LARGE VOCABULARY AUTOMATIC CHORD ESTIMATION WITH AN EVEN CHANCE TRAINING SCHEME

Junqi Deng and Yu-Kwong Kwok
Department of Electrical and Electronic Engineering
The University of Hong Kong
{jqdeng,ykwok}@eee.hku.hk

ABSTRACT

This paper presents a large vocabulary automatic chord estimation system implemented using a bidirectional long short-term memory recurrent neural network trained with a skewed-class-aware scheme. This scheme gives the uncommon chord types much more exposure during the training process. The evaluation results indicate that: compared with a normal training scheme, the proposed scheme can boost the weighted chord symbol recalls of some uncommon chords and significantly improve the average chord quality accuracy, at the expense of the overall weighted chord symbol recall.

1. INTRODUCTION

Automatic chord estimation (ACE) is one of the central problems in music informatics. It asks for an algorithm to extract the harmonic progression within a piece of tonal music and label each harmony region with a chord symbol and a time stamp. For any artificial intelligence that is able to perform music analysis, an ACE algorithm will definitely be an important part of it.

For around two decades, ACE researches have been focusing around a very small vocabulary such as *major* and *minor* (or *majmin*) [1, 7, 16, 18, 22, 25, 30, 31]. Larger vocabularies are mostly only considered in some early works [12, 19, 26, 29]. Until recently, the large vocabulary issue has been brought back to the field [6, 9, 20], but except for the bass-treble chromagram proposed by Mauch and Dixon [21], the pre-segmented large vocabulary chord classification proposed by Deng and Kwok [8], and the Bayesian scaled likelihood estimation proposed by Humphrey [15], there is no technique specially designed for large vocabulary automatic chord estimation (LVACE).

Recently there has been a trend of using deep neural nets to solve ACE problems. Notable examples are: a convolutional neural network (CNN) based system [16], a hybrid fully connected neural network (FCNN) + recurrent neural network (RNN) system [3], a hybrid deep belief network (DBN) + RNN system [27], and a hybrid DBN + hidden Markov model (HMM) system [33]. They all show promising results comparable with or better than the state-of-the-art in terms of metrics that are based on *major* and *minor* triads. While last year there is a hybrid DBN + Gaussian-mixture-hidden-Markov-model (GMM-HMM) system [9] that tries to address the LVACE problem, it does not really pay special attention to the uncommon chords during the training process.

This paper, on the other hand, proposes a scheme that is dedicated to the uncommon and long-tail chords in the large vocabulary. The LVACE system is implemented with a standard feature extraction process and a bidirectional long short-term memory recurrent neural network (BLSTM-RNN) sequence decoder. Unlike the scaled likelihood estimation [15] that incorporates the prior distribution of chords into the estimation system, our large vocabulary strategy is to make sure each chord type has an uniform probability of being "seen" by the network at the start of each training case. This is called the "even chance" training scheme. Compared with a normal scheme that picks training cases at random, evaluation results show that the even chance training scheme can achieve much better uncommon weighted chord symbol recalls and significantly better average chord quality accuracy.

This paper is organized as follows: Section 2 elaborates on the LVACE system design; Section 3 describes the experimental setup, which contains the details of the proposed even chance training scheme; Section 4 reports and discusses the evaluation results; and finally Section 5 concludes the paper with the key findings and gives some possible future directions of LVACE.

2. THE LVACE SYSTEM

Figure 1 shows an overview of the LVACE system, which mainly contains a feature extraction module and a BLSTM-RNN sequence segmentation and classification module. In the following we will first elaborate on the feature extraction process, and then discuss the working mechanisms of the BLSTM-RNN.

2.1 Feature Extraction

The feature extraction process resembles the one described by Deng and Kwok [9]. It starts by resampling the raw

Attribution: Junqi Deng and Yu-Kwong Kwok. "Large Vocabulary Automatic Chord Estimation with an Even Chance Training Scheme", 18th International Society for Music Information Retrieval Conference, Suzhou, China, 2017.

