

Returning to our broad definition of stylistic imitation, we expect successful models to roughly cover the same space as the corpus. The CMDS diagram (Figure 4) shows graphically that this is not occurring in our study, which clearly shows that the problem of stylistic imitation warrants further research.

We have also shown that the task of investigating for significance in the differences of the output is valuable for validating closeness to the corpus. The decision trees inform us, in musical features, where the important differences can be found.

Unlike VOMM and Factor Oracle which have no musical domain knowledge, MusiCOG is informed by music perception and cognitive science. This 'knowledge bias' in MusiCOG may result in output which is more true to the corpus. As such, this encourages the continued investigation into developing musical cognitive models.

We leave the following for future work: the application of the methodology to polyphonic music and models that include harmonic knowledge, an in-depth analysis of the output of the models when trained on different corpora, and an evaluation of the behaviour of the models when combining stylistically diverse corpora (combinatorial creativity),

and the exploration for a more suitable statistical analysis.

8. ACKNOWLEDGEMENTS

We want to thank Klaus Frieler for providing us with his insights and software for melodic analysis as also the reviewers for the valuable comments. This research was made possible by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

9. REFERENCES

[1] C. Allauzen, M. Crochemore, and M. Raffinot. Factor Oracle: A New Structure for Pattern Matching. In *In Proceedings of SOFSEM'99: Theory and Practice of Informatics*, pages 291–306, Berlin, 2009.

[2] S. Argamon, S. Dubnov, and K. Burns. *The Structure of Style: Algorithmic Approaches to Understanding Manner and Meaning*. Springer-Verlag, Berlin, 2010.

[3] G. Assayag and S. Dubnov. Using Factor Oracles for Machine Improvisation. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 8(9):604–610, 2004.

[4] R. Begleiter, R. El-Yaniv, and G Yona. On Prediction Using Variable Order Markov Models. *Journal of Artificial Intelligence Research*, 22:385–421, 2004.

[5] M. A. Boden. Computer Models of Creativity. In *Handbook of Creativity*. ed. R. J. Sternberg, pages 351-372. Cambridge University Press, 1999.

[6] D. Conklin. Music Generation From Statistical Models. In *Proceedings Of The AISB 2003 Symposium On Artificial Intelligence And Creativity In The Arts And Sciences*, pages 30–35, 2003.

[7] D. Conklin and I. H. Witten. Multiple Viewpoint Systems for Music Prediction. *Journal of New Music Research*, 24:51–73, 1995.

[8] A. Cont, S. Dubnov, and G. Assayag. A Framework for Anticipatory Machine Improvisation and Style Imitation. In *Anticipatory Behavior in Adaptive Learning Systems (ABiALS)*. ABIALS, 2006.

[9] T. Eerola and P. Toiviainen. *MIDI Toolbox: MATLAB Tools for Music Research*. University of Jyväskylä, 2004. `www.jyu.fi/musica/miditoolbox/`.

[10] T. Eerola and P. Toiviainen. Suomen Kansan eSavelmat. Digital Archive of Finnish Folk Tunes, 2004. `http://esavelmat.jyu.fi/collection.html`.

[11] K. Frieler and D. Müllensiefen. The SIMILE Algorithms Documentation. Technical Report. 2006. `http://doc.gold.ac.uk/isms/mmm/SIMILE_algo_docs_0.3.pdf`. Last visited on July 2013.

[12] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA Data Mining Software: an Update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[13] B. Manaris, P. Roos, P. Machado, D. Krehbiel, L. Pellicoro, and J. Romero. A Corpus-Based Hybrid Approach to Music Analysis and Composition. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 839–845. AAAI Press, 2007.

[14] J. B. Maxwell, A. Eigenfeldt, and P. Pasquier. ManuScore: Music Notation-Based Computer Assisted Composition. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 357–364, 2012.

[15] J. B. Maxwell, A. Eigenfeldt, P. Pasquier, and N. Gonzalez Thomas. MusiCOG: A cognitive architecture for music learning and generation. *Proceedings of the 9th Sound and Music Computing conference (SMC 2012)*, pages 521–528, 2012.

[16] C. Mckay and I. Fujinaga. jSymbolic: A Feature Extractor for MIDI Files. In *International Computer Music Conference*, pages 302–305, 2006.

[17] D. Müllensiefen and K. Frieler. Cognitive Adequacy in the Measurement of Melodic Similarity: Algorithmic vs. Human Judgments. *Computing in Musicology*, 13(2003):147–176, 2004.

[18] D. Müllensiefen and K. Frieler. Melodic Similarity: Approaches and Applications. In *Proceedings of the 8th International Conference on Music Perception and Cognition (CD-R)*, 2004.

[19] F. Pachet. The Continuator: Musical Interaction With Style. *Journal of New Music Research*, 32(3):333–341, 2003.

[20] P. Pasquier, A. Eigenfeldt, and O. Bown. Proceedings of the First International Workshop on Musical Metacreation. In *Conjunction with the The Eighth Annual AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. AAAI Technical Report WS-12-16.* AAAI Press, 88 pages, 2012.

[21] M. Pearce and G. Wiggins. Evaluating Cognitive Models of Musical Composition. In *Proceedings of the 4th International Joint Workshop on Computational Creativity*, pages 73–80, 2007.

[22] META-MELO. Online Resource. `http://metacreation.net/meta-melo/home.html` Last visited on July 2013.

[23] D. Rizo, P. J. Ponce de León, C. Pérez-Sancho, A. Pertusa, and J. M. Iñesta. A Pattern Recognition Approach for Melody Track Selection in MIDI Files. In *Proc. of the 7th Int. Symp. on Music Information Retrieval ISMIR*, pages 61–66, 2006.

[24] D. Zicarelli. M and Jam Factory. In *Computer Music Journal*, volume 11, pages 13–29. JSTOR, 1987.

Oral Session 8:

Music Similarity

AN EXTENDED AUDIO-FINGERPRINT METHOD WITH CAPABILITIES FOR SIMILAR MUSIC DETECTION

Sébastien Fenet, Yves Grenier, Gaël Richard
Institut Mines-Télécom ; Télécom ParisTech ; CNRS LTCI
37 rue Dareau, 75014 Paris, France
Firstname.Lastname@telecom-paristech.fr

ABSTRACT

Content-based Audio Identification consists of retrieving the meta-data (i.e. title, artist, album) associated with an unknown audio excerpt. Audio fingerprint techniques are amongst the most efficient for this goal: following the extraction of a fingerprint from the unknown signal, the closest fingerprint in a reference database is sought in order to perform the identification. While being able to manage large scale databases, the recent developments in fingerprint methods have mostly focused on the improvement of robustness to post-processing distortions (equalization, amplitude compression, pitch-shifting,...). In this work, we describe a novel fingerprint model that is robust not only to the classical set of distortions handled by most methods but also to the variations that occur when a title is re-recorded (live vs studio version in particular). As a result our fingerprint method is able to identify any signal that is an excerpt of one of the references from the database or that is similar to one of the references. The issue that we cover thus lies at the intersection of audio fingerprint and cover song detection, meaning that the functional perimeter of our method is substantially larger than the classical audio fingerprint approaches.

1. INTRODUCTION

Audio identification consists of retrieving some meta-data associated with an unknown audio excerpt. The typical use-case is the music identification service which is nowadays available on numerous mobile phones. The user captures an audio excerpt with his mobile phone microphone and the service returns meta-data such as the title of the song, the artist, the album... Other applications include automatic copyright control, automatic segmentation and annotation of multimedia streams, jingle detection, ... (see [1] for more details).

Audio fingerprint is the most common strategy for performing audio identification when no meta-data has been embedded in the unknown audio excerpt. It consists of extracting from each audio reference of a given database a compact representation (the fingerprint) which is then stored in a database. When identifying an unknown excerpt, its fingerprint is first calculated and then compared with all fingerprints stored in the database. This general scheme (see Figure 1) has been explored by numerous authors (for example, see [2–5]). The state of the art thus comprises a wide variety of models for the "Fingerprint" block associated with their search strategies ("Matching" block). The authors of recognized works have put their efforts on two major points: first, the captured signal should be recognized even if it has undergone a series of distortions; second, the algorithm has to manage a database containing huge amounts of audio references. This robustness to distortions usually refers to the robustness to different post-processings of the signal. Indeed, the successful fingerprint technologies can achieve identification of a signal that has been cropped, re-equalized, amplitude-compressed, pitch-shifted and/or time-stretched and possibly recorded in a noisy environment with a bad microphone. Clearly, traditional fingerprint methods do not handle the connection that exists between two pieces of music that are similar. This means that if the system has learned one song and is required to perform an identification of a cover version, it will not be able to indicate the correspondence. This includes the case of an artist performing 'live' one of his titles that is stored in its studio version in the database. At a more dramatic level, the same song, with the same orchestration, that is re-recorded in the same studio conditions will not be considered as a match of the first recording. This comes from the fact that traditional fingerprint approaches use low level features to characterize the

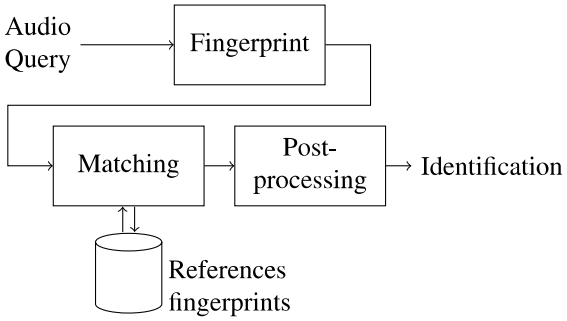


Figure 1. Architecture of a fingerprint system

THIS WORK WAS ACHIEVED AS PART OF THE QUAERO PROGRAMME, FUNDED BY OSEO, FRENCH STATE AGENCY FOR INNOVATION.

signal. These features, which bear little musical meaning, are not preserved from one version of the song to the other.

Conversely, approaches that can match two very different versions of one song have been developed [6–8] in the field of cover recognition. The counterpart of these approaches is that they usually cannot handle a large reference database. The proposed methods are indeed CPU-demanding and since they do not include any kind of fast search mechanism, using them in an audio-identification use-case would require the exhaustive comparison of the unknown signal with every reference. Given the computation time of the method and the number of references in traditional use-cases, this would not be feasible.

We propose in this work an extended fingerprint algorithm, that is able to do the matching between two identical records with different post-processings (called near-exact matches) but that can also recognize a query that is only "similar" to a song of the database, while keeping the ability to manage large scale databases. The paper is organized as follows: Section 2 is dedicated to a detailed description of our method. Following the presentation of the general workflow, the signal model is introduced and it is shown how this model can integrate in an index scheme that allows the management of large-scale databases. We finally explain a scoring procedure that can be applied to a reduced set of candidates that are output after the index phase. In Section 3, we propose some experiments to evaluate the performance of the algorithm on an audio-fingerprint use-case that includes both challenges of identifying broadcast sections that are near-exact matches of references and others that are only similar to references. The paper ends with a synthesis of the work and a critical analysis of the results.

2. DESCRIPTION OF THE SYSTEM

2.1 General Workflow

In our system, any piece of audio signal is modeled by a sequence of states computed in the fashion of Section 2.2. Similarly to the vast majority of fingerprinting approaches (*e.g.* [3] [4]) the efficiency that is necessary to handle large scale databases is achieved thanks to the use of an index-based search. Our method thus includes a learning stage, during which features extracted from the audio references are used to build the index. However, the task of identifying matches that are "near-exact" as well as "similar" is too demanding to be performed by the index search on its own. We thus propose a "two-levels" architecture, such as shown in Figure 2. When analyzing an unknown query (modeled by its sequence of states), the system performs a first search in the index. The output of this step is a reduced set of M best candidates (typically $M = 10$). Thanks to the index, the candidates are efficiently selected among the numerous references. The counterpart is that this matching stays vague. The post-processing step is a finer comparison between the candidates and the query. In our implementation the comparison is achieved thanks to dynamic programming, which has a heavy CPU cost but it is here limited to a small number of candidates. This gives a matching score for each of the candidates with the query. A threshold strategy can finally be set up in order to decide if there is a match or not. The same kind of funnel-shaped architecture is used in [4].

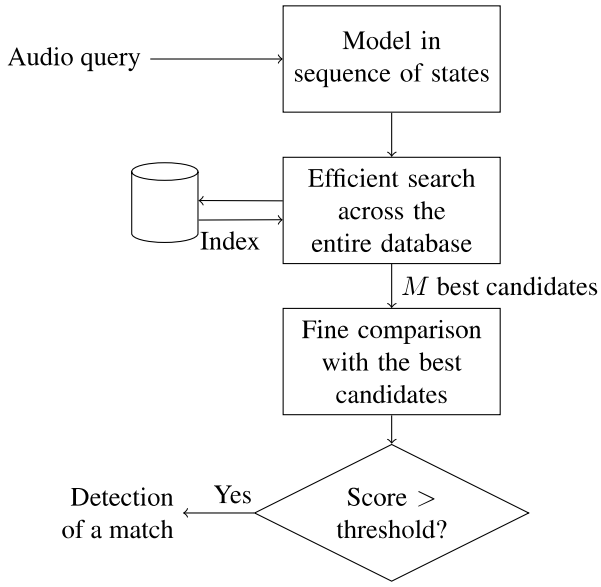


Figure 2. Workflow when identifying an audio query

2.2 Signal Model

The signal model that we suggest can be seen as an extension of the model proposed in [9] in a different context. The main idea to obtain the sequence of states is to sample the signal at instants that are "musically meaningful". These occur at dates $\{t_1, t_2, .., t_n\}$. To each of this date t_k we associate an information i_k which locally characterizes the signal. Finally, we define the *state* as the association of the date with the local piece of information:

$$s_k = (t_k, i_k)$$

In our implementation, the dates $\{t_1, ..., t_n\}$ correspond to the peaks of an onset detection function based on a sub-band decomposition (as suggested in [10]). In brief, the method starts from the magnitude of the spectrogram of the signal. The latter is temporally filtered in order to mask the rapid variations while keeping the sharp attacks. The filtered spectrogram is then log-compressed, which has the effect of remapping the amplitudes of the spectrogram on a scale that is closer to our perception. The Spectral Energy Flux is subsequently computed by estimating the temporal differential of the pre-processed spectrogram and then half-wave rectified in order to keep only positive values. Finally the information is integrated by summing the values of the Spectral Energy Flux across the frequency bands. This gives a global onset detection function that shows high values at instants with a meaningful amount of change in the spectrogram in one or several frequency bands. Onsets are then localized thanks to a peak-peaking strategy.

As for the local information i_k, it is composed of two features:

$$i_k = (c_{k,l}, c_{k,r})$$

where $c_{k,l}$ is the mean chroma vector at the left of t_k and $c_{k,r}$ is the mean chroma vector at the right of t_k. The use of chroma vectors is motivated by the fact that these features show little variations from one version of a song to the other [7].

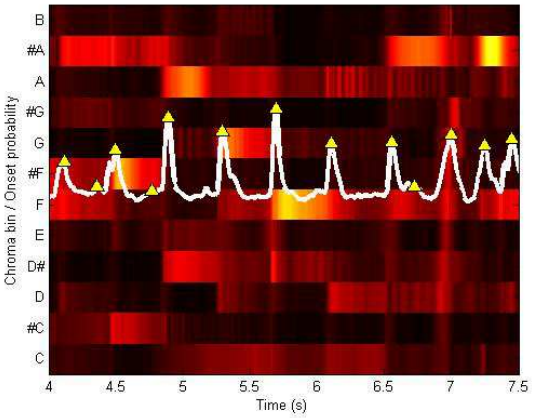


Figure 3. Illustration of the decomposition of the signal

Figure 3 shows the superposition of the graphical representations of an onset detection function (in white) and a chroma representation of the same signal. In our model, the dates $\{t_1, ..., t_n\}$ would be given by the dates of the yellow triangles on the onset function (obtained by peak-picking). For each date, the left mean chroma vector (mean chroma vector between the previous triangle and the current) and the right mean chroma vector (mean chroma vector between the current triangle and the next) would be computed, associated to the date and stored. The advantage of this model is that it is rather compact (which is an interesting feature for the search in large-scale databases) but still contains both rhythmical and harmonic information.

2.3 Indexing Scheme

An index is a function that takes a key in input and outputs a set of values.

$$h : k \longmapsto \{v_i\}$$

The interest of the operation is that it can be done in time $O(1)$. In our case, the values will be pointers to the references of the database and the keys will be audio features that characterize these references. The crucial point is to set up a distortion-invariant key, while still sufficiently discriminating. The invariance to the distortions should handle the classical post-processing distortions (equalization, pitch-shifting, ...) but also the recording of the same song in different conditions (matching of similar items).

For a given audio signal, we propose the following key-generation mechanism. For each state s_k one key is generated. The key is a binary version of the mean chroma vector at the right of t_k. Practically, for any bin b of the chroma vector, the binary chroma vector $\widetilde{c}_{k,r}$ is given by:

$$\widetilde{c}_{k,r}(b) = \begin{cases} 1 & \text{if } c_{k,r}(b) > \overline{c_{k,r}} \\ 0 & \text{otherwise} \end{cases}$$

with $\overline{c_{k,r}}$ the mean value of vector $c_{k,r}$:

$$\overline{c_{k,r}} = \frac{1}{12} \sum_{b=1}^{12} c_{k,r}(b)$$

We can note that at this stage of the process, the mean chroma vector at the left is unused. It will however be involved in the fine comparison step.

The learning phase is similar to [3]. The keys of all the references are extracted. They are then stored in the index. When storing one key, we define its associated value as the identifier of the reference containing this key together with the time of occurrence t_k of the key in the reference. Let us note that if one key k occurs several times, $h(k)$ will consist of all the different values stored with this key.

In the analysis phase, the keys of the query are extracted. For one key k occurring at time $t_q(k)$ in the query we can efficiently retrieve, thanks to the index function, all its occurrences in the references. If k occurs at times $t_r^1(k)$, ..., $t_r^i(k)$ in the reference r, we store the couples $(t_r^1(k), t_q(k))$, ..., $(t_r^i(k), t_q(k))$ in a scatter plot (one scatter plot per reference).

Let us now consider that the query is an excerpt of the reference r starting at time t_0. Let us also consider that the query is time-stretched by a factor κ (either because of some specific post-processing or because the query is another record of the same song with a slightly different tempo). Then, the key k extracted from the query at time $t_q(k)$ should be retrieved in the reference r at time $\kappa(t_q(k) - t_0)$. This means that, in the scatter plot of reference r, all these corresponding keys produce dots that are located on the straight line:

$$Y = \frac{1}{\kappa} X + t_0$$

Though, the scatter plot also contains a meaningful number of dots that are outside this line. These correspond to keys that are found in the query and that occur several times in the reference. We must indeed keep in mind that the keys we defined are not very discriminative. Figure 4 shows an example of such a scatter plot.

The last step consists of finding, for each scatter plot, the 'best' straight line. Intuitively enough, the 'best' line is the one comprising the highest number of dots. This is done thanks to the Hough transform [11]. In brief, it is an efficient counting technique that relies on the set up of an accumulator, which references all possible lines. Each dot of the scatter plot is iteratively tested in the following way: all lines that go through that dot have their values increased in the accumulator. At the end of the process, the line with the highest value in the accumulator is the best one.

In the end, we have, for each reference, the parameters of the best line (κ and t_0) and the number of dots (*i.e.* the number of keys) that match this line. The M references with the highest number of matching keys are considered

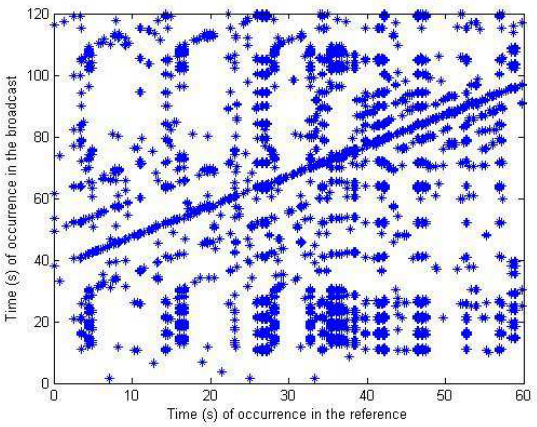


Figure 4. Scatter plot of reference r

as the M best matches to the query. They are output as candidates.

2.4 Final Scoring of the Candidates

At this stage of the process, the system has to compare the reduced set of M best candidates with the query. We can consequently afford a CPU-consuming comparison. The state of the art in similarity offers a wide variety of algorithms. In this work, we suggest a dynamic programming approach that relies on the modeling of the audio signal that we introduced in section 2.2. This can be seen as an extension of the work in [9].

First, the query is cropped and rescaled according to the parameters κ and t_0 that have been output in the search-by-index phase. The corrected query is represented by the sequence of states $\{s_1...s_m\}$. As for the candidate reference, we use the notation $\{s'_1...s'_n\}$.

When dynamically aligning two sequences of states, one has to define three types of penalties.

- $f^d(s_i)$: penalty assigned to the deletion of state s_i
- $f^i(s_i)$: penalty assigned to the insertion of state s_i
- $f^s(s_i, s'_j)$: penalty assigned to the substitution of state s_i by state s'_j.

In our implementation f^d and f^i are both taken constant and equal to 0.3. This value has been experimentally determined. Knowing that $s_i = (t_i, (c_{i,l}, c_{i,r}))$ and $s'_j = (t'_j, (c'_{j,l}, c'_{j,r}))$, we define f^s by:

$$f^s(s_i, s'_j) = \cos(c_{i,l}, c'_{j,l}).\cos(c_{i,r}, c'_{j,r}).\mathrm{e}^{\frac{|t_i - t'_j|}{c}}$$

The first cosine term penalizes the resemblance of the mean chroma vector at the left of s_i with the one at the left of s'_j. The second cosine term does the same for the mean chroma vector at the right. The exponential term penalizes the timing error between the occurrence of state s_i and the occurrence of state s'_j.

Dynamic programming consists of iteratively filling a scoring matrix D. For any $(i, j) \in \{1..m\} \times \{1...n\}$, $D(i, j)$ contains the score of the alignment of the subsequence $\{s_1...s_i\}$ with the subsequence $\{s'_1...s'_j\}$. D is computed in the following manner:

$$D(i,j) = \max \left\{ \begin{array}{l} D(i, j-1).f^d(s'_j) \\ D(i-1, j-1).f^s(s_i, s'_j) \\ D(i-1, j).f^i(s_i) \end{array} \right\}$$

The score of the alignment of $\{s_1...s_m\}$ with $\{s'_1...s'_n\}$ is finally given by $D(n, m)$ (Figure 5 shows an example of matrix D). The candidate reference is considered as an actual match to the query if the score is greater than a predetermined threshold.

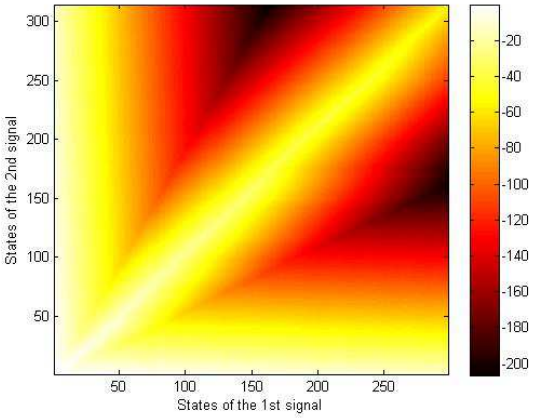


Figure 5. Scoring matrix of a dynamic alignment

3. EXPERIMENTS

3.1 Experimental Framework

Our experiments follow the evaluation framework described in [12]. This framework was designed for the evaluation of audio-fingerprint systems on the basis of real broadcast streams. This constitutes a quite challenging evaluation, even if one restricts the task to the detection of near-exact matches, because the broadcast stations apply a wide variety of different sound-processing to the music, thus resulting in a heavy level of post-processing distortion to be handled by the algorithm. The evaluated system learns a database of reference music titles before the actual evaluation. Each music title broadcast in the stream that corresponds to one of the references has been manually annotated in a global annotation file. The annotation contains a unique identifier of the broadcast title, its start and end times in the broadcast. When the algorithm is run on the stream, it has to detect all the occurrences of the broadcast references. One detection output by the algorithm contains the identifier of the detected reference and its detection time in the stream. If one output detection contains the identifier r_i and has a detection time that lies between the annotated start time and end time of one occurrence of the same title r_i, we make this broadcast a *detected title* (DT). Let us note that multiple detections of the same occurrence

of one given title are counted only once. Conversely, if the algorithm detects a reference during an empty slot (denoted by n) or during a slot that contains another music title, we count one false alarm (FA). There is no kind of integration performed on the false alarm counting: each output corresponding to a false alarm is added up. Figure 6 illustrates the task of the algorithm as well as the scoring procedure. The detections output by the algorithm are figured by the arrows. The way we integrate these detections when scoring is illustrated by the overbraces.

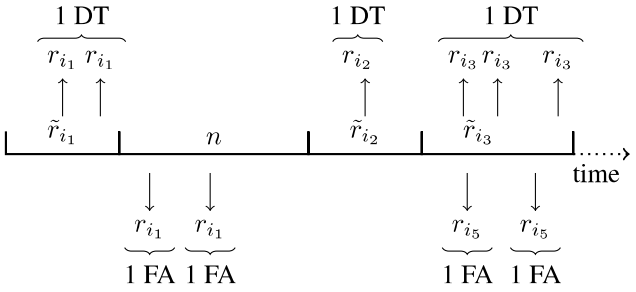


Figure 6. Illustration of the evaluation framework

It is noticeable that the main objective of this framework is to provide an evaluation methodology that fits a realistic use-case with real-world data. The counterpart of this strategy is that we have a limited control over the corpus, which prevents us from displaying metrics such as the level of distortion in the broadcast titles or the detection ability of the algorithm at a scale that is shorter than the duration of a song.

3.2 Corpus

It is quite a major issue to bring together a reference database of real music titles with the corresponding annotations (with a reasonably low number of wrong annotations). In our context, we could assemble an annotated database of 2400 pop-rock music titles. Let us note that such a database still constitutes a realistic reference set for many applications. It is for instance noticeable that the assembled database covers all the music broadcasts of two different radio channels during two weeks. Besides, such a quantity of references is large enough to get a fair idea of the system's capabilities.

As far as the analyzed stream is concerned, we worked with 24h of the French radio RTL. These contain one program that essentially features live performances of contemporary titles. The latter are not explicitly present in the database but the corresponding studio versions are. In total, the stream contains 107 annotated music titles from the database, 99 of which are near-exact matches whereas the 8 remaining are live versions of the corresponding references.

Given the framing that we apply to the stream, whose hop is set to 15 seconds, the algorithm is queried $\frac{24.3600}{15} = 5760$ times. Since the queries are extracted from a real broadcast, they consist of distorted versions of the references, live versions of the references or empty slots (speech, ads or musics that are not in the database). In terms of false alarms, this means that the algorithm has the possibility to output around $5760 \times 2400 = 13,824,000$ of them. These figures confirm the relevance of the experiment.

3.3 Results

Figure 7 shows, in red, the results of the algorithm on a classical ROC diagram. As most detection systems, our algorithm's output relies on a set of candidate detections with given scores. The candidate references are originally selected in the search-by-index step and they are finally scored thanks to a dynamic alignment. In order to complete the process, one can simply set a threshold on the score: only the candidates with a score above the threshold are output by the algorithm. In such a configuration, one can evaluate the algorithm's output with different threshold values. Each point of the ROC curve thus corresponds to the results obtained by the algorithm with one specific threshold value. The X-coordinate corresponds to the number of false alarms and the Y-coordinate to the proportion of *detected titles*. Such a curve allows to observe the overall response of the algorithm and to compare different methods independently of the final post-processing. For comparison, the ROC curve of a 'traditional' audio-fingerprint method [13] is plotted in black. This method has proved to be at the level of the state of the art, notably when working with very large databases ($> 30,000$ references).

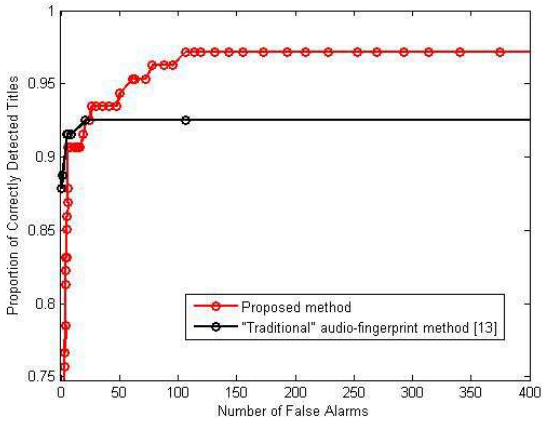


Figure 7. Results of the algorithm (ROC curve)

The results show that the method from [13] is indeed very efficient at detecting near-exact matches: the ROC curve is almost ideal if we do not consider the similar items. The approach can indeed reach the highest number of correct detections (*i.e.* the 92.5% of the corpus that are near-exact matches) while virtually outputting no false alarm. As far as our novel method is concerned, the outputs of the algorithm show that it is also able to detect near-exact matches with a low number of false alarms, albeit a little higher than the traditional system. But most importantly, it is able to overstep the borders of exact matching. We

can indeed see that some of the points of the ROC curve have a higher Y-coordinate than the proportion of near-exact matches in the corpus. This means that the algorithm is able to successfully detect some of the broadcasts that are live versions of music titles that are stored in their studio version in the reference database! Interestingly, the annotated broadcast titles that are never reached by the ROC curve correspond to live versions that have been transposed compared to the studio version in the reference database. Of course the more we go toward similarity, the higher number of false alarms we generate. This sounds logical since by loosening the matching requirements in order to detect the live versions of some references, we also favor the detections of distinct items that are musically similar to some of the references.

In terms of run time, it takes our current Matlab implementation of the method 6 seconds to perform the index search (across the entire database) for a 120s-long query. We are thus confident that the algorithm can scale up to much larger reference databases while keeping reasonable computation times. For the sake of precision, the code was run on an Intel Core 2 Duo @ 3,16 GHz with 6MB of Cache and 8GB of RAM. We do not give details about the running time of the fine comparison step. The latter is indeed meant to process a fixed number (M) of best candidates, whatever the size of the database. Its running time thus has no impact on the scalibility of the algorithm.

4. CONCLUSION

In this work we have presented a novel fingerprint model associated with a tailored innovative search strategy. As a member of the "audio-fingerprint algorithms" family, the resulting algorithm has the ability of managing large reference databases. Besides it has the very interesting capability of indifferently managing queries that are near-exact matches of some reference of the database or queries that are only similar to one reference. This result is achieved thanks to a careful modeling that preserves the musical significance of the signal at every calculation step (computation of the model in state, extraction of binary keys, aggregation of the index outputs).

The algorithm has been tested on a well-tried evaluation framework for the detection of near-exact matches. We have selected a specific corpus that includes both near-exact and similar matches. The conclusion of this experiment is that the algorithm could detect with a performance close to the state of the art the items that were near-exact matches and it could also detect some of the similar items. These results are thus very encouraging.

Our perspectives include a more extensive testing of the algorithm. The conducted experiment indeed involves a middle-size database. We need to go towards bigger corpora to check the scalability of the approach. We also target the further extension of the robustness of the model. The described experiment has indeed emphasized the lack of robustness of the current model to transposition. Besides the search for straight lines in the scatter plots makes it clear that the model is not robust to a change of structure. Trying to handle these while keeping the indexing capability of the system are challenges of high interest.

5. REFERENCES

[1] P. Cano, E. Batlle, E. Gomez, L. de C.T. Gomes, and M. Bonnet, "Audio Fingerprinting: Concepts and Applications," in *International Conference on Fuzzy Systems and Knowledge Discovery*, (Singapore), Nov. 2002.

[2] M. Ramona and G. Peeters, "Audio identification based on spectral modeling of bark-bands energy and synchronization through onset detection.," in *ICASSP*, (Prague, Czech Republic), May 2011.

[3] A. Wang, "An Industrial-strength Audio Search Algorithm," in *ISMIR*, (Baltimore, Maryland, USA), Oct. 2003.

[4] J. Haitsma, T. Kalker, and J. Oostveen, "Robust audio hashing for content identification," in *CBMI*, (Brescia, Italy), Sept. 2001.

[5] C. Cotton and D. Ellis, "Audio Fingerprinting to Identify Multiple Videos of an Event," in *ICASSP*, (Dallas, Texas, USA), Mar. 2010.

[6] D. P. W. Ellis and G. E. Poliner, "Identifying 'Cover Songs' with Chroma Features and Dynamic Programming Beat Tracking," in *ICASSP*, (Honolulu, Hawaii, USA), Apr. 2007.

[7] 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*, Aug. 2008.

[8] M. Müller, F. Kurth, and M. Clausen, "Audio matching via chroma-based statistical features," in *ISMIR*, (London, UK), Sept. 2005.

[9] O. Gillet and G. Richard, "Drum loops retrieval from spoken queries," *Journal of Intelligent Information Systems*, vol. 24, pp. 159–177, May 2005.

[10] M. Alonso, G. Richard, and B. David, "Extracting note onsets from musical recordings," in *ICME*, (Amsterdam, NL), July 2005.

[11] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, pp. 11–15, Jan. 1972.

[12] M. Ramona, S. Fenet, R. Blouet, H. Bredin, T. Fillon, and G. Peeters, "Audio Fingerprinting: a Public Evaluation Framework Based on a Broadcast Scenario," *Applied Artificial Intelligence: An International Journal*, Feb. 2012.

[13] S. Fenet, G. Richard, and Y. Grenier, "A Scalable Audio Fingerprint Method with Robustness to Pitch-Shifting," in *ISMIR*, (Miami, Florida, USA), Oct. 2011.

AUTOMASHUPPER: AN AUTOMATIC MULTI-SONG MASHUP SYSTEM

Matthew E. P. Davies, Philippe Hamel, Kazuyoshi Yoshii and Masataka Goto
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{matthew.davies, hamel.phil, k.yoshii, m.goto} [at] aist.go.jp

ABSTRACT

This paper describes *AutoMashUpper*, an interactive system for creating music mashups by automatically selecting and mixing multiple songs together. Given a user-specified input song, the system first identifies the phrase-level structure and then estimates the "mashability" between each phrase section of the input and songs in the user's music collection. Mashability is calculated based on the harmonic similarity between beat synchronous chromagrams over a user-definable range of allowable key shifts and tempi. Once a match in the collection for a given section of the input song has been found, a pitch-shifting and time-stretching algorithm is used to harmonically and temporally align the sections, after which the loudness of the transformed section is modified to ensure a balanced mix. AutoMashUpper has a user interface to allow visualisation and manipulation of mashups. When creating a mashup, users can specify a list of songs to choose from, modify the mashability parameters and change the granularity of the phrase segmentation. Once created, users can also switch, add, or remove sections from the mashup to suit their taste. In this way, AutoMashUpper can assist users to actively create new music content by enabling and encouraging them to explore the mashup space.

1. INTRODUCTION

Mashups form a key part of the remix culture in music production and listening. Created by mixing together multiple songs, or elements within songs, music mashups hold strong potential for entertaining and surprising listeners by bringing together disparate musical elements in unexpected ways. Until recently, the process for creating mashups relied on two elements: first, the requisite musical imagination (and access to a large and varied music catalogue) to determine which songs to mix together, and second, the technical ability to use a Digital Audio Workstation to produce high quality results.

Due to the high popularity of mashups, some commercial systems and online tools have become available to assist users (both professional DJs and amateurs) in mixing songs and creating mashups. *DJ Mix Generator* [1] is an online database of 30,000 songs, where users can search by tempo, key and genre to find compatible songs to mix together. The *Harmonic Mixing Tool* [2] offers similar functionality, but instead of presenting results through an online search engine, it analyses a user's collection to identify song compatibility and can create a "harmonic fade" when mixing between songs. *Mashup* [3] also provides a harmonic compatibility measure between songs, which is determined using a key signature detection algorithm and relationships in the circle of fifths. To allow the manual creation of mashups, *Mashup* has an advanced audio editing interface.

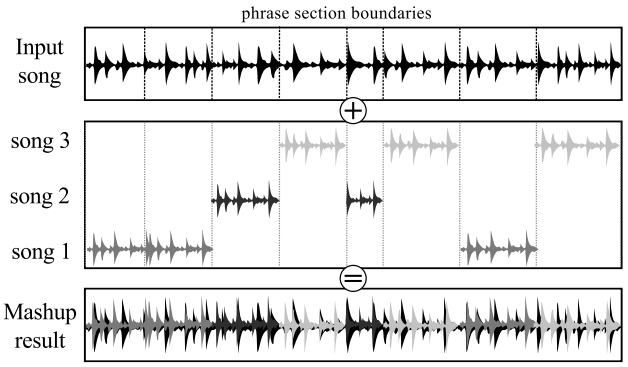


Figure 1. The concept of multi-song mashups.

Given the existence of these commercial mashup tools and their use of MIR techniques such as key detection, beat tracking and tempo estimation, it is quite surprising that so few research papers exist on this topic. Of those which do, their scope appears limited to using just a handful of musical excerpts, and they focus on the engineering aspects of time-stretching multiple songs simultaneously [7] or on visualisation as part of the mashup making process [12]. While these elements are certainly important, we believe that there are many opportunities for the development of MIR techniques within the field of music mashups. Indeed, mashup creation was recently highlighted as one of the "grand challenges" of MIR [5, p.222].

It is within this light that we propose *AutoMashUpper*, a system for making automatic multi-song mashups, as shown in Figure 1. The main novelty of our system lies

[1] http://www.djprince.no/site/DMG.aspx
[2] http://www.idmt.fraunhofer.de/en/Service_Offerings/technologies/e_h/harmonic_mixing_tool.html
[3] http://mashup.mixedinkey.com/

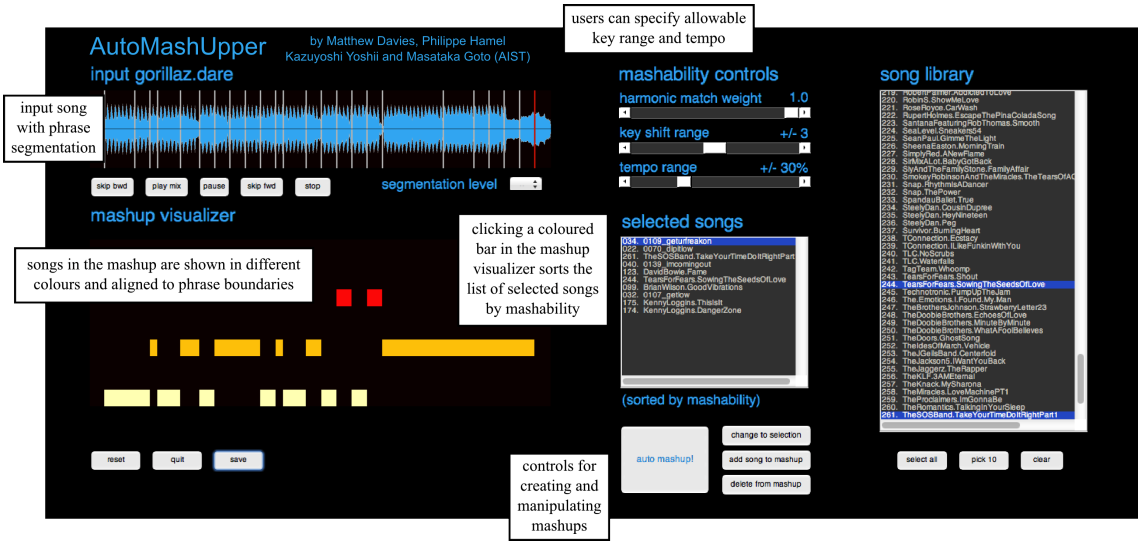


Figure 2. Screenshot of the AutoMashUpper user interface. Additional descriptions of the functionality are overlaid.

in the estimation of what we term "mashability" - a measure of how well two songs fit, or mash together. Looking beyond the functionality of existing mashup systems which guide users to songs with matching key signatures and similar tempi, we incorporate a measure of harmonic similarity of beat synchronous chromagrams. This allows us to look for deeper matches than those possible from key-signature alone. Furthermore our measure of mashability can identify matches between songs in completely different key-signatures, by directly exploiting the knowledge that songs can be pitch-shifted by some number of semitones to "force" a match.

By identifying the phrase-level structure of a given input song, AutoMashUpper can determine the mashability between each phrase section of the input and songs in a user's music collection. In this way, multiple songs can be used in the mashup at different times and this can radically increase the range and variety of possible mashups. To produce the mashup, we use existing techniques for pitch-shifting and time-stretching for harmonic and temporal alignment respectively, and loudness adjustment to create a balanced sound mixture.

In addition to the fully automatic mode, AutoMashUpper has a user interface to allow visualisation and manipulation of mashups, as shown in Figure 2. When creating a mashup, users can specify a list of songs to choose from, modify the mashability parameters and change the granularity of the phrase segmentation. Once created, users can also switch, add, or remove sections from the mashup to suit their taste, and additionally save the results for re-use.

The remainder of this paper is structured as follows. In Section 2 we describe the phrase-level segmentation used to partition songs in AutoMashUpper. In Section 3 we present our method for mashability estimation and describe how we produce the mashups. We then present the interface of AutoMashUpper in Section 4 and illustrate its main modes of operation. Finally, in Section 5, we discuss the potential impact of our system, towards motivating further research into mashup generation, and present some areas for future work.

2. PHRASE-LEVEL SEGMENTATION

A central component of our mashup system, and the key to enabling multiple songs to be used to produce the musical result, is a phrase-level segmentation of the input. While much research has been conducted into structural segmentation of music signals (e.g., [11, 13]) their goal is to identify boundaries of long sections corresponding to intro, verse and chorus, and to apply labels to these sections to identify repetitions. For our purpose in creating mashups, we require a similar type of analysis, however our concern is not directly in labelling the sections, but rather in precisely identifying temporal boundaries. Furthermore we wish to identify shorter sections corresponding to musical phrases, rather than longer time scale structure such as verse or chorus.

Through informal experimentation with existing segmentation algorithms with publicly available implementations (e.g., [13]) we discovered that it was not trivial to reliably sub-divide the estimated sections into downbeat synchronous phrases. On this basis, and in the interest of avoiding multiple separate stages of processing the input signal, we devise our own method for phrase segmentation, adapting elements from existing systems to suit our needs. Since an important element of mashups is the harmonic compatibility of the mixed music signals, we base our phrase-level segmentation on a harmonic representation of the input.

To generate the harmonic signal representations for phrase-level segmentation and the subsequent estimation of mashability, we use the NNLS Chroma plugin [8] within Sonic Annotator [2]. Given an input audio signal, we extract three results from the NNLS Chroma plugin: the global tuning, t, of the input, an 84-bin (7 octave), tuned semitone spectrogram, S, and a 12-dimensional chromagram (the distribution of energy across the chromatic pitch

classes in a musical octave), C. All outputs are extracted using the default parameters. To create beat-synchronous versions of S and C, we use the QM Vamp beat tracking plugin in Sonic Annotator, and take the median across the time frames per beat. For ease of notation, we will continue to use S and C to refer to the beat-synchronous versions.

To simplify our approach, we make the following assumptions about the songs to be used for mashups: all phrase sections are a whole number of measures, all songs have a constant 4/4 time-signature, and the input tempo is approximately constant.

To determine the phrase boundaries, we group the beat-synchronous frames of S into measures, to create a downbeat-synchronous semitone spectrogram. To identify the downbeats we used a modified version of the method by Davies and Plumbley [3] with S as the main input. As shown in [13] it can be beneficial for segmentation performance to "stack" beat frames together when estimating section boundaries. In this way, we group sets of four consecutive beat frames (starting at each downbeat and without overlap) to create a downbeat-synchronous stacked semitone spectrogram.

Given the beats and downbeats, we then follow the classical approach of Foote [4] for structural segmentation. We calculate a self-similarity matrix from the downbeat-synchronous stacked semitone spectrogram using the Itakura-Saito distance [6] and slide a Gaussian checkerboard kernel along the main diagonal to generate a novelty function to emphasise section boundaries. As shown in [4] the size of this kernel has a direct impact on the level of the segmentation and temporal precision of the boundaries. Since our interest is in finding short phrase-level sections, we use a small kernel of size eight downbeats. To obtain an initial set of phrase boundaries we peak-pick the resulting novelty function. We then employ a technique derived from [11] to maximise the regularity of the detected phrase boundaries. A graphical example is shown in Figure 3 along with a flow chart in Figure 4(a).

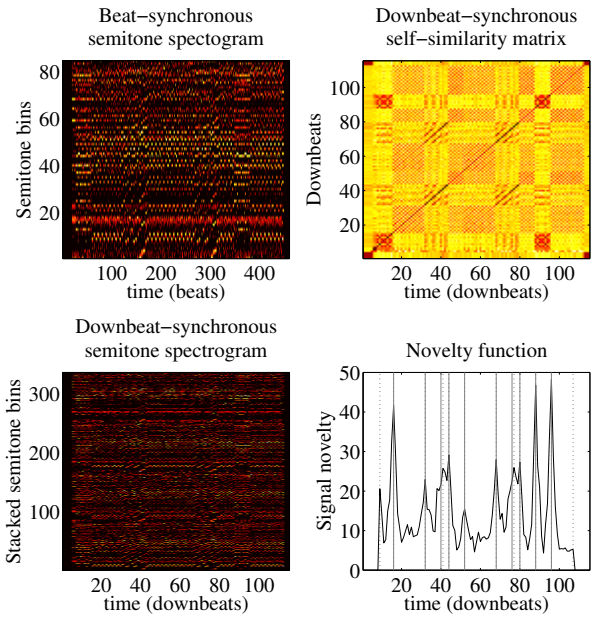


Figure 3. Phrase-level segmentation overview. (top left) a beat synchronous tuned semitone spectrogram. (bottom left) a downbeat-synchronous spectrogram, where groups of four beat frames are stacked into measures. (top right) a self-similarity matrix generated from the downbeat-synchronous semitone spectrogram. (bottom right) a novelty function whose peaks highlight likely phrase boundaries. The vertical dotted lines show raw phrase boundaries and the solid grey lines show the result of regularity-constrained realignment.

3. MAKING MASHUPS

This section describes how the mashability is estimated between beat-synchronous chromagrams for each phrase section of the input song, and the songs in a music collection. Then we address the requisite processing to physically create the mashup itself. A graphical overview of the complete mashup creation process is shown in Figure 4.

3.1 Estimating Mashability

Once the set of phrase segment boundaries has been determined, we turn our attention to finding a match for each phrase section of the input with songs in the users' music collection by estimating what we refer to as "mashability". For each song in the collection, we pre-calculate a beat-synchronous chromagram using the techniques described in Section 2 prior to estimating the mashability.

In contrast to existing systems which guide users towards mixing songs with matching key signature and have similar tempi, we argue that there is a much wider scope of potential matches (and potentially more interesting musical results) by considering mashups between songs in different keys and tempi. In effect, our approach is not only to look for matches according to the existing properties of songs, but also to look for matches in a kind of "transform" domain, in the knowledge that we can subsequently use time-stretching to temporally align songs, and pitch-shifting (by some number of semitones) to create, or indeed "force" a harmonic alignment.

We base our estimation of mashability around the harmonic similarity between beat-synchronous chromagrams. For the current phrase-section p of length K beats from the input song, i, we isolate the beat-synchronous chromagram $C_{i,p}$ (a 12-by-K matrix). To facilitate the search across different key shifts, we rotate the chroma bins of $C_{i,p}$ across a range of integer semitone shifts, r, which can be set from 0 to ± 6 semitones according to user preference. For each key-shifted chroma section of the input, $C_{i,p,r}$ we measure its harmonic similarity across each rotational shift, r to all possible beat increments k, (for K-beat frame chromagrams) for each song n in the user's song collection using the Cosine similarity,

$$H_n(r,k) = \frac{C_{i,p,r} \cdot C_{n,p,k}}{||C_{i,p,r}||\ ||C_{n,p,k}||} \quad (1)$$

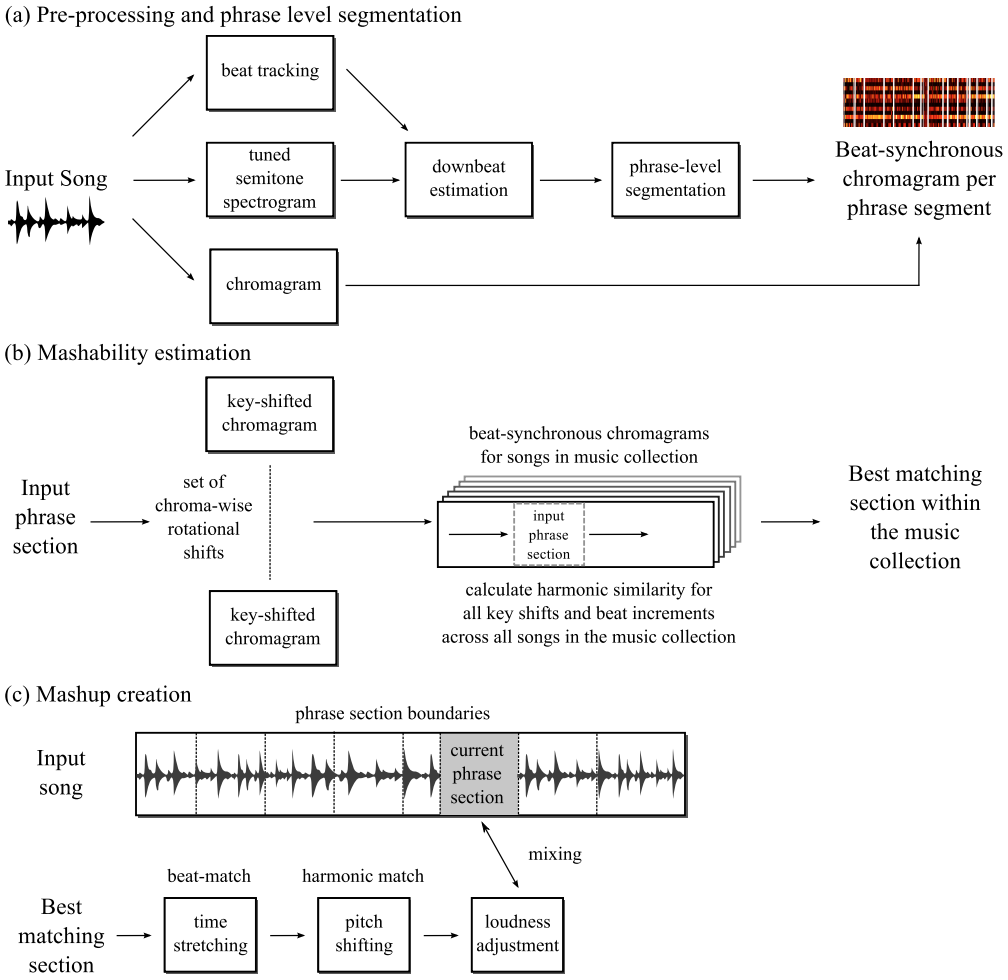


Figure 4. Overview of the mashup creation process. (a) Pre-processing and phrase level segmentation, (b) mashability estimation, and (c) mashup creation.

where high harmonic similarity will have H close to unity and low similarity will have H close to zero. A graphical overview is shown in Figure 4(b).

To move from harmonic similarity to mashability, M, we include an additional term which rewards songs whose tempo, T_n is within a user specified ratio, η, of the input tempo, T_i, such that:

$$M_n(r,k) = \begin{cases} H_n(r,k) + \gamma, & \text{if } |1 - |(T_i/T_n)|| \leq \eta \\ H_n(r,k), & \text{otherwise} \end{cases} \quad (2)$$

where γ=0.2 was found to give favourable results. Note that the greater the value of η the more permissive the system in terms of allowable tempo matches between the input and songs in the collection.

Once the mashability has been calculated across the song collection we find the song, n, beat increment, k, (i.e. starting point) and rotational shift, r, which lead to the highest mashability for the current input phrase section, $C_{i,p,r}$. We apply the rotations of chroma to the *input* chromagram and not to the songs in the database, which remain unaltered by the search across mashability space. Therefore, when we come to implement any required key-shift to match the selected song with the input, we must pitch-shift the selected song by $-1 * r$ semitones.

By measuring the harmonic similarity across all beat increments k and rotational shifts r we create a large search space, which in turn gives the highest possibility for finding regions of high harmonic similarity. Furthermore we have found matching between chroma matrices at incremental beat shifts, rather than looking at individual chroma frames, we can implicitly capture aligned chord changes between songs – a factor we have found improves the quality of the resulting mashup.

3.2 Mashup Creation

The final part of the automatic mashup creation process is to transform the selected section and mix it with the input, as shown in Figure 4(c).

To create this mix, several steps are required. First, we use the open-source Rubberband time-stretching and pitch-shifting library [4] to temporally align (or "beat-match") the matching section with the current phrase section of the input song. This is achieved using the `mapfile` function (within Rubberband) which specifies a set of anchor points, i.e., the beats of the song to be transformed, and a cor-

[4] `http://breakfastquay.com/rubberband/`

responding set of target times – the beats of the current phrase section of the input.

Once aligned in time, the matching section is then harmonically aligned to the input phrase using the pitch-shifting functionality of Rubberband. This harmonic alignment addresses two areas: pitch-shifting by the required number of semitones, r, to match the key-signature of the songs, and a tuning correction, identified as the ratio between the estimated tuning for the input song and selected matching song. In the event that both a tuning and pitch correction are required, we combine these factors into a single processing call to Rubberband. In the event that the two songs are already matched in key, (i.e. r=0) and the difference in tuning is less than 1Hz, then the mashup can be made by beat-matching alone.

The final stage in mixing the sections of the two songs together, is to address any imbalance in the loudness between the current input section and the transformed match. To this end, we estimate the perceptual loudness in the input phrase section and transformed signal using the Replay Gain algorithm [10]. While traditionally used to equalise loudness between songs, we wish to give greater prominence in the mix to the input song, hence we scale the amplitude of transformed section to have 90% of the loudness of the input phrase section.

4. AUTOMASHUPPER

Up to this point we have described the backend processing to enable the *automatic* creation of mashups. To allow users of AutoMashUpper to be involved in the mashup creation process, we have built a user interface, which is shown in Figure 2. To illustrate the functionality of the user interface and to provide sound examples, demonstration videos are available here [5] .

4.1 Interface Overview

The layout of the interface is split into three sections. On the left hand side there are two main panels, the top for visualising the waveform of the input song and the estimated phrase level section boundaries. Below this is the mashup visualizer which shows the songs currently used in the mashup. In addition, a set of playback controls are included for listening to the input song and the mashup. On the right hand side is a panel containing the list of pre-analysed songs in the music collection. Below this listbox are buttons to select songs from the library to use in the mashup. In the central panel we have a listbox which shows the current songs selected for use in the mashup, as well a set of sliders for manipulating the parameters of the mashabilty calculation. These specify the range of allowable key-shifts, and the preferred tempo range. Beneath the list box are buttons for creating the automatic mashup and then subsequent manipulation of the result. This functionality is described in the following subsection.

[5] `http://www.youtube.com/user/automashupper`

4.2 User interaction

The typical scenario we envisage for AutoMashUpper is as follows. The user loads a song of their choice into the system, after which a waveform of the input song appears in the top left panel along with vertical bars to indicate the estimated phrase section boundaries. The user can listen to the input song and click on different sections for the playback to jump directly to these parts of the song. In addition the user can explore finer segmentations where phrase sections can be sub-divided into 16, 8 or 4-beat units using the segmentation level drop-down menu. Having selected a segmentation level, the user can then choose a set of songs from the song library on the right hand side of the interface. For this, three options are available: i) to manually select a sub-set of their choice; ii) to select all of the songs in the library; or iii) to pick ten random songs.