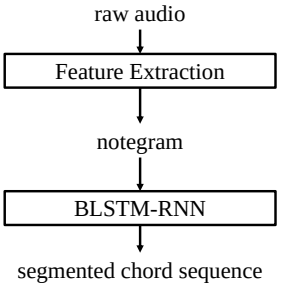

Figure 1. BLSTM-RNN LVACE system overview. The raw audio is transformed by a feature extraction process into a piece of notegram, and then decoded by a BLSTM-RNN into a segmented chord sequence.

audio input at 11025 Hz, which is followed by a short-time-Fourier-transform (STFT) with 4096-point Hamming window and 512-point hop size. It then proceeds to transform the linear-frequency spectrogram (2049-bin) to the log-frequency spectrogram (252-bin, 3 bins per semitone ranging from MIDI note 21 to 104) using the two cosine interpolation kernels proposed by Mauch [20]. The output at this step is a log-spectrogram $X_{k,m}$, where k is the index of frequency bins, and m is the index of time frames. The total number of frames is M, and the total number of bins in each spectrum is K (in this context K = 252).

The process then estimates the amount of deviation from standard tuning using the algorithm in [10], where the amount of detuning is estimated as:

$$\delta = \frac{wrap(-\varphi - 2\pi/3)}{2\pi}, \tag{1}$$

where $wrap$ is a function wrapping its input to $[-\pi, \pi)$ and φ is the phase angle at $2\pi/3$ of the discrete-Fourier-transform (DFT) of $\sum_m X_{k,m}/M$. The tuning frequency τ is then given by:

$$\tau = 440 \cdot 2^{\delta/12}, \tag{2}$$

and the original tuning is thus updated by interpolating the original spectrogram $X_{k,\cdot}$ at $X_{k+p,\cdot}$, where:

$$p = (\log(\tau/440)/\log(2)) \times 36, \tag{3}$$

since there are 36 bins per octave (3 bins per semitone) in $X_{k,\cdot}$. The interpolation results will update the original $X_{k,m}$, and the new $X_{k,m}$ spectrogram will be referred to as "notegram", which will be the input feature of the BLSTM-RNN sequence decoder.

2.2 Recurrent Neural Network

An RNN is a neural network with cyclical connections, so that the network can be recurrently unrolled into multiple frames [11, 17]. It can be used to model the conditional probability of an output sequence $Y(Y^1, Y^2, ...)$ given an input sequence $X(X^1, X^2, ...)$, where the superscripts denote time steps. With a forward hidden layer, it models this relationship in a sequential manner, so that every output frame Y^t is conditioned on not only the current input frame X^t but also all previous input frames $X^{1:t}$. Besides, if a backward hidden layer is added to the model, Y^t will be conditioned on the whole input sequence X. This modified network is called *bidirectional recurrent neural network* (BRNN).

When the training sequence is long, the learning signal may die down gradually via the back-propagation-through-time (BPTT) [24]. This *gradient vanishing* phenomenon [2] often makes the training ineffective or unsuccessful. Using LSTM [13] units instead of normal non-linearities within a (B)RNN is a useful way to circumvent this undesirable effect.

2.3 BLSTM-RNN Architecture

The proposed LVACE system uses a BRNN with LSTM units, or a BLSTM-RNN. It has a forward and a backward hidden layer both with 800 LSTM units. The input layer has 252 real-value nodes, connected to a notegram spectrum. The output layer is a *#-chord-way* softmax layer. In this implementation, we use a typical LSTM configuration, that all LSTM gates employ sigmoid activations, and that both the LSTM cell and the LSTM output use hyperbolic tangent activations. Note that this network is different from the one in [9], in that this BLSTM-RNN could take a variable length input sequence and generate multiple outputs, but the other one is designed to handle a fixed length of input with a single softmax regression output.

3. EXPERIMENTAL SETUP

This section describes the vocabulary, datasets and training-validation schemes used in the experiments.

3.1 Vocabulary

The large vocabulary supported by the proposed system is the *SeventhsBass* introduced in MIREX ACE 2013 [1] . It contains the "*NC*" chord [2] , all *maj* and *min* triads, all *maj7*, *min7*, and *7* chords, and all of their inversions.

3.2 Datasets and Data Augmentation

Six datasets of 546 tracks are used during the experiments. They contain both