When manually choosing a subset, we have found that only picking songs from the same artist, or the same album, i.e., *artist-level-mashups* or *album-level-mashups*, can lead to very pleasing results due to high timbral compatibility.

The songs chosen by the user then appear in the listbox of selected songs in the middle of the interface. Using the sliders above this listbox, the user can specify how wide a range of key shifts and tempi to allow in the mashability estimation. Specifying a small range of key shifts and tempi can lead to somewhat conservative results, whereas allowing a wide range of possibilities in the mashup space can facilitate better matches, but perhaps at the cost of creating more unusual results, for example where a transformed song could be pitch-shifted up or down by five semitones or radically changed in speed.

Once AutoMashUpper has been parameterised, the user can then hit the **auto mashup!** button to generate a mashup. Or, the user may simply hit this button right after loading the input song. As each section is identified and added to the mashup it appears in the lower left hand panel, where each song is displayed in a different colour. When the processing has finished the user can listen to the result – once again with the ability to navigate between phrase sections by clicking in the appropriate region of the waveform representation or the mashup visualizer panel. During playback, a red vertical line indicates the currently playing phrase section of the input song.

Clicking a particular bar in the mashup visualizer will highlight the name of the chosen song in the selected songs listbox. It will also re-order the remaining songs in descending order of mashability. At this point the user can make a subjective judgement over whether they like the mashup as it is or which to change it. The user has three options: first, they can delete the currently used section from the mashup, second, they can choose a different song from the selected songs listbox to replace it or third, they can choose to add another song from the list to the mashup. If the user is pleased with the resulting mashup they can use save button, which will create time-stamped .wav files for the input song, the generated mashup by itself and the mixture of the input and mashup. In addition it records a screenshot of the interface to show the list of songs used

and mashability parameters.

5. DISCUSSION

In this paper we have presented *AutoMashUpper*, a system for the creation of multi-song mashups. Our main contribution in this work is a method for mashability estimation which enables the automatic creation of music mashups. Our work forms part of the emerging field of creative-MIR, where music analysis and transformation techniques are used within real applications towards the transfer of knowledge outside the MIR research community. We have designed AutoMashUpper with the aim of assisting users (who might lack music composition skills) to become music creators through simple interactions with a user interface. Our hope is that AutoMashUpper will encourage users to explore a wide space of mashup possibilities by manipulation of the mashability parameters, perhaps even creating a new genre of "auto" mashups.

We believe a particular advantage of the automatic approach to searching for mashability within a large collections of songs is that such a system can uncover musical relationships which might otherwise never be found. This is especially relevant if we consider the size of the search space when allowing for matches at the phrase-level of songs. Our current system uses a catalogue of around 300 songs from which we have been able to create many interesting mashups, with minimal effort. Furthermore, even when the phrase-level segmentation has errors, this has the potential to create unusual and unexpected results.

We have been particularly surprised by the quality of results achieved when using the "pick ten random songs" option in AutoMashUpper. This indicates that many hidden relationships exist between different sections of songs, and discovering them in the context of a music mashup appears a particularly good way to enjoy them. In this sense, the possibilities when applying this system to a very large music collection could be almost endless. However, the transition from a medium-sized collection to a very large one presents many challenges due to scalability and increased computational cost, and would require a much faster search technique, (e.g., [1]). We intend to explore this area within our future work as well as investigating source separation methods (e.g., [9]) to offer users even greater mashup creation possibilities.

Since mashups, by definition, contain multiple songs playing at once, they represent an interesting category of music from an auditory scene analysis perspective, where it is listeners' familiarity with songs in the mashup which allow them to understand a musical scene which might otherwise be too complex to process and hence appreciate [5]. To further explore these ideas and to address the lack of a formal evaluation of AutoMashUpper, we plan to undertake subjective listening tests to explore listeners' levels of musical engagement and understanding of mashups.

Looking beyond the current version of AutoMashUpper, we recognise that mashability is not fully explained by harmonic similarity alone, and we can envisage many additional uses of MIR techniques for creating more sophisticated measures of mashability, e.g. by exploring rhythmic and spectral compatibility. On this basis we strongly encourage other researchers to explore mashup creation methods to expand the field of creative MIR.

6. ACKNOWLEDGMENTS

This work was supported by OngaCREST, CREST, JST.

7. REFERENCES

[1] T. Bertin-Mahieux and D. Ellis. Large-scale cover song recognition using the 2D Fourier transform magnitude. In *Proceedings of 13th International Society for Music Information Retrieval Conference*, pages 241–246, 2012.

[2] C. Cannam, M. O. Jewell, C. Rhodes, M. Sandler, and M. d'Inverno. Linked data and you: Bringing music research software into the semantic web. *Journal of New Music Research*, 39(4):313–325, 2010.

[3] M. E. P. Davies and M. D. Plumbley. A spectral difference approach to extracting downbeats in musical audio. In *Proceedings of the 14th European Signal Processing Conference (EUSIPCO)*, 2006.

[4] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of IEEE International conference on multimedia and expo*, pages 452–455, 2000.

[5] M. Goto. Grand challenges in music information research. In M. Muller, M. Goto, and M. Schedl, editors, *Multimodal Music Processing*, pages 217–225. Dagstuhl Publishing, 2012.

[6] F. Itakura and S. Saito. Analysis synthesis telephony based on the maximum likelihood method. In *Proceedings of the International Congress on Acoustics*, pages 17–20, 1968.

[7] G. Griffin Y. E. Kim and D. Turnbull. Beat-sync-mash-coder: A web application for real-time creation of beat-synchronous music mashups. In *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 437–440, 2010.

[8] 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*, pages 135–140, 2010.

[9] 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):71–82, 2013.

[10] D. Robinson. *Perceptual model for assessment of coded audio*. PhD thesis, Department of Electronic Systems Engineering, University of Essex, 2002.

[11] G. Sargent, F. Bimbot, and E. Vincent. A regularity-constrained viterbi algorithm and its application to the structural segmentation of songs. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 483–488, 2011.

[12] N. Tokui. Massh!: A web-based collective music mashup system. In *Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts*, pages 526–527, 2008.

[13] R. J. Weiss and J. P. Bello. Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization. In *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pages 123–128, 2010.

LEARNING BINARY CODES FOR EFFICIENT LARGE-SCALE MUSIC SIMILARITY SEARCH

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

ABSTRACT

Content-based music similarity estimation provides a way to find songs in the unpopular "long tail" of commercial catalogs. However, state-of-the-art music similarity measures are too slow to apply to large databases, as they are based on finding nearest neighbors among very high-dimensional or non-vector song representations that are difficult to index.

In this work, we adopt recent machine learning methods to map such song representations to binary codes. A linear scan over the codes quickly finds a small set of likely neighbors for a query to be refined with the original expensive similarity measure. Although search costs grow linearly with the collection size, we show that for commercial-scale databases and two state-of-the-art similarity measures, this outperforms five previous attempts at approximate nearest neighbor search. When required to return 90% of true nearest neighbors, our method is expected to answer 4.2 1-NN queries or 1.3 50-NN queries per second on a collection of 30 million songs using a single CPU core; an up to 260 fold speedup over a full scan of 90% of the database.

1. INTRODUCTION

Content-based music similarity measures allow to scan a collection for songs that *sound* similar to a query, and could provide new ways to discover music in the steadily growing catalogs of online distributors. However, an exhaustive scan over a large database is too slow with state-of-the-art similarity measures. A possible solution are *Filter-and-Refine* indexing methods: To find the k nearest neighbors (k-NN) to a query, a prefilter returns a small subset of the collection, which is then refined to the k best items therein.

Here, we consider the following scenario: (1) We have a commercial-scale music collection, (2) we want to return on average at least a fraction Q of the items an exhaustive scan would find, and (3) we cannot afford costly computations when a song enters or leaves the collection (ruling out nonparametric methods, or precomputing all answers). We then search for the fastest indexing method under these constraints.

Compared to existing work on approximate k-NN search, what makes this quest special is the nature of state-of-the-art music similarity measures, and a low upper bound on database sizes: The largest online music store only offers 26 million songs as of February 2013, while web-scale image or document retrieval needs to handle billions of items.

Among the first approaches to fast k-NN search were space partitioning trees [1]. McFee et al. [12] use an extension of k-d trees on 890,000 songs, reporting a 120 fold speedup over a full scan when missing 80% of true neighbors. No comparison to other methods is given.

Hash-based methods promise cheap lookup costs. Cai et al. [2] apply Locality-Sensitive Hashing (LSH) [6] to 114,000 songs, but do not evaluate k-NN recall. Torralba et al. [23] learn binary codes with Restricted Boltzmann Machines (RBMs) for 12.9 mio. images, achieving 80% 50-NN recall looking at 0.2% of the database and decisively outperforming LSH. Using similar techniques, other researchers learn codes for documents [16] and images [8, 13], but, to the best of our knowledge, never for songs.

Pivot-based methods map items to a vector space only using distances to landmark items. Rafailidis et al. [15] apply L-Isomap to 9,000 songs. Schnitzer et al. [19] apply FastMap to 100,000 songs, achieving 80% 10-NN recall looking at 1% of the collection. Chávez et al. [3] map items to byte vectors with good results, but on our data, their approach needs 50% of a collection to find 80% of 1-NN.

In what follows, we will adapt the most promising methods for application on two music similarity measures and evaluate their performance under our scenario.

2. FILTER-REFINE COST MODEL

The cost of answering a query using a filter-and-refine approach can be decomposed into the cost for running the prefilter and the cost for refining the selection to k items:

$$\mathrm{cost}_{\mathrm{filtref}}(n,k) = \mathrm{cost}_{\mathrm{filt}}(n,k) + \mathrm{cost}_{\mathrm{ref}}(\mathrm{n}_{\mathrm{filt}}(n,k),k)$$

We assume that the refine step is a linear scan over the candidates returned by the prefilter, picking the k best:

$$\mathrm{cost}_{\mathrm{ref}}(n_f,k) = n_f \cdot (R + \log(k)\cdot S), \qquad (1)$$

where R is the cost for computing the distance of a query to a candidate, and $\log(k)\cdot S$ is the cost for updating a k-element min-heap of the best candidates found so far. [1]

[1] This model is not entirely correct, as the heap is generally not updated for each item. However, for $k \leq 100 \ll n_f$, we found the sorting cost to be indeed linear in n_f, which is all we need for our argument.

As a baseline method meeting our requirement of finding on average a fraction Q of the true neighbors, we adopt a zero-cost prefilter returning a fraction Q of the dataset:

$$\mathrm{cost}_{\mathrm{baseline}}(n,k) = \mathrm{cost}_{\mathrm{ref}}(Q \cdot n, k)$$

Under these assumptions, using a prefilter gives the following speedup factor over the baseline:

$$\begin{aligned}\mathrm{spu}(Q,n,k) &= \frac{\mathrm{cost}_{\mathrm{ref}}(Q \cdot n, k)}{\mathrm{cost}_{\mathrm{filtref}}(n,k)} \\ &= \left(\frac{\mathrm{cost}_{\mathrm{filt}}(n,k) + \mathrm{cost}_{\mathrm{ref}}(\mathrm{n}_{\mathrm{filt}}(n,k),k)}{Q \cdot \mathrm{cost}_{\mathrm{ref}}(n,k)}\right)^{-1} \\ &= Q \cdot \left(\frac{\mathrm{cost}_{\mathrm{filt}}(n,k)}{\mathrm{cost}_{\mathrm{ref}}(n,k)} + \frac{\mathrm{n}_{\mathrm{filt}}(n,k)}{n}\right)^{-1} \\ &= Q \cdot \big(\rho_{\mathrm{t}}(n,k) + \rho_{\mathrm{s}}(n,k)\big)^{-1} \qquad (2)\end{aligned}$$

We see that making the prefilter fast compared to a full scan (small ρ_{t}) is just as important as making the filter selective (small ρ_{s}). More specifically, we need to minimize the sum of the two ratios to maximize the speedup factor.

As we will see in our experiments, some existing methods put too much emphasis on a fast prefilter, resulting in $\rho_{\mathrm{t}}(n,k)$ being orders of magnitude smaller than $\rho_{\mathrm{s}}(n,k)$ especially for large databases. In this work we will balance the two ratios better to maximize the performance.

3. MUSIC SIMILARITY MEASURES

For our experiments, we choose two very different similarity measures: One based on high-dimensional vectors, and another based on Gaussian distributions.

3.1 Vector-Based Measure

Seyerlehner et al. [20] propose a set of six *Block-Level Features* to represent different aspects of a song's audio content, totalling in 9448 dimensions. These features work well for genre classification and tag prediction [21], and similarity measures based on them ranked among the top three algorithms in the MIREX Audio Music Similarity (AMS) tasks 2010–2012. For the similarity measure, the six feature vectors are individually compared by Manhattan distance, and the resulting feature-wise distances are combined to form the final similarity estimation.

To combine the feature-wise distances, they must be brought to similar scale. Instead of finding six appropriate scaling factors on an arbitrary dataset, Seyerlehner et al. normalize the feature-wise distance matrices for the handled collection: This *Distance Space Normalization (DSN)* processes each distance matrix entry by subtracting the mean and dividing by the standard deviation of its row and column. [2] The six normalized matrices are added up and normalized once again to form the final similarities.

While the normalizations seem unnecessarily complex, Flexer et al. [5] recently showed that they remove *hubs* – items appearing as neighbors of undesirably many other items – and are vital to achieve state-of-the-art results.

[2] When it is infeasible to compute full distance matrices, the song-wise distance statistics can be approximated from a random subset of the collection and stored with each feature vector.

3.2 Gaussian-Based Measure

As a second method, we use the timbre model proposed by Mandel and Ellis [11]: Each song is represented by the mean vector and covariance matrix of its frame-wise Mel-Frequency Cepstral Coefficients (MFCCs). [3] Song distances are computed as the symmetrized Kullback-Leibler divergence between these multivariate Gaussian distributions [17, p. 24], and normalized with DSN.

This measure does not reach state-of-the-art performance on its own, but forms the main component of [14], which ranked among the top two algorithms in the MIREX AMS tasks 2009–2012. Furthermore, it is easy to reproduce and allows direct comparison to Schnitzer et al. [18, 19].

4. INDEXING METHODS

We will evaluate seven different methods for fast k-NN search: One oblivious to the indexed dataset, four based on song models and two based on song similarities.

4.1 Locality-Sensitive Hashing (LSH)

For vectors in an Euclidean space, the family of projections onto a random line, followed by binary thresholding or fixed-width quantization, is *locality-sensitive*: For such projections, two close items are more probable to be mapped to the same value than two items far apart [6].

LSH uses $L \cdot K$ projections to map each item x_i to L discrete K-dimensional vectors $h_l(x_i)$. Using L conventional hash-table lookups, it can quickly find all items x_j matching a query q in at least one vector, $\exists_{l \le L} h_l(q) = h_l(x_j)$.

Here, this serves as a prefilter for finding neighbor candidates. Increasing K makes it more likely for candidates to be true nearest neighbors, but strongly reduces the candidate set size. Increasing L counters this, but increases query and storage costs. As a complementary way to increase the number of candidates, Multi-probe LSH [10] considers items with a *close* match in one of their vectors.

4.2 Principal Component Analysis (PCA)

PCA finds a linear transformation $\boldsymbol{y} = \boldsymbol{W'x}$ of Euclidean vectors $\boldsymbol{x_i} \in \mathcal{X}$ to a lower-dimensional space minimizing the squared reconstruction error $\sum_i \|\boldsymbol{x_i} - \boldsymbol{WW'x_i}\|_2^2$.

Nearest neighbors in the low-dimensional space are good candidates for neighbors in the original space, so a linear scan over items in the low-dimensional space serves as a natural prefilter. The candidate set size can be tuned at will to achieve a target k-NN recall. Increasing the dimensionality of the space allows to reduce the candidate set size, but increases prefilter costs.

4.3 Iterative Quantization (ITQ)

ITQ [7] finds a rotation of the PCA transformation minimizing squared reconstruction error after bit quantization of the low-dimensional space: $\sum_i \|\boldsymbol{x_i} - \boldsymbol{W}\,\mathrm{b}(\boldsymbol{W'x_i})\|_2^2$, where $\mathrm{b}_i(\boldsymbol{z})$ is 1 for positive z_i and 0 otherwise.

[3] Specifically, we use frames of 46 ms with 50% overlap, 37 Mel bands from 0 Hz to 11025 Hz and retain the first 25 MFCCs.

It can serve as a prefilter just like PCA, but using bit vectors reduces computational costs for the linear scan. For compact bit codes, neighbors within a small hamming distance of a query can alternatively be found with a constant number of conventional hash table lookups.

4.4 PCA Spill Trees

K-d Trees [1] are binary trees recursively partitioning a vector space: Each node splits the space at a hyperplane, assigning the resulting half-spaces to its two child nodes. Spill Trees [9] allow the half-spaces to overlap, making it less likely for close items to be separated. McFee et al. [12] additionally propose to choose hyperplanes perpendicular to a dataset's principal components, and to strongly restrict the depth of the tree. In the resulting *PCA Spill Tree*, each item ends up in one or more leaves, with similar items often sharing at least one leaf. Locating the leaves for an item is linear in the database size [12, Sec. 3.6], but can be avoided by precomputing all leaf sets.

As in [12], we regard all items in the leaf sets of a query to be candidate neighbors. The candidate set size can be increased by decreasing the tree depth or by increasing the overlap at each node.

4.5 Auto-Encoder (AE)

An AE finds a nonlinear transformation of inputs to a low-dimensional or binary code space and back to the input space, minimizing the difference between inputs and reconstructions (e.g., ℓ_2 distance for Euclidean input vectors). Similar to PCA and ITQ, candidate neighbors to a query can quickly be found in the code space.

The transformation is realized as an artificial neural network and can be optimized with backpropagation. For deep networks, it is helpful to initialize the network weights using Restricted Boltzmann Machines (RBMs). Salakhutdinov et al. [16] were the first to use a deep AE for approximate nearest neighbor search, under the term *Semantic Hashing*, and describe the method in detail.

4.6 Hamming Distance Metric Learning (HDML)

HDML [13] finds a nonlinear transformation to a binary code space optimized to preserve neighborhood relations of the input space. Specifically, for any triplet (x, x^+, x^-) of items for which x is closer to x^+ than to x^- in the input space, it aims to have x closer to x^+ than to x^- in the code space. Again, the transformation is realized as an artificial neural network, optimized with backpropagation, and HDML can be used as a prefilter just like ITQ or AE.

4.7 FastMap

FastMap [4] maps items to a d-dimensional Euclidean space based on their (metric) distances to d previously chosen pivot pairs in the input space. Schnitzer et al. [19] show how to apply FastMap to Gaussian-based models and propose an improved pivot selection strategy we will adopt.

FastMap serves as a prefilter like PCA, but supports non-vector models as it is purely distance-based.

5. EXPERIMENTS

We will now compare the seven indexing methods empirically, conducting a range of retrieval experiments.

5.1 Dataset and Methodology

From a collection of 2.5 million 30-second song excerpts used in [18, 19], we randomly select 120k albums of 120k different artists. We use 10k albums (124,013 songs) for training, 20k albums (246,117 songs) for validation and the remaining 90k albums (1,101,737 songs) for testing. In addition, we use 20k albums (253,347 songs) of the latter as a smaller test set.

For each applicable combination of similarity measure and indexing method, we will train different parameterizations of the method on the training set and determine the speedup over the baseline (Eq. 2) for retrieving on average 90% of the 1 or 50 nearest neighbors on the validation set. We will then evaluate the best parameterizations on the small test set to ensure we did not overfit on the validation set, and use the large test set to assess the methods' scalability.

5.2 Vector-based Measure

To be able to compute the speedup, we first determine the costs of the similarity measure. [4] Computing 1 million 9,448-dimensional Manhattan distances takes 2.361 s, finding the (indices of) the smallest 100 distances takes 1.17 ms, and both costs scale linearly with the collection size, as assumed in Eq. 1. Costs for the approximate DSN are negligible (see Sect. 3.1). For prefilters based on a linear scan, computing 1 million 80-dimensional ℓ_2 distances takes 22 ms, and computing 1 million 1024-bit hamming distances takes 9.8 ms. The costs of finding the best candidates in a linear scan depend on the candidate set size; we will use separate measurements for each case.

PCA: We start by evaluating PCA as a prefilter, as it proved useful as a preprocessing step for most other filters as well. To mimic how the similarity measure is combined from six features, we first apply PCA to each feature separately, compressing to about 10% of its size, then rescale each feature to unit mean standard deviation (this brings the distances to comparable ranges, and forms good inputs for the AE later) and stack the compressed features to form an 815-dimensional vector. Finally, we apply another PCA to compress these vectors to a size suitable for prefiltering.

In Table 1, we see that this cuts down query costs: For retrieving 90% of the true nearest neighbors, prefiltering with a linear scan over 40-dimensional PCA vectors takes $\rho_\mathrm{t} = 0.56\%$ the time of a full scan and only needs to examine $\rho_\mathrm{s} = 0.26\%$ of the database afterwards, resulting in a 110 fold speedup over the baseline ($0.9/(0.0052+0.0026)$, Eq. 2). For retrieving 50-NN, it needs a larger candidate set, increasing the prefilter costs (higher sorting costs to find the candidates), but still achieving a 47 fold speedup.

[4] All timings are reported on an Intel Core i7-2600 3.4 GHz CPU with DDR3 RAM, use a single core, and leverage AVX/POPCNT instructions. Implementations are in C, carefully optimized to maximize throughput.

Method		1-NN ρ_t(%)	1-NN ρ_s(%)	1-NN spu	50-NN ρ_t(%)	50-NN ρ_s(%)	50-NN spu
PCA	20 dim	0.38	0.59	93x	0.58	1.85	37x
	40 dim	0.56	0.26	110x	0.71	1.20	47x
	80 dim	1.01	0.18	76x	1.16	1.12	40x
LSH	8 bit	0.00	17.23	5x	0.00	23.82	4x
	16 bit	0.00	6.66	14x	0.00	11.00	8x
	20 bit	0.00	4.68	19x	0.00	8.42	11x
mp-LSH	128x16 bit	0.83	11.12	8x	0.83	34.18	3x
	64x32 bit	0.83	6.03	13x	0.83	12.23	7x
	32x64 bit	0.83	3.65	20x	0.83	7.59	11x
	16x128 bit	0.83	3.58	20x	0.83	7.11	11x
	1x256 bit	0.10	3.85	23x	0.10	7.98	11x
	8x256 bit	0.83	2.80	25x	0.83	6.22	13x
ITQ	64 bit	0.03	5.43	16x	0.03	9.94	9x
	128 bit	0.05	4.74	19x	0.05	8.27	11x
	Spill Tree	0.00	10.25	9x	0.00	21.27	4x
AE	64 bit	0.03	2.14	41x	0.03	4.40	20x
	128 bit	0.05	0.57	144x	0.05	2.47	36x
	256 bit	0.10	0.24	**265x**	0.10	1.28	65x
	512 bit	0.21	0.14	258x	0.21	0.93	79x
	1024 bit	0.42	0.09	177x	0.42	0.70	**81x**
FastMap	40 dim	0.77	1.56	39x	1.11	3.72	19x
	80 dim	1.20	1.36	35x	1.53	3.43	18x
	128 dim	1.73	1.20	31x	2.03	3.07	18x

Table 1. Results for the vector-based music similarity measure on the validation set of 246,117 songs: Ratio of prefilter time to full scan (ρ_t), ratio of candidate set to dataset size (ρ_s) and resulting speedup over baseline (spu) for retrieving 90% of 1 and 50 true nearest neighbors.

Varying the vector dimensionality changes the tradeoff between ρ_t and ρ_s, but does not improve the speedup.

LSH: We apply different versions of LSH to the 815-dimensional intermediate PCA representation.[5] First, we follow Slaney et al. [22] to compute optimal quantization width, dimensionality and table count for 90% 1-NN recall under the assumption that all projections are independent (it suggests 92.192, 25 and 430, respectively). To reach our target 1-NN recall, we need a 3-fold increase in table count and obtain $\rho_s = 16.52\%$, which is not competitive. Turning to binary LSH, we fix the dimensionality K to 8, 16 or 20 bit and increase L until we reach 90% recall (for 20 bits and 50-NN, we need 8353 hash tables). Even assuming zero prefilter costs, speedup is far below PCA. As a third alternative, we use a simple version of multi-probe LSH: We fix L and K, but consider all buckets within a hamming distance of r to the query in any of the tables. We increase r to reach the target k-NN recall, still achieving moderate speedups of up to 25x only.

ITQ directly builds on the PCA transform above, but maps items to bit vectors. Instead of directly tuning the candidate set size, we consider all items in a hamming ball of radius r around the query (in code space), and tune r. This avoids the sorting costs for finding the candidates. ITQ has small ρ_t, but large ρ_s, resulting in low speedups.

PCA Spill Tree: We build a tree with spill factor 0.1 (the best performing in [12]) and adjust the depth to reach our target recall. Assuming zero prefilter costs, it achieves poor speedups as it needs very large candidate sets.

AE: We train a deep AE on the 815-dimensional intermediate PCA representation, pretrained with stacked RBMs as in [8]. We use an encoder architecture of 1024-256-128-64 layers for the shorter codes, 1024-512 and 2048-1024 for the two longer codes.[6] We encourage binary codes by adding noise in the forward pass as in [16]; especially for 128 bits and more, this worked better than thresholding as in [8]. For 256 bits and less, it also helped to encourage zero mean code unit activations as in [13, Eq. 12].

We use the learned codes as in ITQ. We obtain a prefilter which is both faster than PCA and more selective, achieving a 265 fold speedup for 1-NN and 81 fold speedup for 50-NN queries. Note how the accuracy of longer codes pays off for 50-NN, while shorter codes win for 1-NN.

HDML did not yield any improvement over AE.

FastMap is about twice as fast as LSH or ITQ, but falls behind AE and PCA.

We evaluate the best-performing instantiations of each method on the small test set and find results to be very similar to Table 1. As the relative prefilter costs ρ_t stay the same anyway, we only show how the candidate set size ρ_s and 50-NN recall interact (Fig. 1). We can see that the 1024-bit AE again only needs about 0.7% of the dataset to find 90% of 50-NN, and we see that AE and PCA perform best over a wide range of target recall values. Besides, comparison with [12, Fig. 4] shows that our PCA Spill Tree performs similar to its first publication.

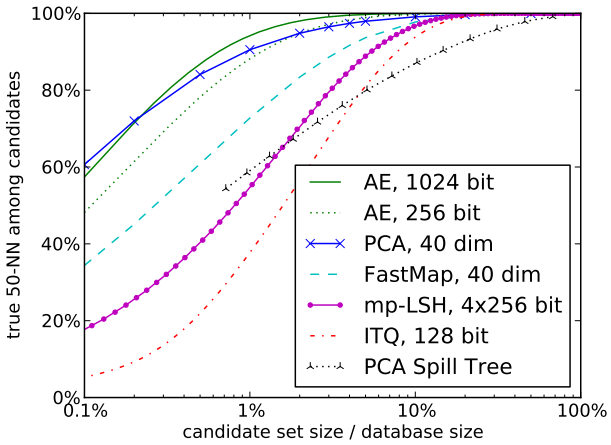


Figure 1. 50-NN recall versus candidate set size for the vector-based music similarity measure on the test set of 253,347 songs, averaged over all 253,347 possible queries.

[5] PCA is a useful stepping stone as the DSN (Sect. 3.1) invalidates any theoretical guarantees of LSH finding the nearest neighbors in the original space. Directly working on the 9448-dimensional vectors, rescaling the six components to comparable range, consistently gave worse results.

[6] Results are robust to the exact architecture as long as there is at least one layer before the code layer, and the first layer is wide enough.

Method		1-NN $\rho_t(\%)$	1-NN $\rho_s(\%)$	1-NN spu	50-NN $\rho_t(\%)$	50-NN $\rho_s(\%)$	50-NN spu
PCA	20 dim	6.18	16.03	4x	10.96	29.80	2x
	40 dim	6.00	14.04	4x	11.11	28.79	2x
AE	64 bit	0.06	11.89	8x	0.06	19.36	5x
	128 bit	0.11	6.95	13x	0.11	15.77	6x
	1024 bit	0.91	4.76	16x	0.91	13.13	6x
HDML	128 bit	0.11	1.46	57x	0.11	3.73	23x
	256 bit	0.23	1.37	56x	0.23	3.93	22x
	2x128 bit	0.23	1.15	**65x**	0.23	3.12	**27x**
	4x128 bit	0.45	1.09	58x	0.45	3.02	26x
	1024 bit	0.91	1.20	43x	0.91	4.65	16x
FastMap	40 dim	2.03	2.62	19x	3.37	6.48	9x
	80 dim	2.78	1.85	19x	3.85	4.94	10x
	128 dim	3.98	1.81	16x	5.03	4.85	9x

Table 2. Results for the Gaussian-based music similarity measure on the validation set of 246,117 songs.

5.3 Gaussian-based Measure

Again, we first determine the costs of the similarity measure: Computing 1 million symmetric Kullback-Leibler (sKL) divergences between 25-dimensional full-covariance Gaussian models takes 1.085 s, using precomputed inverse covariance matrices as in [17, Ch. 4.2]. Note that most indexing methods evaluated above are vector-based and not applicable to Gaussian models, so we expect the most from HDML and FastMap, but still try AE and PCA to be sure.

AE: In order for learned codes to be useful, they must reflect the input space. For the input space at hands, it seems natural to learn codes by minimizing the sKL divergence between inputs and reconstructions. For this to work, the AE must be forced to output valid covariance matrices Σ, otherwise it quickly learns to produce reconstructions that push the sKL divergence unboundedly below zero. We solve this by representing models in terms of the mean vector and Cholesky decomposition of Σ (multiplying the reconstructed Cholesky decomposition by itself transposed always gives a positive-semidefinite Σ), but our sKL-optimizing AEs only learn to reconstruct the centroid of all training data. Interestingly, however, ordinary ℓ_2-optimizing AEs benefit from the modified input representation. Using the same architectures as in Sect. 5.2 and a similar preprocessing (we separately compress mean vectors and Cholesky decompositions with PCA to 99.9% variance, then scale to unit mean standard deviation), we obtain moderate speedups of up to 16x.

PCA on the same representation performs poorly.

HDML learns codes from triplets of items (x, x^+, x^-), see Sect. 4.6. We select x^+ among the k^+ nearest neighbors of x, and x^- outside the 500 nearest neighbors. During training, we gradually increase k^+ from 10 to 200. Instead of training a randomly initialized network as in [13], we fine-tune the existing AEs. We obtain good results with 128-bit codes, but longer codes do not improve the speedup. To close the gap between ρ_t and ρ_s, we instead employ multiple 128-bit codes handled as in mp-LSH, obtaining an up to 65 fold speedup over the baseline.

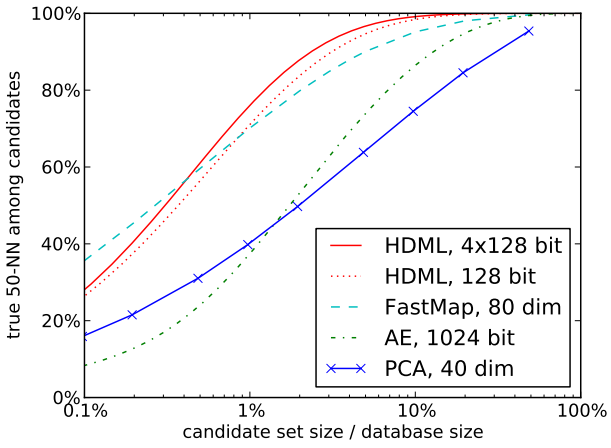


Figure 2. 50-NN recall versus candidate set size for the Gaussian-based music similarity measure on the test set of 253,347 songs.

Method		1-NN $\rho_t(\%)$	1-NN $\rho_s(\%)$	1-NN spu	50-NN $\rho_t(\%)$	50-NN $\rho_s(\%)$	50-NN spu
HDML	128 bit	0.11	1.19	69x	0.11	3.14	28x
	2x128 bit	0.23	0.93	**78x**	0.23	2.61	32x
	4x128 bit	0.45	0.88	67x	0.45	2.08	**36x**
FastMap	40 dim	1.56	0.94	36x	2.17	2.27	20x
	80 dim	2.53	0.85	27x	3.14	2.19	17x
	128 dim	3.67	0.69	21x	4.20	1.86	15x

Table 3. Results for the Gaussian-based music similarity measure on the test set of 1.1 million songs.

FastMap is faster than AE, but slower than HDML. Results fall a bit behind [19] because unlike Schnitzer et al., we evaluate against nearest neighbors found with DSN.

Again, Fig. 2 demonstrates that our conclusions also hold for the test set and a wide range of target recall values.

5.4 Scalability

Finally, we evaluate how the best-performing approaches scale with the collection size. For the large test set of 1.1 million songs, it is computationally infeasible to compute the exact DSN, and we do not want to evaluate an approximate retrieval algorithm against approximate ground truth. Thus, we will limit ourselves to the Gaussian-based measure, omitting the DSN altogether (as in [19]).

From Table 3, we find that the results scale better than linearly, because all methods need smaller candidate sets. For FastMap, the improvement is partly explained by evaluating against non-DSN neighbors: On the validation set, this alone improves the speedup by about 60% (it does not improve HDML, which seemingly learned the DSN well).

Still extrapolating linearly from the validation set to 30 million songs, the best methods are expected to answer 4.2 1-NN queries or 1.3 50-NN queries per second on the vector-based measure, and 2.2 1-NN queries or 0.9 50-NN

queries per second on the Gaussian-based one, using a single CPU core, with 90% true nearest neighbor recall. [7]

6. DISCUSSION

We have shown how to learn binary codes for song representations of two state-of-the-art music similarity measures that are useful for fast k-NN retrieval. Furthermore, we have demonstrated that for collection sizes encountered in MIR, a k-NN index based on a linear scan can outperform sublinear-time methods when we require a particular accuracy – even more so as scan-based methods are *embarrassingly parallel* and can be easily distributed or performed on a GPU. Note that our experiments explicitly targetted commercial-scale collections and song-level search. For user collections, PCA or FastMap will be preferable as they can quickly adapt to any dataset; training AE and HDML took 20 and 180 minutes, respectively. For similarity search on a finer scale (e.g., 10-second snippets), collections could grow to require sublinear-time methods.

For future work, it may be worthwhile to evaluate the binary representation in different scenarios: Do we obtain *qualitatively* good results using the codes alone, omitting the refine step? Do the codes prove useful for classification or clustering?

7. ACKNOWLEDGEMENTS

The author would like to thank Mohammad Norouzi for publishing his HDML implementation, and Maarten Grachten for fruitful discussions.

This research is supported by the Austrian Science Fund (FWF): TRP 307-N23. The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Transport, Innovation, and Technology.

8. REFERENCES

[1] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975.

[2] R. Cai, C. Zhang, L. Zhang, and W.-Y. Ma. Scalable music recommendation by search. In *Proc. of the 15th Int. Conf. on Multimedia (ACM-MM)*, 2007.

[3] E. Chávez, K. Figueroa, and G. Navarro. Proximity searching in high dimensional spaces with a proximity preserving order. In *Proc. of the 4th Mexican Int. Conf. on Artificial Intelligence (MICAI)*, 2005.

[4] C. Faloutsos and K.I. Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, 1995.

[5] A. Flexer, D. Schnitzer, and J. Schlüter. A mirex meta-analysis of hubness in audio music similarity. In *Proc. of the 13th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, Porto, Portugal, 2012.

[7] This assumes all models are held in main memory, which can be practically achieved by distributing queries over a cluster. Preliminary experiments also indicate that the vector-based models can be compressed to 10% of their size with a minor impact on accuracy (also cf. [21]), removing a possible memory bandwidth bottleneck for the refine step.

[6] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. of the 25th Int. Conf. on Very Large Data Bases*, 1999.

[7] Yunchao Gong and Svetlana Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[8] A. Krizhevsky and G. Hinton. Using very deep autoencoders for content-based image retrieval. In *Proc. of the 19th Europ. Symp. on Artificial Neural Networks (ESANN)*, 2011.

[9] T. Liu, A. W. Moore, A. Gray, and K. Yang. An investigation of practical approximate nearest neighbor algorithms. In *Neural Information Processing Systems 17 (NIPS)*. 2005.

[10] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Multi-probe LSH: Efficient indexing for high-dimensional similarity search. In *Proc. of the 33rd Int. Conf. on Very Large Data Bases*, 2007.

[11] M. Mandel and D. Ellis. Song-level features and support vector machines for music classification. In *Proc. of the 6th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, pages 594–599, 2005.

[12] B. McFee and G. Lanckriet. Large-scale music similarity search with spatial trees. In *Proc. of the 12th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, 2011.

[13] M. Norouzi, D. Fleet, and R. Salakhutdinov. Hamming distance metric learning. In *Neural Information Processing Systems 24 (NIPS)*. 2012.

[14] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Proc. of the 10th Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, 2009.

[15] D. Rafailidis, A. Nanopoulos, and Y. Manolopoulos. Nonlinear dimensionality reduction for efficient and effective audio similarity searching. *Multimedia Tools Appl.*, 51(3):881–895, 2011.

[16] R. Salakhutdinov and G. Hinton. Semantic hashing. *Int. Journal of Approximative Reasoning*, 50(7), 2009.

[17] D. Schnitzer. Mirage – high-performance music similarity computation and automatic playlist generation. Master's thesis, Vienna Univ. of Technology, 2007.

[18] D. Schnitzer. *Indexing Content-Based Music Similarity Models for Fast Retrieval in Massive Databases*. PhD thesis, Johannes Kepler Univ. Linz, Austria, 2011.

[19] D. Schnitzer, A. Flexer, and G. Widmer. A filter-and-refine indexing method for fast similarity search in millions of music tracks. In *Proc. of the 10th Int. Soc. of Music Information Retrieval Conf. (ISMIR)*, 2009.

[20] K. Seyerlehner, G. Widmer, and T. Pohle. Fusing block-level features for music similarity estimation. In *Proc. of the 13th Int. Conf. on Digital Audio Effects (DAFx)*, 2010.

[21] K. Seyerlehner, G. Widmer, M. Schedl, and P. Knees. Automatic music tag classification based on block-level features. In *Proc. of the 7th Sound and Music Computing Conf. (SMC)*, Barcelona, Spain, 2010.

[22] M. Slaney, Y. Lifshits, and J. He. Optimal parameters for locality-sensitive hashing. *Proceedings of the IEEE*, 100(9), 2012.

[23] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008.

Oral Session 9:

Structure and Form

FREISCHÜTZ DIGITAL: A CASE STUDY FOR REFERENCE-BASED AUDIO SEGMENTATION OF OPERAS

Thomas Prätzlich
International Audio Laboratories Erlangen
thomas.praetzlich@audiolabs-erlangen.de

Meinard Müller
International Audio Laboratories Erlangen
meinard.mueller@audiolabs-erlangen.de

ABSTRACT

Music information retrieval has started to become more and more important in the humanities by providing tools for computer-assisted processing and analysis of music data. However, when applied to real-world scenarios, even established techniques, which are often developed and tested under lab conditions, reach their limits. In this paper, we illustrate some of these challenges by presenting a study on automated audio segmentation in the context of the interdisciplinary project "Freischütz Digital". One basic task arising in this project is to automatically segment different recordings of the opera "Der Freischütz" according to a reference segmentation specified by a domain expert (musicologist). As it turns out, the task is more complex as one may think at first glance due to significant acoustic and structural variations across the various recordings. As our main contribution, we reveal and discuss these variations by systematically adapting segmentation procedures based on synchronization and matching techniques.

1. INTRODUCTION

In recent years, the availability of digital music material has increased drastically including data of various formats and modalities such as textual, symbolic, acoustic and visual representations. In the case of an opera there typically exist digitized versions of the libretto, different editions of the musical score, as well as a large number of performances given as audio and video recordings, which in its totality constitute the body of sources of a musical work. The goal of the ongoing project "Freischütz Digital" [1] is to develop and apply automated methods to support musicologists in editing, analyzing and comparing the various musical sources. The opera "Der Freischütz" by Carl Maria von Weber is a work of central musical importance offering a rich body of sources. Working out and understanding the variations and inconsistencies within and across the different sources constitutes a major challenge tackled in this project. Another more general objective is to apply and to adjust computer-based methods to real-world scenarios and to the needs of domain experts.

One particular problem arising in this case study concerns the automated segmentation of all available audio recordings of the opera. The opera "Der Freischütz" is a *number opera* in the style of a *Singspiel*, starting with an overture followed by 16 numbers (arias, duets, trios, instrumental pieces, etc.) which are interspersed by spoken text (dialogues). In our scenario, the musicologists are interested in a specific segmentation of the opera, which we refer to as the *reference segmentation*. The audio segmentation task is aimed at automatically transferring this reference segmentation onto all available recordings of the opera, see Figure 1 for illustration.

A related scenario is described in [6], where the goal is to identify unknown audio recordings. By applying automated matching procedures, the unknown recordings are compared to well-annotated audio material in a database. Upon identification, the matching result also allows for segmenting the unknown recording. However, this segmentation is more a byproduct, which is not evaluated in detail. In our scenario, the focus lies on the segmentation and, in a certain sense, we follow a reversed approach as we start from known material that we match to a database which we assume to contain representatives of the same musical work.

The contributions of this paper are twofold. First, we apply and adjust existing synchronization and matching procedures to realize an automated reference-based segmentation procedure. The second and even more important goal of this paper is to highlight the various challenges arising in the context of this seemingly easy segmentation scenario. In fact, the various audio recordings reveal significant acoustic and structural deviations. Considering digitized material from old sound carriers (shellac, LP, tape recordings etc.), one often has to deal with artifacts. Structurally, there are omissions or changes of numbers, repetitions, verses and dialogues. By systematically adjusting the segmentation procedure to reveal these variations, we not only successively improve the segmentation quality, but also gain insights into and a better understanding of the audio material.

The remainder of this paper is organized as follows. In Section 2, we describe the various types of sources that naturally exist in the opera scenario and describe the dataset in more detail. In Section 3, we review some basic music synchronization and audio matching procedures. Then, in

[1] http://freischuetz-digital.de/

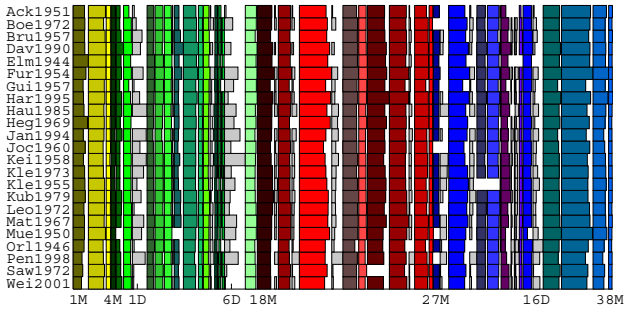


Figure 1. Segmentation result for 23 different audio recordings of "Der Freischütz" according to a reference segmentation specified by musicologists. The reference segmentation includes 38 musical sections (Overture: yellow, Act I: green, Act II: red, Act III: blue) as well as 16 spoken dialogue sections (gray).

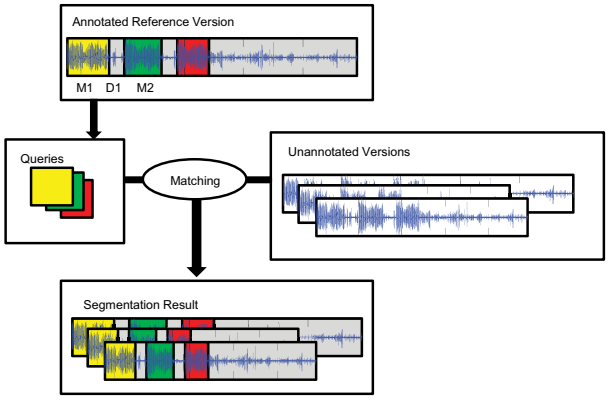


Figure 2. Illustration of the reference-based segmentation procedure.

Section 4, we introduce various segmentation procedures and present a musically informed evaluation of the various results. In Section 5, we conclude the paper and give an outlook to future work. Related work is discussed in the respective sections.

2. MUSICAL BACKGROUND

Music in itself is complex and manifested in many different formats and modalities [5, 9]. For example, for "Der Freischütz" by Carl-Maria von Weber, there are *textual* representations in form of the libretto (text of the opera), *symbolic* representations (musical score), *acoustic* representations (audio recordings) and *visual* representations (video recordings). In the following, "Der Freischütz" – an important representative of the German romantic opera [11] – serves as a challenging case study. The opera is structured in three acts which are further subdivided into an overture and 16 following numbers interspersed by spoken text passages (dialogues). The numbers cover a wide range of musical material (arias, duets, trios, instrumental pieces, etc.). Some of the melodic and harmonic material of the numbers is already introduced in the overture. Also, some of the numbers contain repetitions of musical parts or verses of songs. In the acoustic domain, these are not always part of the performance, as a the conductor or producer may take the artistic freedom to deviate substantially from what is specified in the musical score. Besides differences in the number of played repetitions, further deviations include omissions of other parts or entire numbers as well as variations in the spoken text and the length of the dialogues. Apart from such structural deviations, audio recordings of the opera usually differ in overall length, sound quality, language and many other aspects. For example, our dataset includes historic recordings that are often prone to noise, artifacts, or tuning problems resulting from the digitization process. Furthermore, the recordings show a high variability in their duration, which can be explained by significant tempo differences and also by omissions of material, see Table 1 and Table 2 for details. Also, there are versions which were adapted into French, Italian and Russian language.

Our raw audio data mostly originates from CD recordings, which were initially segmented in CD tracks, see Table 1. These track segmentations are not consistent, varying between 17 and 41 tracks per recording. In some recordings, each number of the opera was put into a separate track, whereas in others the numbers were divided into music and dialogue tracks, and sometimes the remaining music tracks were even subdivided. In order to compare semantically corresponding parts in different versions of the opera, a consistent segmentation is needed. In the context of the "Freischütz Digital" project, such a segmentation is a fundamental requirement for further analysis and processing steps such as the computation of linking structures across different musical sources, including sheet music and audio material.

In our scenario, a reference segmentation of the musical score into musically meaningful sections was specified by a domain expert (musicologist), who divided the opera into 38 musical segments and 16 dialogue segments. According to this reference segmentation, we manually created an annotation for each of the 23 audio recordings in our database, resulting in over 1000 audio segment, see Figure 1 for an overview. The objective of this paper is to recover this annotation using automated methods and to get a better understanding of the variations and inconsistencies in the audio material.

3. SYNCHRONIZATION AND MATCHING TECHNIQUES

As discussed before, the basic task is to segment an unknown audio recording (assuming no pre-segmentation) according to a given reference segmentation. In the following, we assume that this reference segmentation is specified on the basis of a reference audio recording. Then the objective of the segmentation task is to transfer the segmentation from the reference version to the unknown recording. In this section, we introduce some mathematical notions to model our segmentation problem and then review some standard audio synchronization and matching techniques that are applied in the subsequent section.

Let $X := (x_1, x_2, \ldots, x_N)$ be a suitable feature representation of a given audio recording (the feature type is specified later). Then, a *segment* α is a subset $\alpha = [s:t] \subseteq$

$[1:N] := \{1, 2. \ldots, N\}$ with $s \leq t$. Let $|\alpha| := t - s + 1$ denote the length of α. Furthermore, we define a (partial) *segmentation* of X to be a sequence $(\alpha_1, \ldots, \alpha_I)$ of pairwise disjoint segments, i. e. $\alpha_i \cap \alpha_j = \emptyset$ for $i, j \in [1:I]$, $i \neq j$. Note that in this definition we do not assume that $[1:N]$ is completely covered by the segmentation.

In our scenario we assume that we have a reference sequence X with a reference segmentation $\mathcal{A} = (\alpha_1, \ldots, \alpha_I)$. Furthermore, let $Y := (y_1, y_2, \ldots, y_M)$ be a feature representation of an unknown audio recording. In the case that X and Y are structurally similar on a global scale, the transfer of the reference segmentation of X onto Y can be done by using standard synchronization or alignment techniques [1,3,7]. Here, *music synchronization* denotes a procedure which, for a given position in one representation of a piece of music, determines the corresponding position within another representation. When synchronizing two audio recordings, the first step consists in transforming the recordings into feature representations, typically chroma-based audio features. [2] Based on these feature representations and a suitable cost measure, one applies dynamic time warping (DTW) to compute a cost minimizing warping path which realizes the linking between X and Y, see [7, Chapter 4].

This synchronization-based transfer works as long as X and Y globally coincide. However, problems arise in the presence of significant structural differences. Furthermore, in case X and Y are long (as is the case for complete recordings of entire operas), running time and memory issues arise when performing DTW. Even though (multiscale, forward estimation) acceleration techniques exist [1,10], such techniques are not suited when structural differences occur. As an alternative, one may apply more locally oriented *audio matching* techniques, where the individual segments α_i of the reference segmentation (used as "queries") are matched to subsegments of the unknown sequence Y (resulting in "matches" or "hits"), see [4]. In other words, the cost-intensive global DTW alignment is replaced by several smaller local alignments (realized by a subsequence variant of DTW), see also Figure 2 for illustration. Another positive effect is that using local matches allows for a better handling of missing segments and structural differences. On the downside, by querying the reference segments individually, one may loose temporal coherence, while the chance of obtaining local mismatches is increased (in particular for short segments).

In the subsequent section, we systematically apply, modify and combine both techniques – global synchronization and local matching – for performing our segmentation task. Here, besides the actual segmentation, our main goal is to obtain a better understanding of various kinds of variations and inconsistencies in the audio material.

4. AUDIO SEGMENTATION

In this section, after introducing our evaluation measure to assess the accuracy of segmentation results (Section 4.1), we discuss various strategies to tackle the segmentation task based on global synchronization (Section 4.2) and local matching procedures (Section 4.3 – 4.6). Furthermore, we discuss the benefits and limitations of the respective procedures while revealing the musical and acoustic variations and inconsistencies in the audio material.

4.1 Evaluation Measure

First of all, we need a measure that allows us to compare two given segments α and β. To this end, we define the *relative overlap measure* between α and β to be the value

$$\mu(\alpha, \beta) := \frac{|\alpha \cap \beta|}{|\alpha \cup \beta|} \in [0, 1],$$

which indicates the ratio of the absolute overlap and the length of the union segment. Note that $\mu(\alpha, \beta) = 1$ if and only if $\alpha = \beta$, and $\mu(\alpha, \beta) = 0$ if $\alpha \cap \beta = \emptyset$.

As before, let us assume that the reference version is represented by the sequence $X := (x_1, x_2, \ldots, x_N)$ and the *reference segmentation* by $\mathcal{A} := (\alpha_1, \ldots, \alpha_I)$. Furthermore, let $Y := (y_1, y_2, \ldots, y_M)$ be the unknown version to be segmented. For the purpose of evaluation, we assume that there is also a *ground truth segmentation* $\mathcal{B} := (\beta_1, \ldots, \beta_I)$ for Y, where each β_i musically corresponds to the α_i. The goal is to automatically derive the segmentation of Y. Let P denote such a segmentation procedure, which automatically transfers each reference segment α_i to a computed segment $\mathrm{P}(\alpha_i) \subseteq [1:M]$. Then, the relative overlap measure $\mu(\beta_i, \mathrm{P}(\alpha_i))$ indicates the segmentation quality of the procedure P.

Because of the mentioned structural variations, the version Y does not necessarily contain a segment that musically corresponds to a reference segment α_i. In this case, the ground truth segment is set to $\beta_i = \emptyset$. Furthermore, the procedure P does not have to output a computed segment, which is modeled by setting $\mathrm{P}(\alpha_i) = \emptyset$. In the case that both the segment $\mathrm{P}(\alpha_i)$ and β_i are empty, we define $\mu(\beta_i, \mathrm{P}(\alpha_i)) = 1$ (a non-existing segment has been identified as such). Note that if only one of the segments is empty, $\mu(\beta_i, \mathrm{P}(\alpha_i)) = 0$.

4.2 Global Approach ($\mathrm{S1}, \mathrm{S2}$)

In the following matching procedures and evaluation, we only consider the musical sections (indicated by the non-gray segments in Figure 1) while leaving the dialogue sections (the gray segments in Figure 1) unconsidered. Exemplarily, we use a reference segmentation $\mathcal{A} = (\alpha_1, \alpha_2, \ldots, \alpha_{38})$ based on the recording conducted by Carlos Kleiber in 1973 (`Kle1973`), which is a performance that closely follows the musical score. Quantitative results for all procedures to be discussed are presented in Table 1 (relative overlap averaged over versions) and Table 2 (relative overlap averaged over segments).

In the two procedures S1 and S2, we apply a global synchronization approach. For S1, we employ DTW using the step size condition $\Sigma_1 = \{(1,1), (1,2), (2,1)\}$, see [7, Chapter 4]. This strategy is usually very robust as long as there are no significant deviations in structure

[2] In our experiments, we use chroma-based CENS features of 2 Hz resolution as supplied by the chroma toolbox [8].

and tempo between the two versions compared. However, the procedure S1 is not able to compensate well for structural variations leading to an average relative overlap of 0.852, see Table 1 When using the step size condition $\Sigma_2 = \{(1,1),(1,0),(0,1)\}$ (calling this procedure S2), performance improves significantly, yielding the average relative overlap of 0.930, see Table 1. For example, in the version `Saw1972`, the dialogues are comparatively short, see also the gray rectangles in Figure 1. Such a situation causes S1 to fail, resulting in an overlap of 0.615 compared to 0.896 for S2, see Table 1. For both procedures, the alignment accuracy for α_{38} is very low with 0.714 (S1) and 0.724 (S2), see Table 2. This is due to audio material not belonging to the actual opera that is appended at the end (CD bonus tracks) in some versions. In this case, the global synchronization procedures do not allow to skip the final tracks. Despite the promising results of S2, this approach has several limitations. First, it is inefficient considering runtime and memory requirements, especially when increasing the feature resolution, see also Section 3. Secondly, it is not well suited to accommodate for structural changes in a controlled manner. And thirdly, the procedure does not give deeper insights into the musical and acoustic properties of the underlying audio material.

Our goal in the following sections is to develop a more flexible segmentation strategy that achieves a quality comparable to S2 while yielding better insight into the versions' properties.

4.3 Local Approach (M1)

The remaining approaches discussed below rely on a local matching procedure based on a subsequence variant of DTW using the step size condition Σ_1. Here, for each $\alpha_i \in \mathcal{A}$ (used as a query) applied to a given unknown version, we compute a ranked list of *matching candidates*. For the segmentation procedure M1, we only consider the top match in the list, see also Figure 2 for illustration of the general matching strategy.

In Figure 3a, the relative overlap values for M1 computed on all recordings in our dataset are presented in a gray-scale matrix visualization, where the rows indicate the audio versions and the columns indicate the segments. Black corresponds to $\mu = 0$ (no overlap) and white to $\mu = 1$ (perfect overlap). Row-wise, the segmentation accuracy of a specific version becomes obvious, whereas column-wise, segments which are problematic across versions can easily be spotted. An example for a problematic version is `Elm1944`, which generally seems to perform poorly, showing many black entries in Figure 3a and having a low average relative overlap of 0.705, see Table 1. A closer look at the audio material revealed that there are some issues concerning the tuning of this version, probably resulting from the digitization process. Furthermore, there are segments which show a poor segmentation accuracy across versions, see for example the black entries for α_{14} to α_{16} in Figure 3a. It turns out that these three segments correspond to the three verses of a song (No. 4) in the opera. The reason why this song has been divided into individual segments is that there are dialogues between the verses (recall that a requirement of the reference segmentation was to separate music and dialogue sections). The verses all share the same melodic and harmonic material and are thus easily confused with each other in the matching procedure. Another interesting problem appears for α_{32}, where M1 nearly fails for every version, resulting in an overall segmentation accuracy of 0.157, see Table 2 and Figure 3a. Actually, α_{32} (having a duration of only 12.4 seconds) is a short snippet of a chorus section for which many repetitions exist in the surrounding segments α_{31} (song with several verses and chorus sections) and α_{33} (chorus) which are interspersed by dialogues. Thus it is very likely that α_{32} is matched into the harmonically similar parts within α_{31} or α_{33}. For the version `Kle1955`, segment α_{38} seems to be problematic, see Figure 3a. Actually, α_{38} contains musical material which is already used in the overture of the opera (covered by α_3). A closer look into the matching results for `Kle1955` revealed that α_{38} matched indeed into the musically very similar section in the overture.

In conclusion, procedure M1 is more efficient[3], see also Section 3, while its main drawback is the loss of robustness due to confusion of local matches.

4.4 Tuning Issues (M2)

In real world scenarios, the tuning of a music performance often slightly deviates from the standard tuning, where a chamber tone of 440 Hz serves as reference frequency. This usually influences pitch related audio features such as chroma features. To compensate for different tunings, one typically integrates a tuning estimation procedure in the feature extraction process [2]. In the previous approaches, we already used tuned chroma features. But since an unkown version of the opera also contains a lot of non-music material (dialogues, applause, etc.), which is also considered in the tuning estimation, the resulting estimate may be incorrect.

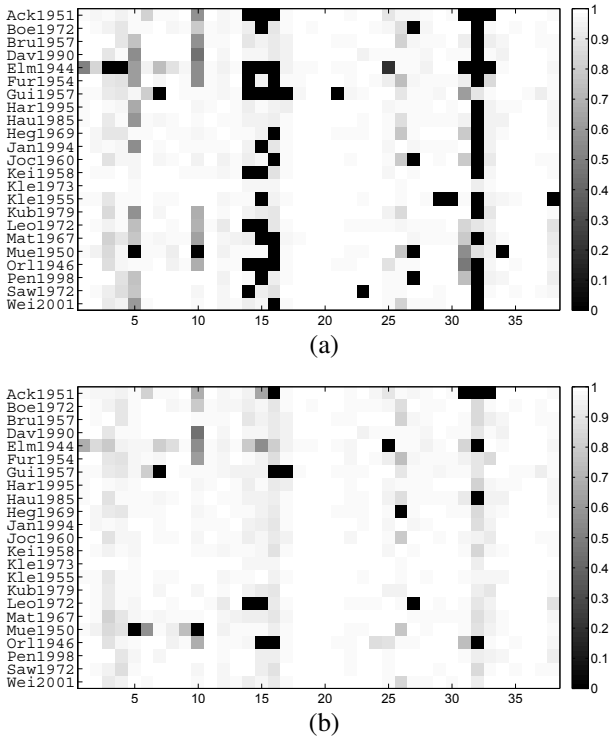
With procedure M2, we evaluate the influence of the tuning estimation on the matching procedure. This problem can either be addressed on the side of the unknown version or on the query side. In our approach, we use the same chroma sequence for the unknown version as in M1, and simulate the tuning deviations on the query side by computing the chroma sequence for the query with respect to six different reference frequencies (in the range of a semitone). Doing this for each query α_i, we then use the chroma sequence yielding the minimum cost in the matching.

For `Elm1944`, the local tuning adjustment indeed leads to a substantial improvement from 0.705 (M1) to 0.777 (M2), see Table 1. Also, there are improvements for certain segments, e.g., α_{38} with 0.921 (M1) compared to 0.968 (M2), see Table 2. In this example, the improvement

[3] On a 64bit machine, the average memory requirement for a global DTW run on one piece of our dataset is 1.7 GB (2 Hz feature resolution) and 42.6 GB (10 Hz), computed from the length of the reference version and the average version length. Upper bounds for the local matching approaches (derived from the maximum query length and the average version length) are 114 MB (2 Hz) and 2.9 GB (10 Hz).

Version	#O	*dur.*	S1	S2	M1	M2	M3	M4
Ack1951	19	6904.81	0.596	0.811	0.808	0.851	0.850	0.853
Boe1972	30	7771.77	0.784	0.931	0.889	0.865	0.962	0.962
Bru1957	24	7439.33	0.906	0.933	0.927	0.905	0.923	0.966
Dav1990	30	8197.88	0.972	0.984	0.926	0.926	0.950	0.961
Elm1944	19	7081.52	0.698	0.827	**0.705**	**0.777**	0.806	0.865
Fur1954	34	9121.69	0.923	0.936	0.866	0.861	0.938	0.949
Gui1957	18	6911.30	0.908	0.937	0.801	0.851	0.860	0.886
Har1995	17	8044.99	0.974	0.981	0.944	0.943	0.965	0.973
Hau1985	17	8245.23	0.955	0.957	0.935	0.933	0.932	0.943
Heg1969	25	7436.75	0.896	0.958	0.913	0.895	0.943	0.946
Jan1994	30	7843.21	0.881	0.987	0.916	0.917	0.964	0.976
Joc1960	26	7178.21	0.922	0.948	0.887	0.911	0.968	0.967
Kei1958	32	8043.00	0.886	0.965	0.904	0.902	0.976	0.975
Kle1973	29	7763.00	1.000	0.996	0.989	0.990	0.990	0.990
Kle1955	41	7459.35	0.776	0.873	0.849	0.876	0.980	0.980
Kub1979	23	8044.65	0.959	0.985	0.927	0.929	0.953	0.974
Leo1972	19	7726.17	0.861	0.926	0.905	0.900	0.875	0.896
Mat1967	17	8309.35	0.984	0.983	0.874	0.876	0.948	0.965
Mue1950	35	7559.97	0.814	0.881	0.825	0.824	0.885	0.895
Orl1946	32	7368.58	0.559	0.807	0.853	0.854	0.852	0.883
Pen1998	26	7768.00	0.866	0.904	0.890	0.891	0.968	0.977
Saw1972	29	6871.02	**0.615**	**0.896**	0.893	0.894	0.968	0.974
Wei2001	38	7220.13	0.859	0.974	0.915	0.916	0.965	0.975
∅	26	7665.65	**0.852**	**0.930**	0.884	0.891	0.931	0.945

Table 1. Relative overlap values averaged over segments for different versions and different procedures. The first column indicates the version, the second (#O) the number of segments on the original sound carrier, and the third column (*dur.*) the overall duration in seconds of the recording. S1, S2, M1, M2 M3, and M4 denote the respective segmentation procedures.



(a)

(b)

Figure 3. Matrix visualization of relative overlap values, where the versions correspond to rows and the segments to columns. **(a)**: P = M1. **(b)**: P = M4.

mainly comes from the version Kle1955, where α_{38} is now matched onto the correct position.

4.5 Global Constraints (M3)

As mentioned in Section 4.3, the local matching procedure can easily confuse musically similar parts. Also, the computed segments obtained by individual matches may not be disjoint. In the procedure M3, we impose additional global constraints on the overall segmentation to cope with these two problems.

α_i	No.	*occ.*	*dur.*	S1	S2	M1	M2	M3	M4
1	0	23	216.5	0.995	0.994	0.968	0.975	0.975	0.975
2	0	23	283.3	0.996	0.995	0.977	0.976	0.976	0.976
3	0	23	081.4	0.962	0.972	0.881	0.918	0.918	0.918
4	1	23	069.9	0.888	0.927	0.900	0.937	0.937	0.937
5	1	22	070.9	0.808	0.938	0.753	0.747	**0.747**	**0.930**
6	1	23	138.4	0.826	0.986	0.969	0.970	0.952	0.952
7	2	23	122.9	0.854	0.983	0.930	0.932	0.932	0.932
8	2	23	152.4	0.959	0.992	0.977	0.977	0.977	0.977
9	2	23	139.8	0.987	0.987	0.986	0.988	0.970	0.977
10	3	22	073.1	0.930	0.945	0.775	0.772	**0.772**	**0.842**
11	3	23	230.3	0.989	0.992	0.985	0.985	0.985	0.985
12	3	23	074.6	0.990	0.993	0.964	0.967	0.967	0.967
13	3	23	092.1	0.939	0.982	0.979	0.976	0.976	0.976
14	4	23	034.6	0.749	0.876	0.617	**0.735**	**0.904**	0.904
15	4	23	029.3	0.635	0.798	0.496	**0.524**	**0.838**	0.838
16	4	20	026.4	0.550	0.692	0.519	**0.479**	**0.789**	0.789
17	5	23	186.0	0.979	0.985	0.930	0.930	0.930	0.930
18	6	23	287.8	0.984	0.994	0.987	0.989	0.989	0.989
19	7	23	223.9	0.963	0.972	0.992	0.992	0.992	0.992
20	8	23	499.4	0.989	0.997	0.995	0.994	0.994	0.994
21	9	23	258.6	0.979	0.992	0.945	0.988	0.988	0.988
22	9	23	137.6	0.971	0.978	0.985	0.980	0.980	0.980
23	10	22	337.3	0.944	0.951	0.944	0.943	0.987	0.987
24	10	23	301.9	0.977	0.986	0.989	0.988	0.981	0.981
25	10	23	243.8	0.910	0.986	0.933	0.932	0.924	0.924
26	10	23	059.7	0.740	0.889	0.908	0.847	0.883	0.883
27	11	19	104.5	0.631	0.725	0.807	0.807	**0.938**	0.938
28	12	23	356.9	0.882	0.988	0.982	0.982	0.982	0.982
29	13	22	161.5	0.794	0.940	0.943	0.943	0.986	0.986
30	13	22	208.8	0.814	0.951	0.945	0.944	0.984	0.987
31	14	23	168.4	0.729	0.923	0.796	0.790	**0.796**	**0.917**
32	14	19	**012.4**	0.439	0.643	**0.157**	**0.198**	**0.698**	0.735
33	14	22	057.7	0.714	0.846	0.864	0.869	0.827	0.913
34	15	23	147.2	0.745	0.946	0.938	0.937	0.980	0.980
35	16	23	303.6	0.827	0.996	0.990	0.989	0.989	0.989
36	16	23	503.2	0.812	0.965	0.994	0.994	0.994	0.994
37	16	23	241.2	0.781	0.894	0.987	0.987	0.987	0.987
38	16	23	068.2	**0.714**	**0.724**	**0.921**	**0.968**	0.968	0.968
∅		22.5	176.47	**0.852**	**0.930**	0.884	0.891	0.931	0.945

Table 2. Relative overlap values averaged over versions for different segments and different procedures. The first column (α_i) indicates the reference segment, the second column (No.) the musical number within the opera, the third column (*occ.*) the number of occurrences of α_i in the 23 versions of the dataset, and the fourth column (*dur.*) refers to the duration in seconds of α_i. S1, S2, M1, M2 M3, and M4 denote the respective segmentation procedures.

When using α_i as query, we now consider the entire ranked list of matches (instead of only using the top match as in M1 and M2). From each list we choose the best candidate so that the following global constraints are satisfied:

i) *Disjointness condition*: $\mathrm{P}(\alpha_i) \cap \mathrm{P}(\alpha_j) = \emptyset$

ii) *Temporal monotonicity*: $\alpha_i \prec \alpha_j \Rightarrow \mathrm{P}(\alpha_i) \prec \mathrm{P}(\alpha_j)$.

Here, we define the partial order $\prec$ on the set of segments by $\alpha_1 = [s_1 : t_1] \prec \alpha_2 = [s_2 : t_2] :\Leftrightarrow t_1 < s_2$. An optimal selection of matches from the ranked lists satisfying these global constraints can be computed using dynamic programming (similar to DTW). Howevever, note that in this case the dynamic programming is performed on the coarse segment level and not on the much finer frame level as in the case of global synchronization.

Applying this strategy does indeed improve the overall matching accuracy, on a version level as well as for individual segments, see Table 1 and Table 2. For example, for the segments α_{14}/α_{15}/α_{16}, the results improve from 0.735/0.524/0.479 for M2 to 0.904/0.838/0.789 for M3. Also, the results for α_{32} improve from 0.198 (M2) to 0.698 (M3).

Another interesting example is the relative overlap of 0.938 for α_{27}. This segment is actually missing in four

recordings of the opera. Using global constraints, the nonexistence of these segments was correctly identified by procedure $\mathrm{P} = \mathrm{M3}$ resulting in $\mathrm{P}(\alpha_{27}) = \emptyset$. However, the corresponding segment in `Leo1972` was misclassified as nonexistent by M3. A closer inspection revealed that the assumption modeled in the constraint that segments always appear in the same order as in the reference version was violated in this audio version. Here, the musical section covered by α_{27} was placed after α_{30} and used as an introduction before α_{31}. Thus, although strategy M3 stabilizes the overall matching, flexibility concerning the temporal order of segments is lost.

4.6 Structural Issues (M4)

Another problem occurs for the segments α_5, α_{10} and α_{31}, having the relative overlap values of 0.747, 0.772, and 0.796 for M3, respectively. According to the musical score, all these sections include repetitions of some music material. The segment α_5 for example should, according to the musical score, follow the structure $IA_1A_2B_1B_2O$, where I is an introductory and O an "outro" part. However, not all the repetitions are always performed. For example, the alternative structures IA_1B_1O, $IA_1A_2B_1O$, or $IA_1B_1B_2O$ for α_5 all appear in recordings of our dataset (similar variations occur for α_{10} and α_{31}). Such structural deviations can generally not be compensated well in the local matching procedure. Also, for further processing and analysis steps, such as the synchronization between corresponding segments in different recordings, it is important to know the exact structure of a given segment.

For M4, we investigate how structural correspondence of the query with an unknown version influences the segmentation quality. We manually annotated the musical structures occurring for α_5, α_{10} and α_{31} in the different audio versions of the opera. This information is then used in the matching to generate a query which structurally corresponds to the unknown version. The actual matching algorithm is the same as in M3. From the quantitative results in Table 2, we can conclude that the structural variations were indeed the cause of the poor performance for these segments: α_5 improves from 0.747 (M3) to 0.930 (M4), α_{10} from 0.772 (M3) to 0.842 (M4) and α_{31} from 0.796 (M3) to 0.917 (M4), see also Figure 3b.

5. CONCLUSIONS

In this paper, we presented a case study on segmenting given audio versions of an opera into musically meaningful sections that have been specified by a domain expert. Adapting existing synchronization and matching techniques, we discussed various challenges that occur when dealing with real-world scenarios due to the variability of acoustic and musical aspects. Rather than presenting technical details, our main motivation was to show how automated methods may be useful for systematically revealing and understanding the inconsistencies and variations hidden in the audio material. Furthermore, we showed how a procedure based on a combination of local matching and global constraints yields a more flexible and efficient alternative to a global black-box synchronization approach. Besides yielding slightly better results, this alternative procedure also provides a more explicit control to handle the various musical aspects and yields deeper insights into the properties of the audio material. For the future, we plan to expand our segmentation approach by explicitly including the dialogue sections into the analysis. Furthermore, the segmentation results will serve as basis for a finer grained analysis and multimodal processing including informed source separation.

Acknowledgments: This work has been supported by the BMBF project *Freischütz Digital* (Funding Code 01UG1239A to C). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer IIS.

6. REFERENCES

[1] Simon Dixon and Gerhard Widmer. MATCH: A music alignment tool chest. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, GB, 2005.

[2] Emilia Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, UPF Barcelona, 2006.

[3] Ning Hu, Roger B. Dannenberg, and George Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, USA, 2003.

[4] Frank Kurth and Meinard Müller. Efficient index-based audio matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):382–395, 2008.

[5] Cynthia C. S. Liem, Meinard Müller, Douglas Eck, George Tzanetakis, and Alan Hanjalic. The need for music information retrieval with user-centered and multimodal strategies. In *Proceedings of the International ACM Workshop on Music Information Retrieval with User-centered and Multimodal Strategies (MIRUM)*, pages 1–6, 2011.

[6] Nicola Montecchio, Emanuele Di Buccio, and Nicola Orio. An efficient identification methodology for improved access to music heritage collections. *Journal of Multimedia*, 7(2):145–158, 2012.

[7] Meinard Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.

[8] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 215–220, Miami, FL, USA, 2011.

[9] Meinard Müller, Masataka Goto, and Markus Schedl, editors. *Multimodal Music Processing*, volume 3 of *Dagstuhl Follow-Ups*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, 2012.

[10] Meinard Müller, Henning Mattes, and Frank Kurth. An efficient multiscale approach to audio synchronization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 192–197, Victoria, Canada, 2006.

[11] John Warrack. *Carl Maria von Weber*. Cambridge University Press, 1976.

AUTOMATED METHODS FOR ANALYZING MUSIC RECORDINGS IN SONATA FORM

Nanzhu Jiang
International Audio Laboratories Erlangen
nanzhu.jiang@audiolabs-erlangen.de

Meinard Müller
International Audio Laboratories Erlangen
meinard.mueller@audiolabs-erlangen.de

ABSTRACT

The sonata form has been one of the most important large-scale musical structures used since the early Classical period. Typically, the first movements of symphonies and sonatas follow the sonata form, which (in its most basic form) starts with an exposition and a repetition thereof, continues with a development, and closes with a recapitulation. The recapitulation can be regarded as an altered repeat of the exposition, where certain substructures (first and second subject groups) appear in musically modified forms. In this paper, we introduce automated methods for analyzing music recordings in sonata form, where we proceed in two steps. In the first step, we derive the coarse structure by exploiting that the recapitulation is a kind of repetition of the exposition. This requires audio structure analysis tools that are invariant under local modulations. In the second step, we identify finer substructures by capturing relative modulations between the subject groups in exposition and recapitulation. We evaluate and discuss our results by means of the Beethoven piano sonatas. In particular, we introduce a novel visualization that not only indicates the benefits and limitations of our methods, but also yields some interesting musical insights into the data.

1. INTRODUCTION

The musical form refers to the overall structure of a piece of music by its repeating and contrasting parts, which stand in certain relations to each other [5]. For example, many songs follow a strophic form where the same melody is repeated over and over again, thus yielding the musical form $A_1A_2A_3A_4....$[1] Or for a composition written in rondo form, a recurring theme alternates with contrasting sections yielding the musical form $A_1BA_2CA_3D....$ One of the most important musical forms in Western classical music is known as *sonata form*, which consists of an *exposition* (E), a *development* (D), and a *recapitulation* (R), where the exposition is typically repeated once. Sometimes, one can find an additional introduction (I) and a closing coda (C), thus yielding the form IE_1E_2DRC. In particular, the exposition and the recapitulation stand in close relation to each other both containing two subsequent contrasting subject groups (often simply referred to as first and second theme) connected by some transition. However, in the recapitulation, these elements are musically altered compared to their occurrence in the exposition. In particular, the second subject group appears in a modulated form, see [4] for details. The sonata form gives a composition a specific identity and has been widely used for the first movements in symphonies, sonatas, concertos, string quartets, and so on.

In this paper, we introduce automated methods for analyzing and deriving the structure for a given audio recording of a piece of music in sonata form. This task is a specific case of the more general problem known as *audio structure analysis* with the objective to partition a given audio recording into temporal segments and of grouping these segments into musically meaningful categories [2, 10]. Because of different structure principles, the hierarchical nature of structure, and the presence of musical variations, general structure analysis is a difficult and sometimes a rather ill-defined problem [12]. Most of the previous approaches consider the case of popular music, where the task is to identify the intro, chorus, and verse sections of a given song [2, 9–11]. Other approaches focus on subproblems such as audio thumbnailing with the objective to extract only the most repetitive and characteristic segment of a given music recording [1, 3, 8].

In most previous work, the considered structural parts are often assumed to have a duration between 10 and 60 seconds, resulting in some kind of medium-grained analysis. Also, repeating parts are often assumed to be quite similar in tempo and harmony, where only differences in timbre and instrumentation are allowed. Furthermore, *global* modulations can be handled well by cyclic shifts of chroma-based audio features [3]. When dealing with the sonata form, certain aspects become more complex. First, the duration of musical parts are much longer often exceeding two minutes. Even though the recapitulation can be considered as some kind of repetition of the exposition, significant local differences that may last for a couple of seconds or even 20 seconds may exist between these parts. Furthermore, there may be additional or missing sub-structures as well as relative tempo differences be-

[1] To describe a musical from, one often uses the capital letters to refer to musical parts, where repeating parts are denoted by the same letter. The subscripts indicate the order of repeated occurrences.

tween the exposition and recapitulation. Finally, these two parts reveal differences in form of *local* modulations that cannot be handled by a global cyclic chroma shift.

The goal of this paper is to show how structure analysis methods can be adapted to deal with such challenges. In our approach, we proceed in two steps. In the first step, we describe how a recent audio thumbnailing procedure [8] can be applied to identify the exposition and the recapitulation (Section 2). To deal with local modulations, we use the concept of transposition-invariant self-similarity matrices [6]. In the second step, we reveal finer substructures in exposition and recapitulation by capturing relative modulation differences between the first and the second subject groups (Section 3). As for the evaluation of the two steps, we consider the first movements in sonata form of the piano sonatas by Ludwig van Beethoven, which constitutes a challenging and musically outstanding collection of works [13]. Besides some quantitative evaluation, we also contribute with a novel visualization that not only indicates the benefits and limitations of our methods, but also yields some interesting musical insights into the data.

2. COARSE STRUCTURE

In the first step, our goal is to split up a given music recording into segments that correspond to the large-scale musical structure of the sonata form. On this coarse level, we assume that the recapitulation is basically a repetition of the exposition, where the local deviations are to be neglected. Thus, the sonata form IE_1E_2DRC is dominated by the three repeating parts E_1, E_2, and R.

To find the most repetitive segment of a music recording, we apply and adjust the thumbnailing procedure proposed in [8]. To this end, the music recording is first converted into a sequence of chroma-based audio features [2], which relate to harmonic and melodic properties [7]. From this sequence, a suitably enhanced self-similarity matrix (SSM) is derived [8]. In our case, we apply in the SSM calculation a relatively long smoothing filter of 12 seconds, which allows us to better bridge local differences in repeating segments. Furthermore, to deal with local modulations, we use a *transposition-invariant* version of the SSM, see [6]. To compute such a matrix, one compares the chroma feature sequence with cyclically shifted versions of itself, see [3]. For each of the twelve possible chroma shifts, one obtains a similarity matrix. The transposition-invariant matrix is then obtained by taking the entry-wise maximum over the twelve matrices. Furthermore, storing the shift index which yields the maximum similarity for each entry results in another matrix referred to as *transposition index matrix*, which will be used in Section 3. Based on such transposition-invariant SSM, we apply the procedure of [8] to compute for each audio segment a fitness value that expresses how well the given segment explains

[2] In our scenario, we use a chroma variant referred to as `CENS` features, which are part of the Chroma Toolbox `http://www.mpi-inf.mpg.de/resources/MIR/chromatoolbox/`. Using a long smoothing window of four seconds and a coarse feature resolution of 1 Hz, we obtain features that show a high degree of robustness to smaller deviations, see [7] for details.

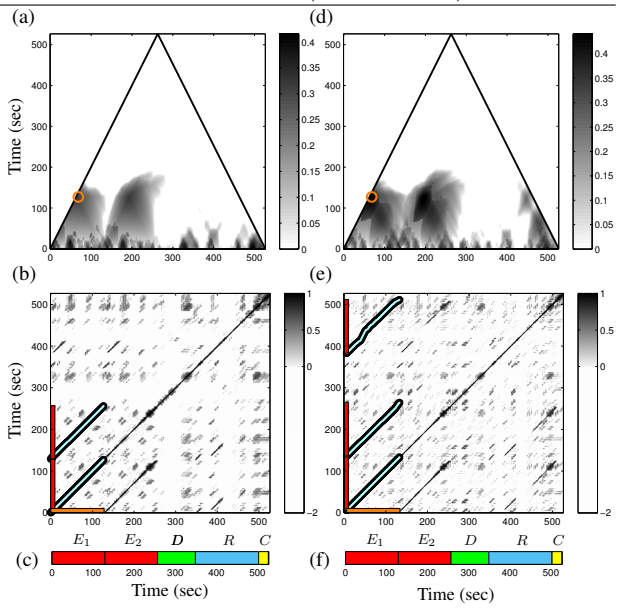


Figure 1: Thumbnailing procedure for `Op031No2-01` ("Tempest"). **(a)/(d)** Scape plot representation using an SSM without/with transposition invariance. **(b)/(e)** SSM without/with transposition invariance along with the optimizing path family (cyan), the thumbnail segment (indicated on horizontal axis) and induced segments (indicated on vertical axis). **(c)/(f)** Ground-truth segmentation.

other related segments (also called induced segments) in the music recording. These relations are expressed by a so-called path family over the given segment. The thumbnail is then defined as the segment that maximizes the fitness. Furthermore, a triangular scape plot representation is computed, which shows the fitness of all segments and yields a compact high-level view on the structural properties of the entire audio recording.

We expect that the thumbnail segment, at least on the coarse level, should correspond to the exposition (E_1), while the induced segments should correspond to the repeating exposition (E_2) and the recapitulation (R). To illustrate this, we consider as our running example a Barenboim recording of the first movement of Beethoven's piano sonata Op. 31, No. 2 ("Tempest"), see Figure 1. In the following, we also use the identifier `Op031No2-01` to refer to this movement. Being in the sonata form, the coarse musical form of this movement is E_1E_2DRC. Even though R is some kind of repetition of E_1, there are significant musical differences. For example, the first subject group in R is modified and extended by an additional section not present in E_1, and the second subject group in R is transposed five semitones upwards (and later transposed seven semitones downwards) relative to the second subject group in E_1. In Figure 1, the scape plot representation (top) and SSM along with the ground truth segmentation (bottom) are shown for our example, where on the left an SSM without and on the right an SSM with transposition invariance has been used. In both cases, the thumbnail segment corresponds to part E_1. However, without using transposition-invariance, the recapitulation is not among the induced segments, thus not representing the complete sonata form, see Figure 1b. In contrast, using transposition-invariance, also the R-segment is identified by the procedure as a repetition

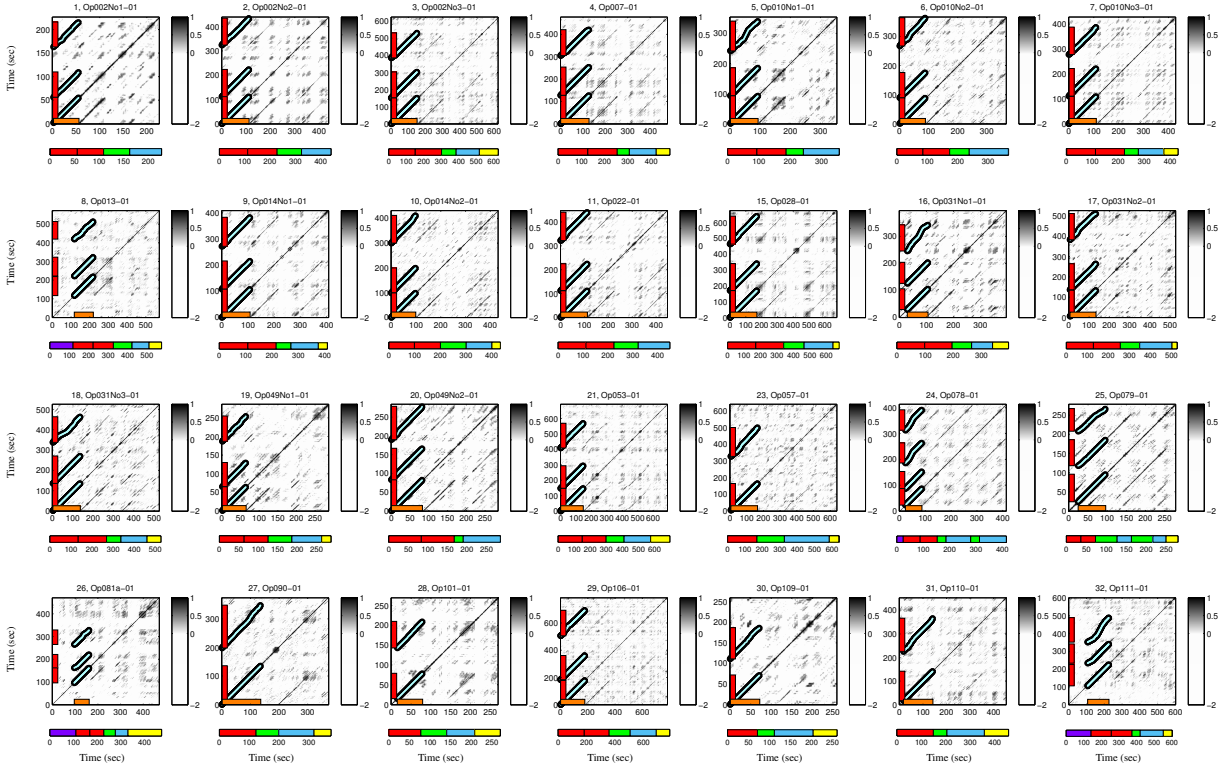


Figure 2: Results of the thumbnailing procedure for the 28 first movements in sonata form. The figure shows for each recording the underlying SSM along with the optimizing path family (cyan), the thumbnail segment (indicated on horizontal axis) and the induced segments (indicated on vertical axis). Furthermore, the corresponding GT segmentation is indicated below each SSM.

of the E_1-segment, see Figure 1e.

At this point, we want to emphasize that only the usage of various smoothing and enhancement strategies in combination with a robust thumbnailing procedure makes it possible to identify the recapitulation. The procedure described in [8] is suitably adjusted by using smoothed chroma features having a low resolution as well as applying a long smoothing length and transposition-invariance in the SSM computation. Additionally, when deriving the thumbnail, we apply a lower bound constraint for the minimal possible segment length of the thumbnail. This lower bound is set to one sixth of the duration of the music recording, where we make the musically informed assumption that the exposition typically covers at least one sixth of the entire movement.

To evaluate our procedure, we use the complete Barenboim recordings of the 32 piano sonatas by Ludwig van Beethoven. Among the first movements, we only consider the 28 movements that are actually composed in sonata form. For each of these recording, we manually annotated the large-scale musical structure also referred to as ground-truth (GT) segmentation, see Table 1 for an overview. Then, using our thumbnailing approach, we computed the thumbnail and the induced segmentation (resulting in two to four segments) for each of the 28 recordings. Using pairwise P/R/F-values [3], we compared the computed segments with the E- and R-segments specified by the GT annotation, see Table 1. As can be seen, one obtains high P/R/F-values for most recordings, thus indicating a good performance of the procedure. This is also reflected by Figure 2, which shows the SSMs along with the path families and ground truth segmentation for all 28 recordings. However, there are also a number of exceptional cases where our procedure seems to fail. For example, for `Op079-01` (No. 25), one obtains an F-measure of only 0.55. Actually, it turns out that for this recording

No.	Piece ID	GT Musical Form	P	R	F
1	Op002No1-01	E_1E_2DR	0.99	0.90	0.90
2	Op002No2-01	E_1E_2DR	0.99	0.96	0.96
3	Op002No3-01	E_1E_2DRC	0.95	0.97	0.97
4	Op007-01	E_1E_2DRC	1.00	0.99	0.99
5	Op010No1-01	E_1E_2DR	0.99	0.93	0.93
6	Op010No2-01	E_1E_2DR	0.95	0.86	0.86
7	Op010No3-01	E_1E_2DRC	0.93	0.94	0.94
8	Op013-01	IE_1E_2DRC	0.96	0.95	0.95
9	Op014No1-01	E_1E_2DRC	0.97	0.97	0.97
10	Op014No2-01	E_1E_2DRC	0.94	0.96	0.96
11	Op022-01	E_1E_2DR	1.00	0.97	0.97
12	Op026-01	-	-	-	-
13	Op027No1-01	-	-	-	-
14	Op027No2-01	-	-	-	-
15	Op028-01	E_1E_2DRC	1.00	0.99	0.99
16	Op031No1-01	E_1E_2DRC	0.83	0.74	0.74
17	Op031No2-01	E_1E_2DRC	0.90	0.85	0.85
18	Op031No3-01	E_1E_2DRC	0.99	0.98	0.98
19	Op049No1-01	E_1E_2DRC	0.96	0.91	0.91
20	Op049No2-01	E_1E_2DR	1.00	0.96	0.96
21	Op053-01	E_1E_2DRC	0.99	0.97	0.97
22	Op054-01	-	-	-	-
23	Op057-01	$EDRC$	0.92	0.78	0.78
24	Op078-01	$IE_1E_2D_1R_1D_2R_2$	0.98	0.84	0.84
25	Op079-01	$E_1E_2D_1R_1D_2R_2C$	0.50	0.55	0.55
26	Op081a-01	IE_1E_2DRC	0.86	0.88	0.88
27	Op090-01	$EDRC$	0.76	0.85	0.85
28	Op101-01	$EDRC$	0.97	0.89	0.89
29	Op106-01	E_1E_2DRC	0.99	0.98	0.98
30	Op109-01	$EDRC$	0.92	0.86	0.86
31	Op110-01	$EDRC$	0.91	0.81	0.81
32	Op111-01	IE_1E_2DRC	0.65	0.64	0.64
	Average		0.92	0.86	0.89

Table 1: Ground truth annotation and evaluation results (pairwise P/R/F values) for the thumbnailing procedure using Barenboim recordings for the first movements in sonata form of the Beethoven piano sonatas.

[3] These values are standard evaluation measures used in audio structure analysis, see, e. g. [10].

the D-part as well as R-part are also repeated resulting in the form $E_1E_2D_1R_1D_2R_2C$. As a result, our minimum length assumption that the exposition covers at least one sixth of the entire movement is violated. However, by reducing the bound to one eighth, one obtains for this recording the correct thumbnail and an F-measure of 0.85. In particular, for the later Beethoven sonatas, the results tend to become poorer compared to the earlier sonatas. From a musical point of view, this is not surprising since the later sonatas are characterized by the release of common rules for musical structures and the increase of compositional complexity [13]. For example, for some of the sonatas, the exposition is no longer repeated, while the coda takes over the role of a part of equal importance.

3. FINE STRUCTURE

In the second step, our goal is to find substructures within the exposition and recapitulation by exploiting the relative harmonic relations that typically exist between these two parts. Generally, the exposition presents the main thematic material of the movement that is contained in two contrasting *subject groups*. Here, in the first subject group ($G1$) the music is in the *tonic* (the home key) of the movement, whereas in the second subject group ($G2$) it is in the *dominant* (for major sonatas) or in the *tonic parallel* (for minor sonatas). Furthermore, the two subject groups are typically combined by a *modulating transition* (T) between them, and at the end of the exposition there is often an additional closing theme or *codetta* (C). The recapitulation contains similar sub-parts as the exposition, however it includes some important harmonic changes. In the following discussion, we denote the four sub-parts in the exposition by E-$G1$, E-T, E-$G2$, and E-C. Also, in the recapitulation by R-$G1$, R-T, R-$G2$, and R-C. The first subject groups E-$G1$ and R-$G1$ are typically repeated in more or less the same way both appearing in the tonic. However, in contrast to E-$G2$ appearing in the dominant or tonic parallel, the second subject group R-$G2$ appears in the tonic. Furthermore, compared to E-T, the transition R-T is often extended, sometimes even presenting new material and local modulations, see [4] for details. Note that the described structure indicates a tendency rather then being a strict rule. Actually, there are many exceptions and modifications as the following examples demonstrate.

To illustrate the harmonic relations between the subject groups, let us assume that the movement is written in C major. Then, in the exposition, E-$G1$ would also be in C major, and E-$G2$ would be in G major. In the recapitulation, however, both R-$G1$ and R-$G2$ would be in C major. Therefore, while E-$G1$ and R-$G1$ are in the same key, R-$G2$ is a modulated version of E-$G2$, shifted five semitones upwards (or seven semitones downwards). In terms of the maximizing shift index as introduced in Section 2, one can expect this index to be $i = 5$ in the transposition index matrix when comparing E-$G2$ with R-$G2$.[4] Similarly, for minor sonatas, this index is typically $i = 9$, which corresponds to shifting three semitones downwards from the tonic parallel to the tonic.

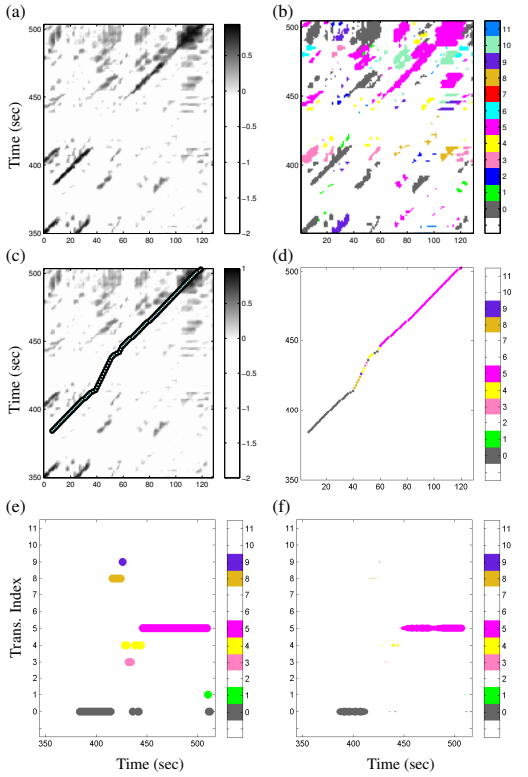


Figure 3: Illustration for deriving the WRTI (weighted relative transposition index) representation using `Op031No2-01` as example. **(a)** Enlarged part of the SSM shown in Figure 1e, where the horizontal axis corresponds to the E_1-segment and the vertical axis to the R-segment. **(b)** Corresponding part of the transposition index matrix. **(c)** Path component of the optimizing path family as shown in Figure 1e. **(d)** Transposition index restricted to the path component. **(e)** Transposition index plotted over time axis of R-segment. **(f)** Final WRTI representation.

Based on this observation, we now describe a procedure for detecting and measuring the relative differences in harmony between the exposition and the recapitulation. To illustrate this procedure, we continue our example `Op031No2-01` from Section 2, where we have already identified the coarse sonata form segmentation, see Figure 1e. Recall that when computing the transposition-invariant SSM, one also obtains the *transposition index matrix*, which indicates the maximizing chroma shift index [6]. Figure 3a shows an enlarged part of the enhanced and thresholded SSM as used in the thumbnailing procedure, where the horizontal axis corresponds to the exposition E_1 and the vertical axis to the recapitulation R. Figure 3b shows the corresponding part of the transposition index matrix, where the chroma shift indices are displayed in a color-coded form.[5] As revealed by Figure 3b, the shift indices corresponding to E-$G1$ and R-$G1$ are zero (gray color), whereas the shift indices corresponding to E-$G2$ and R-$G2$ are five (pink color). To further emphasize these relations, we focus on the path that encodes the sim-

[4] We assume that the index encodes shifts in upwards direction. Note that the shifts are cyclic, so that shifting five semitones upwards is the same as shifting seven semitones downwards.

[5] For the sake of clarity, only those shift indices are shown that correspond to the relevant entries (having a value above zero) of the SSM shown in Figure 3a.

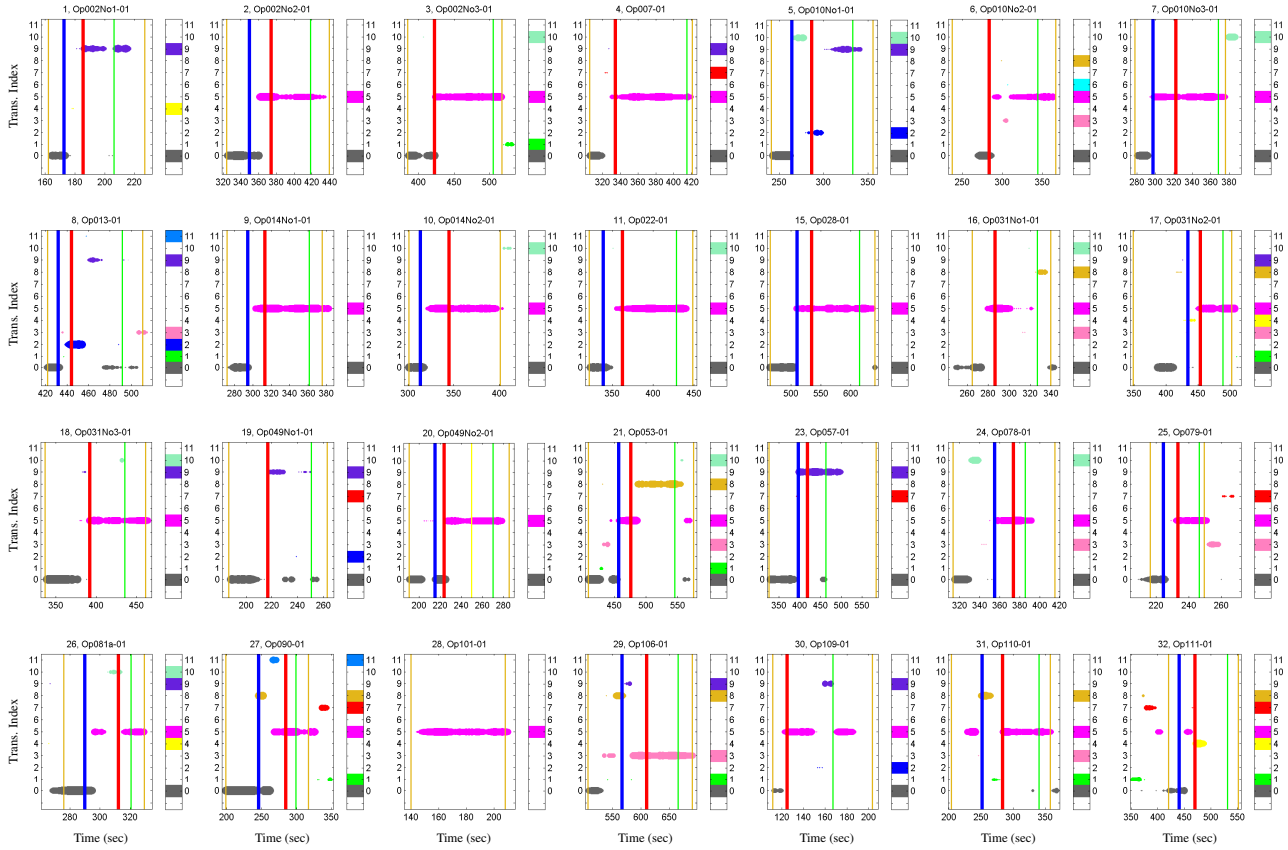


Figure 4: WRTI representations for all 28 recordings. The manual annotations of the segment boundaries between R-$G1$, R-T, R-$G2$, and R-C are indicated by vertical lines. In particular, the blue line indicates the end of R-$G1$ and the red line as the beginning of R-$G2$.

ilarity between E_1 and R, see Figure 3c. This path is a component of the optimizing path family computed in the thumbnailing procedure, see Figure 1e. We then consider only the shift indices that lie on this path, see Figure 3d. Next, we convert the vertical time axis of Figure 3d, which corresponds to the R-segment, into a horizontal time axis. Over this horizontal axis, we plot the corresponding shift index, where the index value determines the position on the vertical index axis, see Figure 3e. In this way, one obtains a function that expresses for each position in the recapitulation the harmonic difference (in terms of chroma shifts) relative to musically corresponding positions in the exposition. We refine this representation by weighting the shift indices according to the SSM values underlying the path component. In the visualization of Figure 3f, these weights are represented by the thickness of the plotted dots. In the following, for short, we refer to this representation as the WRTI (weighted relative transposition index) representation of the recapitulation.

Figure 4 shows the WRTI representations for the 28 recordings discussed in Section 2. Closely following [13], we manually annotated the segments corresponding to $G1$, T, $G2$, and C within the expositions and recapitulations of these recordings [6], see Table 2. In Figure 4, the segment corresponding to R-T is indicated by a blue vertical line (end of R-$G1$) and a red vertical line (beginning of R-$G2$). Note that for some sonatas (e. g., `Op002No3-01` or `Op007-01`) there is no such transition, so that only the red vertical line is visible. For many of the 28 recordings, as the theory suggests, the WRTI representation indeed indicates the location of the transition segment by a switch from the shift index $i = 0$ to the shift index $i = 5$ (for sonatas in major) or to $i = 9$ (for sonatas in minor). For example, for the movement `Op002No1-01` (No. 1) in F minor, the switch from $i = 0$ to $i = 9$ occurs in the transition segment. Or for our running example `Op031No2-01` (No. 17), there is a clearly visible switch from $i = 0$ to $i = 5$ with some further local modulations in between. Actually, this sonata already constitutes an interesting exception, since the shift of the second subject group is from the dominant (exposition) to the tonic (recapitulation) even though the sonata is in minor (D minor). Another more complex example is `Op013-01` (No. 8, "Pathétique") in C minor, where E-$G1$ starts with $E^\flat$ minor, whereas R-$G1$ starts with F minor (shift index $i = 2$) before it reaches the tonic C minor (shift index $i = 9$). Actually, our WRTI representation reveals these harmonic relations.

To obtain a more quantitative evaluation, we located the transition segment R-T by determining the time position (or region) where the shift index $i = 0$ (typically corresponding to R-$G1$) changes to the most prominent non-zero shift index within the R-segment (typically corresponding to R-$G2$ and usually $i = 5$ or $i = 9$), where we neglect all other shift indices. This position (or region) was computed by a simple sweep algorithm to find the optimal position that separates the weighted zero-indices (which should be on the left side of the optimal sweep line) and the weighted indices of the prominent index (which should

[6] As far as this is possible due to many deviations and variations in the actual musical forms.

No.	Piece ID	$G1$	T	$G2$	C	$\Delta(G1)$	In(T)	$\Delta(G2)$
1	Op002No1-01	10.6	12.6	20.8	20.4		y	
2	Op002No2-01	26.0	24.4	44.2	21.1		y	
3	Op002No3-01	37.9	-	82.9	12.3	-0.6	n	
4	Op007-01	29.0	-	80.7	5.7	-11.5	n	
5	Op010No1-01	23.2	22.4	45.9	22.4		y	
6	Op010No2-01	46.2	-	60.3	22.2		n	2.0
7	Op010No3-01	20.1	24.7	46.2	7.5	-5.6	n	
8	Op013-01	10.1	12.1	47.2	18.8		y	
9	Op014No1-01	22.8	18.6	48.4	13.9		y	
10	Op014No2-01	13.0	31.4	55.7	-		y	
11	Op022-01	17.5	23.5	65.7	19.8		y	
12	Op026-01	-	-	-	-	-	-	-
13	Op027No1-01	-	-	-	-	-	-	-
14	Op027No2-01	-	-	-	-	-	-	-
15	Op028-01	45.2	24.7	80.3	25.4	-4.0	n	
16	Op031No1-01	21.6	-	40.2	12.6	-12.5	n	
17	Op031No2-01	85.7	19.6	34.9	13.6	-5.4	n	
18	Op031No3-01	55.4	-	42.9	25.7	-10.3	n	
19	Op049No1-01	30.5	-	33.5	12.5	-6.0	n	
20	Op049No2-01	24.6	8.6	26.2	15.2		n	8.9
21	Op053-01	47.6	19.3	69.2	29.1		y	
22	Op054-01	-	-	-	-	-	-	-
23	Op057-01	70.3	22.7	43.7	120.8	-7.3	n	
24	Op078-01	41.7	18.9	11.7	29.5	-15.9	n	
25	Op079-01	8.0	8.9	13.2	2.9		y	
26	Op081a-01	13.9	22.3	8.3	8.8		y	
27	Op090-01	47.1	38.9	14.1	18.2		y	
28	Op101-01	-	-	-	-	-	-	-
29	Op106-01	60.0	43.4	55.5	24.9	-36.7	n	
30	Op109-01	13.7	-	41.9	36.6	-6.1	n	
31	Op110-01	47.8	32.0	56.0	17.3	-26.0	n	
32	Op111-01	20.3	29.9	61.0	20.4		y	

Table 2: Ground truth annotation and evaluation results for finer-grained structure. The columns indicate the number of the sonata (No.), the identifier, as well as the duration (in seconds) of the annotated segments corresponding to R-$G1$, R-T, R-$G2$, and R-C. The last three columns indicate the position of the computed transition center (CTC), see text for explanations.

be on the right side of the optimal sweep line). In the case that there is an entire region of optimal sweep line positions, we took the center of this region. In the following, we call this time position the *computed transition center* (CTC). In our evaluation, we then investigated whether the CTC lies within the annotated transition R-T or not. In the case that the CTC is not in R-T, it may be located in R-$G1$ or in R-$G2$. In the first case, we computed a negative number indicating the directed distance given in seconds between the CTC and the end of R-$G1$, and in the second case a positive number indicating the directed distance between the CTC and the beginning of R-$G2$. Table 2 shows the results of this evaluation, which demonstrates that for most recordings the CTC is a good indicator for R-T. The poorer values are in most case due to the deviations in the composition from the music theory. Often, the modulation differences between exposition and recapitulation already start within the final section of the first subject group, which explains many of the negative numbers in Table 2. As for the late sonatas such as `Op106-01` (No. 29) or `Op110-01` (No. 31), Beethoven has already radically broken with conventions, so that our automated approach (being naive from a musical point of view) is deemed to fail for locating the transition.

4. CONCLUSIONS

In this paper, we have introduced automated methods for analyzing and segmenting music recordings in sonata form. We adapted a thumbnailing approach for detecting the coarse structure and introduced a rule-based approach measuring local harmonic relations for analyzing the finer substructure. As our experiments showed, we achieved meaningful results for sonatas that roughly follow the musical conventions. However, (not only) automated methods reach their limits in the case of complex movements, where the rules are broken up. We hope that even for such complex cases, automatically computed visualizations such as our introduced WRTI (weighted relative transposition index) representation may still yield some musically interesting and intuitive insights into the data, which may be helpful for musicological studies.

Acknowledgments: This work has been supported by the German Research Foundation (DFG MU 2682/5-1). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer IIS.

5. REFERENCES

[1] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, 2005.

[2] Roger B. Dannenberg and Masataka Goto. Music structure analysis from acoustic signals. In David Havelock, Sonoko Kuwano, and Michael Vorländer, editors, *Handbook of Signal Processing in Acoustics*, volume 1, pages 305–331. Springer, New York, NY, USA, 2008.

[3] Masataka 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.

[4] Hugo Leichtentritt. *Musikalische Formenlehre*. Breitkopf und Härtel, 12. Auflage, Wiesbaden, Germany, 1987.

[5] Richard Middleton. Form. In Bruce Horner and Thomas Swiss, editors, *Key terms in popular music and culture*, pages 141–155. Wiley-Blackwell, 1999.

[6] Meinard Müller and Michael Clausen. Transposition-invariant self-similarity matrices. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 47–50, Vienna, Austria, 2007.

[7] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 215–220, Miami, FL, USA, 2011.

[8] Meinard Müller, Nanzhu Jiang, and Peter Grosche. A robust fitness measure for capturing repetitions in music recordings with applications to audio thumbnailing. *IEEE Transactions on Audio, Speech & Language Processing*, 21(3):531–543, 2013.

[9] Meinard Müller and Frank Kurth. Towards structural analysis of audio recordings in the presence of musical variations. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007.

[10] Jouni Paulus, Meinard Müller, and Anssi P. Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 625–636, Utrecht, The Netherlands, 2010.

[11] Geoffroy Peeters. Deriving musical structure from signal analysis for music audio summary generation: "sequence" and "state" approach. In *Computer Music Modeling and Retrieval*, volume 2771 of *Lecture Notes in Computer Science*, pages 143–166. Springer Berlin / Heidelberg, 2004.

[12] 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 *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, pages 555–560, Miami, FL, USA, 2011.

[13] Donald Francis Tovey. *A Companion to Beethoven's Pianoforte Sonatas*. The Associated Board of the Royal Schools of Music, 1998.

COMBINING HARMONY-BASED AND NOVELTY-BASED APPROACHES FOR STRUCTURAL SEGMENTATION

Johan Pauwels, Florian Kaiser and Geoffroy Peeters
STMS IRCAM-CNRS-UPMC
johan.pauwels@ircam.fr, florian.kaiser@ircam.fr, geoffroy.peeters@ircam.fr

ABSTRACT

This paper describes a novel way to combine a well-proven method of structural segmentation through novelty detection with a recently introduced method based on harmonic analysis. The former system works by looking for peaks in novelty curves derived from self-similarity matrices. The latter relies on the detection of key changes and on the differences in prior probability of chord transitions according to their position in a structural segment. Both approaches are integrated into a probabilistic system that jointly estimates keys, chords and structural boundaries. The novelty curves are herein used as observations. In addition, chroma profiles are used as features for the harmony analysis. These observations are then subjected to a constrained transition model that is musically motivated. An information theoretic justification of this model is also given. Finally, an evaluation of the resulting system is performed. It is shown that the combined system improves the results of both constituting components in isolation.

1. INTRODUCTION

Structural segmentation of music is the process in which an audio recording is divided into a number of non-overlapping sections that correspond to the macro-temporal organisation of a piece. These entities usually take the form of verses and choruses in popular music, or of movements in classical music. The obtained sections can then be used for interactive listening, audio summarization, synchronization or as an intermediate step in further content-based indexing.

Traditional approaches to structural segmentation have been categorized into three categories [14]: repetition-based, novelty-based and homogeneity-based methods. A mid-level representation, called self-similarity matrix, is often used for these task. It is obtained from the feature sequence by comparing each instance with all time-delayed copies of itself according to some similarity measure. The result is a visualisation of the musical structure. It was originally introduced into the music domain by Foote [2].

Repetition-based approaches rely on the hypothesis that recurring patterns in the feature sequence cause a perception of a higher structure. In a self-similarity matrix this becomes visible as stripes on the off-diagonals [4]. Novelty-based systems try to identify transitions between two contrasting parts, which are also perceived as structural boundaries by humans [1]. Initially, these methods were the dual approach of homogeneity-based methods [10], as one was looking exclusively for transitions between two distinct sections that are similar according to some musical property [3]. Recently however, this approach has been extended to also include contrast between a homogeneous and a non-homogeneous section [6].

The method we propose builds upon the novelty-based method of [3], but integrates this with a novel approach that is based on the estimated harmony of the piece [16]. Previous efforts of deriving a structural segmentation from harmony have mostly been concerned with using chroma features for the construction of a self-similarity matrix, instead of or in addition to timbre-related features [5]. We however work with a higher-level harmony description, in the form of key and chord estimates. In this sense, our approach is somewhat similar to previous systems by Maddage [11] or Lee [8], but in contrast to their systems, ours works simultaneously, not sequentially. They first extract key and chord estimates, which they subsequently use as inputs for a structure estimation. A sequential system can also be constructed the other way around, using an estimate of the structure as input to aid with chord estimation. An example of this kind is the method of Mauch [12]. We, on the other hand, construct a probabilistic system that jointly estimates keys, chords and structural boundaries. It is based on the assumption that some chord combinations are more common around structural boundaries. This is especially clear when they are expressed as relative chords in a key, as this gives a musicologically richer representation. These relative chord combinations will then be used as evidence for structural boundaries, together with the positions of key changes and peaks in the novelty measure.

In the remainder of this paper, we'll first give an outline of our probabilistic system in Section 2.1. Then we'll go deeper into the details of how to integrate the harmony-based and the novelty-based approach into this framework in Section 2.2, respectively Section 2.3. Afterwards, we describe the experiments we performed and analyse the results in Section 3. We conclude with some closing remarks and ideas for future work in Section 4.

2. A PROBABILISTIC SYSTEM FOR THE JOINT ESTIMATION OF KEYS, CHORDS AND STRUCTURAL BOUNDARIES

2.1 Overview

In this section we will describe the probabilistic system that we propose for the determination of a structural segmentation of a track, along with an estimation of the local keys and chords. As a starting point we use the system of Pauwels et al. [15] for the simultaneous estimation of keys and chords. It consists of an HMM in which each state represents a combination of a key and a chord. We extend it by letting each state q represent a structural position in addition to a key and a chord. A key k can take one of N_k values, a chord one of N_c values and the structural positions s can take one of two values: L which means that q is the last state of a structural segment or O which means that it is not. Finally, we add a single state to handle the case when no chord is being played, notably at the beginning and end of a recording. In this state, the key will accordingly take a "no-key" value and the structural position will take a value of $s = R$. In summary, $q = (s, k, c)$ with $s \in \{L, O\}, k \in \{K_1, \ldots, K_{N_k}\}, c \in \{C_1, \ldots, C_{N_c}\}$ or $q = (R, \text{no-key}, \text{no-chord})$.

Finding the most likely sequence of key, chord and structure labels given a sequence of observations $X = \{x_1, x_2, \ldots, x_T\}$ then amounts to finding the state sequence $\hat{Q} = \{\hat{q}_1, \hat{q}_2, \ldots, \hat{q}_T\}$ that optimally explains these observations. Because the state variable q consists per definition of the combination of a chord, key and structure variable, these three optimal sequences will always be jointly estimated. Afterwards, the structural boundaries $\hat{S}$ can be derived from the optimal state sequence by inserting a boundary for every transition from a state where $s = L$ to one where $s = O$, or from or to a state with $s = R$. The derivation of the optimal key $\hat{K}$ and chord sequence $\hat{C}$ from the latter is even more trivial.

By applying Bayes' theorem, and further assuming the first order Markov property and independence of the observations, we can rewrite the probability to be maximized to

$$\hat{S}, \hat{K}, \hat{C} = \arg\max \prod_{t=1}^{T} P\left(\mathbf{x_t}|s_t, k_t, c_t\right) P\left(s_t, k_t, c_t|s_{t-1}, k_{t-1}, c_{t-1}\right)$$

The time-interval t here indicates the index of the interbeat segments, where the beats are estimated by ircambeat [17]. The probabilities of this HMM will now be determined by the combination of 2 different components that are each using their separate observations. The first component is a method to estimate a structural segmentation simultaneously with the harmony of a piece. It uses chroma features. The second component determines structural boundaries based on a novelty measure that is derived from timbral features. For clarity reasons, we introduce separate notations for the chroma observations $\mathbf{y_t}$ and for the novelty measure $\mathbf{z_t}$ (so $\mathbf{x_t} = [\mathbf{y_t}\mathbf{z_t}]$). We will consider the novelty observations independent of the chroma observations:

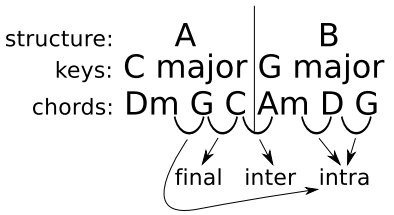


Figure 1. An example annotated sequence with the three structure dependent positions indicated

$P\left(\mathbf{x_t}|s_t, k_t, c_t\right) = P\left(\mathbf{y_t}|s_t, k_t, c_t\right) P\left(\mathbf{z_t}|s_t, k_t, c_t\right)$. The final probability to be maximized will therefore be

$$\hat{S}, \hat{K}, \hat{C} = \arg\max \prod_{t=1}^{T} P\left(\mathbf{y_t}|s_t, k_t, c_t\right) P\left(\mathbf{z_t}|s_t, k_t, c_t\right) P\left(s_t, k_t, c_t|s_{t-1}, k_{t-1}, c_{t-1}\right)$$

Of these three terms, the chroma observation probability $P\left(\mathbf{y_t}|s_t, k_t, c_t\right)$ and the transition probability $P\left(s_t, k_t, c_t|s_{t-1}, k_{t-1}, c_{t-1}\right)$ will be set by the harmony-based component of our system, while the novelty observation probability $P\left(\mathbf{z_t}|s_t, k_t, c_t\right)$ will be set by the novelty-based component.

We use different observations for both components to capture different types of information, in hopes of them being complementary. For instance, a change of instrumentation won't be detected as a structure boundary by the harmony-based component, while the novelty-based component won't recognize a chord sequence that is typical for an ending. The choice of their respective observations reflects this.

2.2 The harmony-based component

2.2.1 Motivation and information theoretical justification

The basic premise upon which our approach is built, is that chord sequences, and more specifically chord pairs, exhibit a different prior probability depending on their position with respect to structural boundaries. In addition to the specificity of these chord combinations, we also argue that the number of distinct chord pairs at the end of a segment is lower compared to all possible chord sequences in the middle of a structural segment. Structural boundaries seem an implausible place to experiment with some less common chord combinations, established by well-known, diatonic chord combinations. Real world examples that support these statements are the observation that movements in classical music typically end with one of a select number of chord combinations, called cadences, or that musical section changes in jazz and blues are often preceded by so-called "turn-arounds".

In order to verify these statements in a methodological way, we first identify three categories of chord pairs based on their position with respect to structural boundaries. An example sequence for which these three categories are indicated can be found in Figure 1. The first class is named *final* and this contains the chord pairs that form the two last chords of a structural segment. The second category we'll

	Isophonics		Quaero	
	major	minor	major	minor
intra	6.17	6.25	4.29	4.98
inter	3.91	4.52	2.99	2.37
final	2.91	3.51	2.36	3.18

Table 1. Perplexity of relative chord transition models per mode according to structural position

call *inter* and this consists of all chord pairs that straddle a structural boundary. The last class of chord pairs is called *intra* and this includes all the remaining chord pairs, the ones that occur in the beginning and middle of a structural segment.

To reason about chord pairs in a musicologically more informative way, we will interpret them as relative chords interpreted in a key. This representation reflects more closely the way scholars analyse harmonic movement. Also in accordance to musicological analysis, both chords will be interpreted in the same key. Because of the forward motion in music, this will be the key annotated at the time of the first chord. In mathematical notation, we will define a key k as the combination of a tonic t and a mode m. A chord c is defined as the combination of a root r and a type p. Both t and r belong to one of the 12 different pitch classes. We restrict ourselves in this paper to 2 modes and 4 chord types ($m \in \{\text{major}, (\text{natural})\text{minor}\}$, $p \in \{\text{maj}, \text{min}, \text{dim}, \text{aug}\}$). We then define a relative chord c' with respect to a key k by expressing the root r as the interval between the tonic and the root: $i = d(t, r)$. Therefore we can equivalently express a key-chord pair as a key-relative chord pair $(k, c) = (t, m, r, p) = (t, m, i, p) = (k, c')$. In order to take the inherent shift-invariance in harmony analysis into account, we just ignore the tonic of the key and only keep the mode. For sequences of 2 key-chord pairs, we end up with just a mode and a pair of relative chords as a representation for the local harmony: $\left(m_n, c'_n, c'_{n+1}\right)$ where n is the chord index.

We then construct relative chord transition models for each of the three structural categories by counting occurrences of all successions of 2 consecutive relative chords in a corpus that is annotated with keys, chords and structural segments. Sequences that do not appear in the data set are assigned a probability using Kneser-Ney smoothing [7]. The corpus should be annotated such that all the positions are indicated where at least one of key, chord or structural segment changes. We have two such data sets at our disposal. The first one is the publicly available "Isophonics" set and more specifically the subset that has been used for the MIREX 2010 chord estimation competition. It consists of 217 full songs, mostly by the Beatles (180 songs), the remainder by Queen (20) and Zweieck (17). The second one is a private data set called "Quaero" and it contains 53 songs from a number of diverse artists in the popular genre. We can now quantify the difference in relative chord distribution between the various structure dependent transition models by calculating the bigram model perplexity $PP\left(C'_1, C'_2|m\right)$ per mode m for each of them. C'_1 and C'_2 represent the collection of all relative chords that appear as first, respectively second, element in the bigrams. The model perplexity is defined as the exponential of the entropy $H\left(C'_1, C'_2|m\right)$ expressed in nats:

$$\begin{aligned} PP(C'_1, C'_2|m) &= \exp\left(H\left(C'_1, C'_2|m\right)\right) \\ &= \exp\left(-\sum_{c'_1, c'_2} P\left(c'_1, c'_2|m\right) \log P\left(c'_2|c'_1, m\right)\right) \\ &= \exp\left(-\sum_{c'_1} P\left(c'_1|m\right) \right. \\ &\qquad \left. \sum_{c'_2} P\left(c'_2|c'_1, m\right) \log P\left(c'_2|c'_1, m\right)\right) \end{aligned}$$

This expresses the mean prior uncertainty of a bigram as a function of its mode. A lower value means that the transition probability is concentrated into fewer combinations of two chords. The perplexities for both data sets can be found in Table 1 and as can be seen, they confirm our hypothesis: the values for the "intra"-model are indeed significantly higher than those for the "inter" and "final"-model and this for both corpora.

For the calculation of the transition probabilities in the following paragraphs, we will make use of the information captured in these structure dependent relative chord transition models. The reasoning will therefore be reversed: instead of showing that a structural boundary often suggests a specific set of relative chord pairs, we'll use the occurrence of such relative chord combinations as evidence to estimate structural boundaries.

2.2.2 *Transition probabilities*

The transition probabilities $P(s_t, k_t, c_t|s_{t-1}, k_{t-1}, c_{t-1})$ are calculated by a prior musicological model that consists of a number of submodels. By introducing some musicologically motivated constraints to the transition probabilities, we want to enforce a number of relationships between the concepts of keys, chords and structural segments. These will ensure that our estimation always produces sensible results and have as an added benefit that this also speeds up the calculation. The first three constraints we impose are 1) a key change $k_t \neq k_{t-1}$ is only allowed to occur together with a chord change $c_t \neq c_{t-1}$, 2) a structural segment must contain at least two different chords (or a single no-chord), 3) there must be a change in chord or in key between segments. These three limitations can be easily enforced by ensuring that every state change implies a chord change. This makes the state duration model effectively a chord duration model that we control by a single parameter P_s:

$$\begin{aligned} &P\left(s_t, k_t, c_t|s_{t-1}, k_{t-1}, c_{t-1}\right) = \\ &\qquad \begin{cases} P_s & s_t = s_{t-1} \wedge k_t = k_{t-1} \\ 0 & s_t \neq s_{t-1} \vee k_t \neq k_{t-1} \end{cases}, \forall c_t = c_{t-1} \end{aligned} \quad (1)$$

We use the same value for P_s as found in the original system [15].

The remaining probabilities $P(s_t, k_t, c_t|s_{t-1}, k_{t-1}, c_{t-1}), \forall c_t \neq c_{t-1}$ of the chord changing transitions are calculated by the combination of three submodels. We further apply Bayes' theorem repeatedly to arrive at a decomposition into three terms

$$\begin{aligned} &P\left(s_t, k_t, c_t|s_{t-1}, k_{t-1}, c_{t-1}\right) \\ &\quad = P\left(s_t|s_{t-1}, k_{t-1}, c_{t-1}\right) P\left(c_t|s_t, s_{t-1}, k_{t-1}, c_{t-1}\right) \\ &\qquad P\left(k_t|s_t, s_{t-1}, k_{t-1}, c_t, c_{t-1}\right) \end{aligned}$$

The first term $P\left(s_t|s_{t-1}, k_{t-1}, c_{t-1}\right)$ will be used to control the ease of changing the structure variable s and thus to control the insertion rate of segment boundaries. We use a simple model that ignores the key and chord influence and consists of a single parameter ω that balances the probability of going to $s = O$ or $s = L$ after leaving $s = O$.

$$P\left(s_t|s_{t-1}\right) = \begin{cases} \omega & s_{t-1} = O, s_t = O \\ 1-\omega & s_{t-1} = O, s_t = L \\ 1 & s_{t-1} = L, s_t = O \\ 0 & s_{t-1} = L, s_t = L \end{cases}$$

We can already recognize the structure-dependent relative chord transition model of the previous section in the second term $P\left(c_t|s_t, s_{t-1}, k_{t-1}, c_{t-1}\right)$. Our three categories of chord transitions – *inter*, *intra* and *final* – each correspond to a certain combination of the state variables. The *intra* model will be used when $s_{t-1} = O$ and $s_t = O$, *inter* when $s_{t-1} = L$ and $s_t = O$ and *final* when $s_{t-1} = O$ and $s_t = L$. Finally, from our definition of L it follows that when $s_{t-1} = L$ and $s_t = L$, only the self probability P_s should be allowed, to account for the fact that the last chord of a structural segment can – and most likely will – last more than one time step. The other probabilities are set to zero.

Since we already established that a key change implies a chord change, we can neglect the influence of the chords c_{t-1}, c_t in the third term $P\left(k_t|s_t, s_{t-1}, k_{t-1}, c_t, c_{t-1}\right)$. We thus end up with $P\left(k_t|s_t, s_{t-1}, k_{t-1}\right)$. Furthermore, we impose the supplemental constraint that a key change can only occur between segments. Mathematically, this can be expressed as $s_{t-1} = O \Rightarrow P\left(k_t|s_t, s_{t-1}, k_{t-1}\right) = \delta_{k_t, k_{t-1}}$ with δ the Kronecker-delta. For the *inter* key transitions $P(k_t|s_t = O, s_{t-1} = L, k_{t-1})$, we reuse the theoretical model from [15], based on Lerdahl's distance [9] between keys.

In Figure 2 one can find a simplified state diagram of our system, in which states are regrouped by the structure variable. Only the transitions from the point of view of the structure variable are drawn in order not to overload the picture, but the constraints on key and chord transitions are indicated next to the arrows. The names of the structure dependent relative chord transition models are also indicated.

2.2.3 System complexity

The result of adding the additional constraints is that the complete transition matrix will have a well-defined, sparse

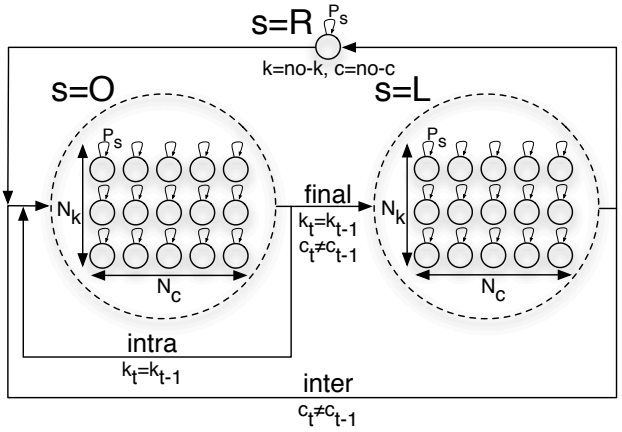


Figure 2. State diagram of the system

structure. The two upper quadrants consist of block diagonal matrices with N_k blocks of side N_c, the lower left quadrant is dense and the lower right quadrant is a diagonal matrix. In comparison to a system that only estimates keys and chords concurrently, the number of states gets doubled by repeating every key-chord state for $s = O$ and $s = L$. On the other hand, because of the sparsity of the transition matrix, the increase in the number of transitions remains limited. More specifically, the number of transitions is $(N_k N_c)^2 + 2N_k N_c^2 + N_k N_c + 1$, which corresponds in our configuration to an increase of 8% instead of the theoretically maximum of 400% that would be reached for a dense transition matrix. This sparsity is exploited in the implementation of the Viterbi algorithm to limit the increase in computation time.

2.2.4 Chroma emission probabilities

As our chroma features, we use the implementation by Ni et al. [13] known as Loudness Based Chromagrams. These are 24-dimensional vectors that represent the loudness of each of the 12 pitch classes in both the treble and the bass spectrum. They are calculated with a hop size of 23 ms and are afterwards averaged over the interbeat interval. We make the assumption that keys and chords can be independently tested for compliance with an observation and that the structure position is conditionally independent of the observations, such that $P\left(\mathbf{y_t}|s_t, k_t, c_t\right) = P\left(\mathbf{y}_t|c_t\right) P\left(\mathbf{y}_t|k_t\right)$. The chord acoustic probability $P\left(\mathbf{y}_t|c_t\right)$ is modelled as a multi-variate Gaussian with full covariance matrix. Its parameters are trained on the aforementioned "Isophonics" data set. The key acoustic probability $P\left(\mathbf{y}_t|k_t\right)$ is calculated by taking the cosine similarity between the observation vector y_t and Temperley's key templates [18]. These represent the stability of each of the 12 pitch classes relative to a given key.

2.3 The novelty-based component

The other major component of our system is a novelty-based structural segmentation algorithm. We'll use a simple implementation that is conceptually very close to Foote's original proposal [3]. First, a self-similarity ma-

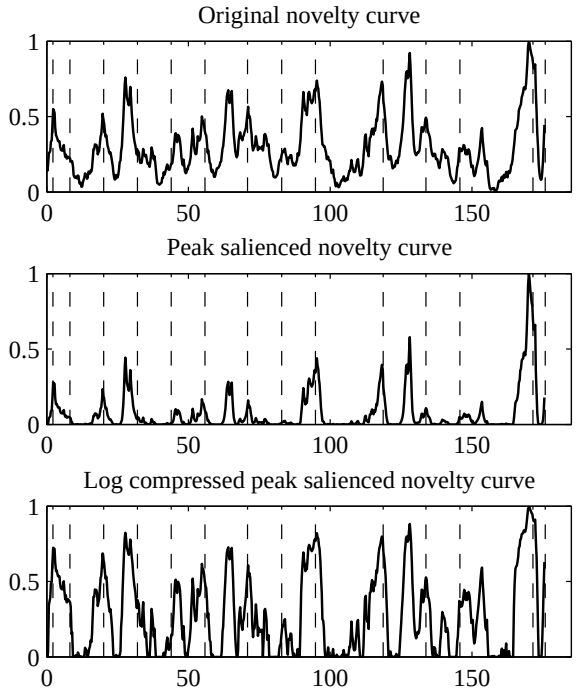


Figure 3. An example of the transformations of the novelty curve

trix is calculated from a sequence of MFCC's and the first four spectral moments (spectral centroid, spread, skewness and kurtosis). The similarity measure that is used is the cosine similarity. From this matrix, a time varying novelty curve is derived by convolving the matrix with a two-dimensional novelty kernel along its diagonal. We use a kernel size of 22.5 s and a step size of 250 ms. In a stand-alone system, the peaks of the resulting novelty curve are detected and structural boundaries are inserted at those positions. In our case however, we'll use the complete curve to calculate the novelty observation probability $P(\mathbf{z_t}|s_t, k_t, c_t)$.

We assume that the novelty observations are conditionally independent of chord and key state, such that we end up with $P(\mathbf{z_t}|s_t, k_t, c_t) = P(\mathbf{z_t}|s_t)$. We thus need to model the novelty observations for each of the possible values of s_t, i.e. O, L or R. By definition, there is a clear relation between the peaks of the novelty curve and a high probability of being in a state that will insert a structural boundary upon leaving it ($s = O$ or $s = R$). Therefore we model the structure acoustic probability for the L and R states as a (half) Gaussian centered on 1. Likewise, the probability of the $s = O$ states will be modelled by a half Gaussian centered on 0. However, we won't use the values of the novelty curve directly as the observations $\mathbf{z_t}$ for the novelty observation probability $P(\mathbf{z_t}|s_t)$. A first remark is that the novelty curves are designed to be used in conjunction with peak picking. Therefore their relative value with respect to surrounding valleys is what matters, and not their absolute value as is desired for our probabilistic system. In order to adapt the novelty values to our use, we therefore first transform the values of the curve. If we represent the values of the novelty curve by $\mathbf{v}$, then the transformation is the following:

$$v'_j = (v_j - \min(v_{j-w} : v_j))\,(v - min(v_j : v_{j+w}))$$

where j is the index of the novelty curve, which has a fixed sample rate of 4 Hz, and w the size of a window around the j-th value. An example of this transformation can be seen in Figure 3, where the original signal is represented in the first row and the processed one in the second. The effect is that valleys now reach all the way down to 0 and that highs indicate peaks with a high salience. As evident in our example, there can be quite a large difference between the saliences of the peaks corresponding to annotated segments. In order to diminish the differences, we apply a log compression, which is subsequently rescaled to the interval $[0, 1]$ for convenience:

$$v''_j = log_{10}\left(1 + \alpha v'_j\right)$$

The result on our example curve is shown in the third row of Figure 3. Finally, we want to account for small deviations of the peak positions with respect to the actual structural boundaries. Therefore we look for extrema of the log-compressed novelty curve in the current beat segment and the segments that are adjacent on each side. For $P(z_t|s_t = L)$ and $P(z_t|s_t = R)$ this will be the maximum and for $P(z_t|s_t = O)$ the minimum, so that the final dimension of $\mathbf{z_t}$ will actually be 2-dimensional: one dimension with the local minima of v'' and one with its local maxima. This step also changes the time scale from a fixed step size of 250 ms to beat-synchronous observations (index j to t).

3. EXPERIMENTAL RESULTS

In this section, we will evaluate our system for various configurations. We evaluate the structural segmentation by calculating the precision $\mathcal{P}(tol)$ and recall $\mathcal{R}(tol)$ between the generated and the annotated structure. They are a function of a tolerance interval tol whose purpose is to allow for small deviations from the desired result to be still considered correct. The precision is defined as the number of estimated boundaries for which an annotated boundary lies within the tolerance interval centered around its position divided by the total number of estimated boundaries. The recall on the other hand, is the relative number of annotated boundaries that have an estimated boundary within its tolerance interval. Both measures are combined in an F-measure $\mathcal{F}(tol)$. These measures are calculated for every song of the data set and are afterwards averaged to give one global result.

We calculate the results for two tolerance intervals, 0.5 s and 3 s, in accordance with the MIREX structure segmentation competition. Both the harmony-based and the novelty-based system were tested separately, as well as the combination of both approaches. For the harmony-based and the combined system, we performed our experiments twice: once with the structure dependent relative chord models derived from the Isophonics data set and once with

	$\mathcal{F}(3s)$	$\mathcal{F}(0.5s)$
harmony-based (Isophonics)	54.72	34.90
harmony-based (Quaero)	52.44	29.47
novelty-based	61.84	33.53
combined (Isophonics)	64.38	35.41
combined (Quaero)	64.13	34.08

Table 2. Results on the Isophonics data set

those from the Quaero set. The results on the Isophonics data can be found in Table 2. We can see that the combination of both approaches has a synergetic effect. Separately, they each have different strengths. The harmony-based approach is better in precisely locating the structure boundaries, as apparent from the $\mathcal{F}(0.5s)$ results, while the novelty-based approach performs better when a larger deviation is allowed. As could be expected, the harmony-based and combined systems work better with the Isophonics relative chord models, since they are perfectly matched with the test set. However, most of the synergy remains when using the models learned on the Quaero set, showing the generality of these models. Additionally, the combined system is also less sensitive to the choice of relative chord models than the harmony-based method is.

4. CONCLUSION

In this paper, we proposed a method for structure estimation by combining 2 different approaches. The first is a traditional way of segmenting structure based on a timbral novelty measure. The second is based on a harmonic analysis that is performed concurrently with the structure estimation. It makes use of chroma features. Together they form a probabilistic system for the simultaneous estimation of keys, chords and structure boundaries. We've shown that the combination of both approaches works better than each of the two systems on its own.

In the future, we will experiment with a post-processing step to extend our resulting structural segmentation into a full structure estimation that includes the identification and labelling of repeated segments. After all, for each of our estimated segments we have a harmonic analysis available that could be used as a feature for the clustering of similar segments, in addition to the more low-level features that are currently used for this task.

5. ACKNOWLEDGEMENTS

This work was partly supported by the Quaero Program funded by Oseo French agency.

6. REFERENCES

[1] M. J. Bruderer, M. McKinney, and A. Kohlrausch. Structural boundary perception in popular music. In *Proc. ISMIR*, 2006.

[2] J. Foote. Visualizing music and audio using self-similarity. In *Proc. ACM Multimedia*, 1999.

[3] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *Proc. ICME*, 2000.

[4] M. Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Trans. Audio, Speech, Language Process.*, 15(5), 2006.

[5] K. Jensen. Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal Advances in Signal Processing*, 073205, 2007.

[6] F. Kaiser and G. Peeters. Multiple hypotheses at multiple scales for audio novelty computation within music. In *Proc. ICASSP*, 2013.

[7] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *Proc. ICASSP*, 1995.

[8] K. Lee. *A system for acoustic chord transcription and key extraction from audio using hidden Markov models trained on synthesized audio*. PhD thesis, Stanford University, 2008.

[9] F. Lerdahl. *Tonal pitch space*. Oxford University Press, New York, 2001.

[10] M. Levy and M. Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2), 2008.

[11] N. C. Maddage. Automatic structure detection for popular music. *IEEE MultiMedia*, 13(1), 2006.

[12] M. Mauch and S. Dixon. Using musical structure to enhance automatic chord transcription. In *Proc. ISMIR*, 2009.

[13] Y. Ni, M. McVicar, R. Santos-Rodríguez, and T. De Bie. An end-to-end machine learning system for harmonic analysis of music. *IEEE Transactions on Audio, Speech and Language Processing*, 20(6), 2012.

[14] J. Paulus, M. Müller, and A. Klapuri. Audio-based music structure analysis. In *Proc. ISMIR*, 2010.

[15] J. Pauwels, J.-P. Martens, and M. Leman. Improving the key extraction accuracy of a simultaneous key and chord estimation system. In *Proc. ICME*, 2011.

[16] J. Pauwels and G. Peeters. Segmenting music through the joint estimation of keys, chords and structural boundaries. In *Proc. ACM Multimedia*, 2013.

[17] 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.

[18] D. Temperley. *The cognition of basic musical structures*. MIT Press, 1999.

AUTOMATIC ALIGNMENT OF MUSIC PERFORMANCES WITH STRUCTURAL DIFFERENCES

Maarten Grachten[1] Martin Gasser[1]
[1]Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria
http://www.ofai.at/~maarten.grachten

Andreas Arzt[2], Gerhard Widmer[1,2]
[2]Dept. of Computational Perception Johannes Kepler Universität, Linz, Austria

ABSTRACT

Both in interactive music listening, and in music performance research, there is a need for automatic alignment of different recordings of the same musical piece. This task is challenging, because musical pieces often contain parts that may or may not be repeated by the performer, possibly leading to structural differences between performances (or between performance and score). The most common alignment method, dynamic time warping (DTW), cannot handle structural differences adequately, and existing approaches to deal with structural differences explicitly rely on the annotation of "break points" in one of the sequences. We propose a simple extension of the Needleman-Wunsch algorithm to deal effectively with structural differences, without relying on annotations. We evaluate several audio features for alignment, and show how an optimal value can be found for the cost-parameter of the alignment algorithm. A single cost value is demonstrated to be valid across different types of music. We demonstrate that our approach yields roughly equal alignment accuracies compared to DTW in the absence of structural differences, and superior accuracies when structural differences occur.

1. INTRODUCTION AND RELATED WORK

A variety of music processing scenarios involve alignment of music in the form of either symbolic scores, or audio recordings (or both). In some cases, alignment is used to compute a similarity score between instances of a musical piece. This is useful for example in plagiarism detection [7] and cover song identification [3, 19]. In other cases, it is the alignment itself that is of use. Examples are automatic transcription [20], computer assisted music production [14], real-time score-following for automatic page turning [1], and automatic accompaniment [5, 6].

There are several factors that make accurate alignment of music a challenging task. Firstly, in case of audio alignment, the acoustic properties of the recordings may be very different, due to differences in instrumentation, recording, mixing, and mastering. Secondly, interpretations of musical pieces by human performers tend to have expressive variations, causing different interpretations of a piece to diverge in both global and local tempo and dynamics. Thirdly, performance errors may lead to occasional missing, or inserted notes. A fourth complicating factor is the fact that musical pieces are often composed of smaller musical units, where units may be repeated or not, or even left out completely, according to the taste of the musician or conductor. This may lead to what we refer to as *structural differences* between performances of the piece.

The problem of aligning music with structural differences has been addressed in a number of studies. In most of these, the problem setting is score-to-performance alignment, in which a symbolic representation of a musical score is mapped to a performance of that score. In a symbolic score representation, it is relatively easy to mark points where performances are likely to diverge. For example, Fremerey et al. [9] develop a method that relies on explicit annotations of possible jump points in the score where double bar lines occur. A similar approach is taken by Pardo and Birmingham [18].

In performance-performance alignment, as opposed to score-performance alignment, it is generally not possible to rely on such annotations, since there is no score representation involved. Müller and Appelt [16] propose a method to deal with structural differences in performance-performance alignment. This approach uses dynamic time warping (DTW) in combination with pre-processing of the similarity matrix, and post-processing of alignment paths.

In this paper we start from the observation that DTW has shortcomings when dealing with structural differences in music recordings (Section 2). Our intention is to show that other variants of dynamic programming alignment are more effective. In particular, it is beneficial to include skip operations, as well as one-to-many and many-to-one matching, as in the algorithm of Mongeau and Sankoff [13], who use this approach for measuring similarity between melodies as sequences of notes, and Grachten et al. [10], who design alignment operations to capture the semantics of expressive musical behavior, like spontaneous ornamentations of notes in a performance. To our knowledge, such extensions of the classical dynamic programming variants have not been used in the context of audio alignment.

Along with this alternative alignment method (Section 3.1), we propose a method to estimate the optimal value for

the gap penalty, a parameter that controls the behavior of the alignment (Section 3.3). In Section 4, we experimentally determine the utility of various audio features with respect to the effectiveness of the gap penalty across different types of music. Based on the most successful feature, and the corresponding optimal gap penalty, we perform a quantitative evaluation of the alignment accuracy of our proposed approach, in comparison to DTW (Section 4.2).

2. PROBLEM DESCRIPTION

Aligning two sequences requires a distance measure that quantifies how different their elements are. In Section 4.1.1, we will discuss different types of features and distance measures in more detail in the context of audio alignment. For now, let s and t be the sequences to be compared, of lengths M and N respectively. We refer to s and t as the *source* and *target* sequence, respectively. We use $d(i, j)$ to denote the distance between the i-th element of s and the j-th element of t, where $1 \leq i \leq M$ and $1 \leq j \leq N$.

2.1 Dynamic time warping (DTW)

DTW computes the minimal cost of aligning s and t. It can be expressed as $\text{dtw}(M, N)$, where dtw is defined by the recursive equation:

$$\text{dtw}(i,j) = \begin{cases} 0, & \text{if } i = 0, j = 0 \\ \infty, & \text{if } i = 0, j \neq 0 \text{ or } j = 0, i \neq 0 \\ d(i,j) + \min \begin{cases} \text{dtw}(i-1, j-1)) \\ \text{dtw}(i-1, j) \\ \text{dtw}(i, j-1), \end{cases} & \text{otherwise} \end{cases} \quad (1)$$

The alignment that leads to $\text{dtw}(M, N)$ is called the *optimal alignment*, and can be easily recovered by keeping track which argument of the $\min$ operator is selected in (1).

As the name of the algorithm states, dynamic time warping is a method to align sequences that are *time warped* versions of each other. That means that the sequences represent the same order of events, but the duration of events may differ from one sequence to the other. This time warping assumption explains why in DTW, each element of one sequence must be matched to an element of the other sequence. When the sequences are structurally different however, this assumption is violated: the sequences contain elements that are not to be matched to elements in the other sequence. By forcing a match between elements, DTW produces undesired alignments in such cases.

2.2 Needleman-Wunsch alignment (NW)

A solution to this problem is to allow the alignment algorithm to skip unmatchable parts of either sequence. The cost of skipping should not be proportional to the distances between the elements of the sequences, since these distances are not relevant in the case of unmatchable sequences. This type of alignment is achieved by another member of the family of dynamic programming algorithms for optimal sequence alignment – the Needleman-Wunsch algorithm (NW) [17]. This algorithm computes the minimal cost $\text{nw}(M, N)$ of aligning s and t using the equation:

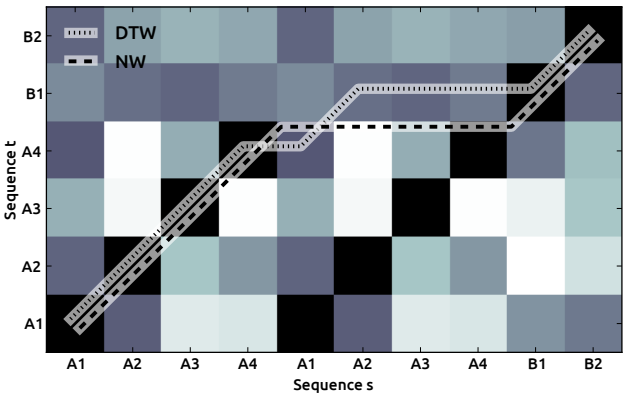


Figure 1. Distance matrix between two structurally different sequences; Dark cells represent low distances, light cells high distances; The DTW path jumps over a repeated section 'uncleanly', the NW path makes a clean jump

$$\text{nw}(i,j) = \begin{cases} 0, & \text{if } i = 0, j = 0 \\ \gamma + \text{nw}(i, j-1), & \text{if } i = 0, j \neq 0 \\ \gamma + \text{nw}(i-1, j), & \text{if } j = 0, i \neq 0 \\ \min \begin{cases} d(i,j) + \text{nw}(i-1, j-1) \\ \gamma + \text{nw}(i-1, j) \\ \gamma + \text{nw}(i, j-1), \end{cases} & \text{otherwise} \end{cases} \quad (2)$$

where γ is a constant referred to as *gap penalty*. (2) shows that the distance $d(i, j)$ between two elements i and j is only relevant to the alignment when it is sufficiently low. As soon as $d(i, j) > \gamma$, the algorithm will favor an insertion or deletion to a match (the final decision for an insertion or a deletion will only be made after the algorithm has processed the sequences entirely).

The difference between DTW and NW is illustrated in Figure 1, displaying the distances between the elements of two artificial sequences s and t, and the optimal DTW and NP alignments. The rows and columns are labeled to clarify the structure of s and t. In particular, s consists of two repetitions of a 4-tuple A, plus a 2-tuple B. Sequence t is the concatenation of one instance of A, and B. The DTW path aligns the elements of the second A in s partly to element $A4$ in t, and partly to element $B1$, where the exact alignment depends on the distances between those (non-matching) elements. The NW path (computed with a suitable value for γ) favors the deletion of the elements of the second occurrence of A over a sequence of poor matches. Note that this yields a clean and intuitive jump of the NW path across the second A in s.

2.3 Two problems of NW alignment

The use of Needleman-Wunsch for aligning music recordings introduces two problems. The first is that although NW handles structural differences, it does not handle *time*

warping. Since elements in the sequences can be either matched to a single other element, or skipped entirely, there is no way to deal with the fact that the music of the two recordings may be played at different tempos. Fortunately, a simple extension of the Needleman-Wunsch algorithm, described in Section 3.1, remedies this shortcoming.

The second problem is that – unlike DTW in its basic form [1] – NW involves a parameter γ, and the quality of the alignment will depend on the value of γ. Which value of γ gives good alignments will depend on the audio content and the features used to represent that content. In Section 3.3, we propose a method to estimate the optimal value for γ based on empirical data. In Section 4, we use this method to evaluate different features on various types of music.

3. PROPOSED SOLUTION

In this section we propose solutions to the two problems of NW alignment described above. Firstly, we propose an extension of the NW algorithm to deal with the time warping aspects of aligning music performances. Secondly, we describe a method to estimate the gap penalty γ.

3.1 Needleman-Wunsch time warping (NWTW)

DTW handles time warping by matching multiple elements of one sequence to a single element in the other sequence. Although this is not possible in the original NW algorithm, it is easy to add further arguments to the $\min$ operator, that represent many-to-one and one-to-many operations. In the following equation, which is a revision of (2), a 1-to-2 and a 2-to-1 operation have been included:

$$\mathrm{nw}(i,j) = \begin{cases} 0, & \text{if } i=0,\, j=0 \\ \gamma + \mathrm{nw}(i, j-1), & \text{if } i=0,\, j\neq 0 \\ \gamma + \mathrm{nw}(i-1, j), & \text{if } j=0,\, i\neq 0 \\ \min \begin{cases} d(i,j) + \mathrm{nw}(i-1, j-1) \\ d(i,j) + d(i,j-1) + \mathrm{nw}(i-1, j-2) \\ d(i,j) + d(i-1,j) + \mathrm{nw}(i-2, j-1) \\ \gamma + \mathrm{nw}(i-1, j) \\ \gamma + \mathrm{nw}(i, j-1), \end{cases} & \text{otherwise} \end{cases} \tag{3}$$

Appropriate names for these operations are *lengthen* and *shorten*, respectively, since the first is cost-effective when the music in t is up to two times slower than the music in s, and the second is cost-effective when it is (up to two times) faster. In case there is only a slight difference in tempo between s and t, shorten and lengthen operations occur only occasionally among a majority of *match* operations. Additional operations may be defined to handle even greater tempo differences, but such differences rarely occur in practice.

3.2 Algorithmic complexity

The NW algorithm – like DTW – requires the computation of a full matrix of intermediate results which are assembled into the final result in a backtracking step. This implies time and space requirements of order $O(MN)$, where M and N are the lengths of the two sequences. Compared to NW, our extension NWTW introduces a higher per-cell cost during the construction of the dynamic programming matrix, since we add *lengthen* and *shorten* operations that have to be taken into consideration when finding the optimal operation in (3). However, as this cost is constant and not dependent on M and N, it does not change the overall complexity of the algorithm.

In practice, NWTW based on fully computing the dynamic programming matrix is feasible on current desktop computers for audio files up to about 15 minutes. For longer audio files, we use multi-step dynamic programming [15], where full dynamic programming is used for downsampled feature vectors. Subsequent alignments for higher resolution feature vectors are computed for a band of fixed width around the previously computed (coarse) alignment path.

3.3 Estimation of optimal gap penalty γ

The gap penalty γ value serves as an upper bound on the distance between pairs of elements that are considered to match: if the distance is larger than γ, the alignment will favor skipping one of the elements. This means that the choice of γ is essentially a binary classification problem, in which pairs of elements are to be classified as *match* or *non-match*, based on their distance. Let $p(x|match)$ denote the distribution of distances between matching elements, and $p(x|non\ match)$ the distribution of distances between non-matching elements, then the optimal value $\hat{\gamma}$ can be defined as the value of γ that minimizes the expected classification error:

$$\begin{aligned} \hat{\gamma} &= \underset{\gamma}{\operatorname{argmin}} \int_0^{\gamma} p(x|non\ match)\ dx\ + \int_{\gamma}^{1} p(x|match)\ dx \\ &= \underset{\gamma}{\operatorname{argmin}} \int_0^{\gamma} p(x|non\ match) - p(x|match)\ \ dx \end{aligned} \tag{4}$$

Figure 2 shows $p(x|match)$, and $p(x|non\ match)$ for imaginary data, together with the corresponding optimal value of γ. Since DTW reliably finds correct alignments between recordings in the absence of structural differences [8], the alignments it produces on such recordings provide samples from the population of matching audio features, allowing us to estimate $p(x|match)$. By sampling randomly from the distance matrix (excluding cells on the DTW path), we obtain samples from $p(x|non\ match)$. With these distributions (which can often be well approximated by *beta-distributions*), we can obtain $\hat{\gamma}$ from data using a numerical approximation of (4).

Obviously, the actual form of these two distributions will depend on the musical content, the audio features, and the distance function used for alignment. In this context, the best combination of audio features and distance function is that which maximizes the divergence between the two distributions across musical content, since it facilitates distinguishing matching audio from non-matching audio.

[1] Extensions of DTW that include weights for operations are discussed in [15]

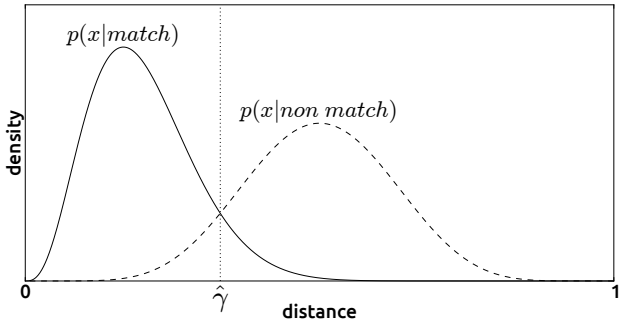


Figure 2. Schematic diagram of distance distributions between pairs of matching elements (solid), and non-matching elements (dashed); the optimal value of γ is indicated with a dotted vertical line

4. EXPERIMENTAL EVALUATION

In this section, we evaluate our proposed method NWTW by comparing it to DTW, on recordings both with and without structural differences. Before we do this, we assess different audio features by looking at how well they allow for separating matching from non-matching audio, as described in Section 3.3. Based on the results of that evaluation, we choose an audio feature, and choose the optimal γ for that feature. With this value of γ, we instantiate the NWTW algorithm, and perform a quantitative comparison of NWTW and DTW.

4.1 Choice of audio features and γ: Method and data

The purpose of this experiment is to find audio features for which the values in the distance matrix are as low as possible when they lie on the correct alignment path, and as high as possible otherwise. More specifically, we are interested in the features that maximize the divergence between the distance distributions of those two classes. The rationale for this is that with increasing divergence, the separation of the two classes by the gap penalty parameter will be more successful.

The pairs of audio recordings used for the evaluation are manually selected such that no structural differences occur. For each of the pairs, the optimal DTW path is computed to align the audio. From this alignment, the two distance distributions are computed.

4.1.1 Features

The features we evaluate are all known from the literature, including both standard features, such as MFCC and CQT coefficients, and more special purpose features such as PSD [8] and LNSO/NC [2]. Except for the LNSO/NC features (which are deliberately chosen to be used in conjunction with an adaptive distance function), all features are used in combination with the cosine distance measure, normalized to the interval $[0, 1]$.

Mel frequency cepstral coefficients (MFCC). Mel Frequency Cepstral Coefficients [12] are an STFT based audio representation well known in speech and music processing. MFCC's are a compressed version of the spectral envelope of a short-term section of an audio signal, and they are especially useful for capturing the timbre and the formant structure of speech/music signals. It is common to ignore the first MFCC coefficient, and to take only the first *n* coefficients. Here we evaluate both $n = 13$ and $n = 50$. In addition to that, we use two FFT sizes: 46ms, and 372ms.

Constant Q transform (CQT). The Constant Q transform [4] is a time-frequency transform, but unlike the STFT – which implements a constant bin width and therefore yields a non-constant bin center frequency to bin width ratio (the Q value) — it forces the Q value to stay constant and modifies the bin widths accordingly. The CQT is suitable for representing musical audio signals since its structure resembles the diatonic scale: all octaves are an equal number of bins apart.

Positive spectral difference (PSD). This feature was proposed in [8], and is designed to capture onset information for performance-to-performance alignment. It is based on a Short Time Fourier Transform with the frequency bins mapped to a musically meaningful scale. The PSD feature is computed as the half-wave rectification of the energy difference per frequency bin from one audio frame to the next.

Locally adaptive features/distance (LNSO/NC). For the purpose of audio alignment, Arzt et al. [2] propose to compute a weighted sum of distances of two features. The first, Locally Normalized Semitone Onset (LNSO) is an adaptation of PSD, and responds strongly to onsets. In absence of onsets, the distance is dominated by a second feature, Normalized Chroma (NC), capturing harmonic information.

4.1.2 Data

In order to ensure generality of the results beyond a single type of acoustic signals, we use three different classes of recordings (all obtained from commercial cd's):

Symphony orchestra: 7 Pieces from 5 different Beethoven symphonies, by 10 different conductors, amounting to 148 comparisons between 59 recordings (4.4 hours of music).

Solo guitar: The complete guitar works of Villa-Lobos (23 pieces), by 5 performers, amounting to 103 comparisons between 83 recordings (4.6h).

Solo piano: 14 Movements from 6 different piano sonatas by Mozart, by 7 performers, amounting to 720 comparisons between 308 recordings (6.4h).

4.2 Comparison of NWTW and DTW: Method and data

In this part of the experimentation we evaluate alignment accuracies quantitatively using manual annotations of the beat in a set of recordings of Mozart piano sonatas. We use the evaluation procedure used in [8], in which for each annotated beat the alignment error is the Manhattan distance

(in frames) to the closest point on the computed alignment. We compare DTW and NWTW, both on pairs of recording with and without structural differences. When structural differences occur, the alignment error for a given beat is the minimum error among the instances of that beat in the repetitions. Informally, the error criterion does not penalize the alignment for passing through one repetition of a section rather than through another.

4.2.1 Data

The recordings we use for this are from the same solo piano data set as the data described in Section 4.1.2, for which manual beat annotations are available (details on the annotation process can be found in [21]). We take pairs or recordings from this set such that each pair is a recording of the same piece by a different performer. Of these pairs, 74 are without structural differences. This set involves 6 performers, playing 41 movements from 20 sonatas. In addition, we take pairs of recordings with structural differences. This set consists of 133 pairs, and involves 8 performers, and 56 movements from 26 sonatas.

4.3 Results and discussion

Figure 3 shows the distance distributions between matching and non-matching audio, computed on the various data sets, using the various features. In general, the distributions vary more strongly across features than across the different types of audio. The pitch-oriented features with high frequency resolution (most notably MFCC50 / FFT.372s, and CQT) tend to be those with highest Jensen-Shannon divergence (JSD, shown in the plots). That said, the solo guitar data set in combination with the MFCC features shows a substantial reduction in JSD. This could be a consequence of the sensitivity of the MFCC features to the guitar tuning. Note also that this leads to a rightward shift of the optimal γ value, shown as dotted vertical lines in the plots. Although this discrepancy between optimal γ values across data sets is principally undesirable, the MFCC50 / FFT.372s feature still yields the highest JSD when over the joint data set (bottom row in Figure 3). For this reason, we use the MFCC50 / FFT.372s feature, and the corresponding optimal parameter value $\hat{\gamma} = 0.346$, for the subsequent quantitative evaluation of the DTW and NWTW methods.

The success of MFCC features for alignment is at odds with the findings of [11], even if they only evaluate the features indirectly through a retrieval task. An explanation for this may be the number of MFCC's selected: we obtain best results with 50 MFCC's, which is substantially more than the first 13 MFCC's typically used.

Table 1 shows the alignment accuracies for DTW and NWTW. When no structural differences occur between recordings, no jumps are required. In that case, the accuracy of NWTW alignment is very similar to that of DTW. When differences do occur, DTW tends to align parts of non-matching audio segments (as illustrated schematically in Figure 1), leading to higher alignment errors. The straight jumps that NWTW tends to make, ensure that the alignment path is always close to a matching position in either one or the other of a section that is repeated in only one of the recordings.

Error $\leq$ (ms)	0	20	40	60	80	100	200	500	1000
	alignment of performances *without* structural differences								
DTW	**47.1**	72.6	84.5	90.6	93.6	95.4	98.3	**99.6**	99.9
NWTW	46.3	**73.3**	**85.5**	**91.5**	**94.5**	**96.1**	**98.6**	**99.6**	**100.0**
	alignment of performances *with* structural differences								
DTW	37.0	60.6	73.1	80.2	83.6	85.6	89.2	91.5	92.9
NWTW	**38.1**	**66.0**	**79.5**	**86.7**	**90.1**	**91.7**	**94.5**	**96.4**	**97.3**

Table 1. Alignment accuracies for DTW and NWTW, for pairs of recordings without (top) and with (bottom) structural differences; The values represent the percentages of annotated beats with a Manhattan distance less or equal to the corresponding times in the top row; For example, using NWTW in structurally different audio, 96,4% of the beats are aligned no more than 500ms apart (91.5% for DTW)

5. CONCLUSIONS

In this paper we propose Needleman-Wunsch time warping (NWTW), a pure dynamic programming method to align music recordings that contain structural differences, and propose a way to estimate the optimal value for the gap penalty parameter γ. Experiments show that audio features with high frequency resolution allow for the most effective use of the gap penalty parameter. Moreover, a single value for γ is (close to) optimal for different types of music, including both solo instruments, and symphonic orchestra.

The advantage of our method over classical dynamic time warping is that it handles structural differences better, and the advantage over the original NW algorithm is that it handles tempo discrepancies between different recordings.

A limitation of the method in its current form is that it does not prefer jumps at the beginnings or ends of structural units over jumps at intermediate positions. Although this is not problematic for application scenarios that only require a matching position in one recording for each position in the other, jumps at intermediate positions in a structural unit are counter-intuitive from a musical point of view. We are currently investigating a further extension of the method to resolve this.

6. ACKNOWLEDGMENTS

This research is supported by the European Union Seventh Framework Programme FP7 / 2007-2013, through the PHENICX project (grant agreement no. 601166).

7. REFERENCES

[1] A. Arzt, G. Widmer, and S. Dixon. Automatic page turning for musicians via real-time machine listening. In *ECAI*, pages 241–245, 2008.

[2] A. Arzt, G. Widmer, and S. Dixon. Adaptive distance normalization for real-time music tracking. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 2689–2693. IEEE, 2012.

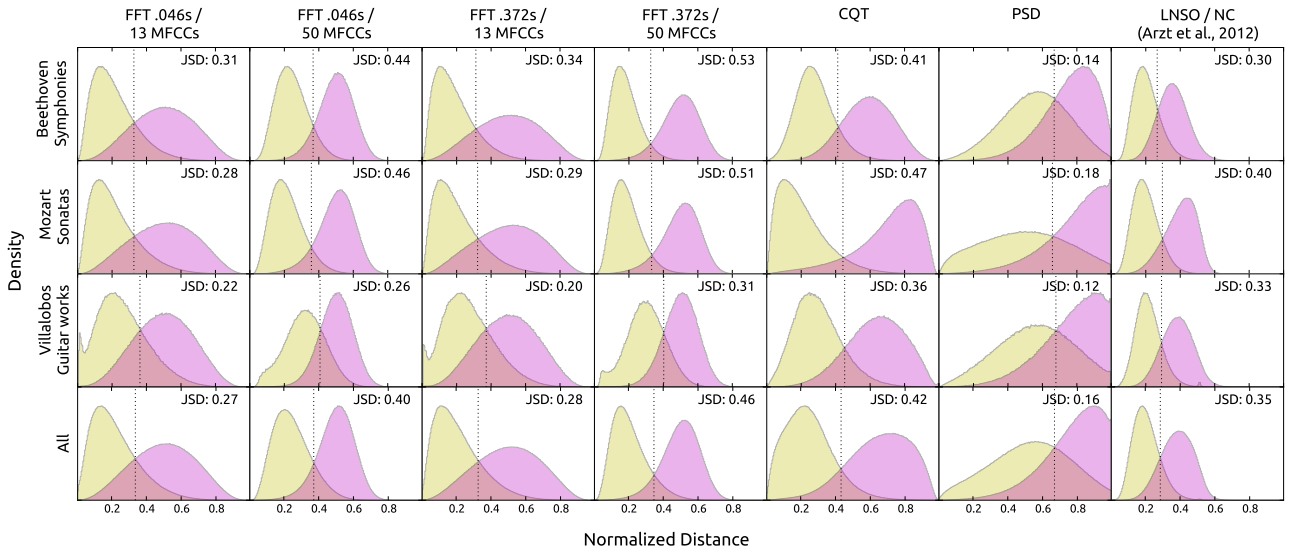


Figure 3. Distance distributions per feature for different types of music; In each subplot, the yellow distribution (left) is the distribution of distances between features of matching audio, the magenta distribution (right) is the distribution of distances between features of non-matching audio; The dashed vertical lines indicates the optimal value for γ; The value indicated with JSD is the Jensen-Shannon divergence between the two distributions

[3] J. P. Bello. Audio-based cover song retrieval using approximate chord sequences: Testing shifts, gaps, swaps and beats. In *Proc. of the 8th International Conference on Music Information Retrieval*, pages 239–244, Vienna, Austria, September 23-27 2007.

[4] 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*, 92:2698, 1992.

[5] R. Dannenberg. An on-line algorithm for real-time accompaniment. In *Proceedings of the 1984 International Computer Music Conference*. International Computer Music Association, 1984.

[6] R. B. Dannenberg and C. Raphael. Music score alignment and computer accompaniment. *Commun. ACM*, 49(8):38–43, 2006.

[7] C. Dittmar, K. Hildebrand, D. Gaertner, M. Winges, F. Muller, and P. Aichroth. Audio forensics meets music information retrieval; a toolbox for inspection of music plagiarism. In *Proc. of the 20th European Signal Processing Conference (EUSIPCO)*, pages 1249–1253, 2012.

[8] S. Dixon and G. Widmer. Match: A music alignment tool chest. In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, UK, September 11-15 2005.

[9] C. Fremerey, M. Müller, and M. Clausen. Handling repeats and jumps in score-performance synchronization. In *Proc. of the 11th International Society for Music Information Retrieval Conference*, pages 243–248, Utrecht, The Netherlands, August 9-13 2010.

[10] M. Grachten, J. L. Arcos, and R. López de Mántaras. A case based approach to expressivity-aware tempo transformation. *Machine Learning*, 65(2–3):411–437, 2006.

[11] N. Hu, R. B. Dannenberg, and G. Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pages 185–188. IEEE, 2003.

[12] B. Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the 1st International Conference on Music Information Retrieval*, Plymouth, Massachusetts, 2000.

[13] M. Mongeau and D. Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24:161–175, 1990.

[14] N. Montecchio and A. Cont. Accelerating the mixing phase in studio recording productions by automatic audio alignment. In *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 627–632, Miami (Florida), USA, 2011.

[15] M. Müller. *Information Retrieval for Music and Motion*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[16] M. Müller and D. Appelt. Path-constrained partial music synchronization. In *Proceedings of the 34th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 65–68, Las Vegas, Nevada, USA, Apr. 2008.

[17] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443 – 453, 1970.

[18] B. Pardo and W. Birmingham. Modeling form for on-line following of musical performances. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*, AAAI'05, pages 1018–1023. AAAI Press, 2005.

[19] 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*, 16:1138–1151, 2008.

[20] R. J. Turetsky and D. P. W. Ellis. Ground-truth transcriptions of real music from force-aligned midi syntheses. In *Proceedings of the Fourth International Conference on Music Information Retrieval*, Baltimore (Maryland), USA, 2003.

[21] G. Widmer, S. Dixon, W. Goebl, E. Pampalk, and A. Tobudic. In search of the Horowitz factor. *AI Magazine*, 24(3):111–130, 2003.

Author Index


CNPq
Conselho Nacional de Desenvolvimento
Científico e Tecnológico


CAPES


PANDORA®


Google™


SCIENTIA, VITA ET FIDES
PUCPR
GRUPO MARISTA


UFPR
UNIVERSIDADE FEDERAL DO PARANÁ


UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ismir

Curitiba • Brazil

4 | 8 de november 2013

Organization

PUCPR
GRUPO MARISTA

In Partnership with

CAPES

UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico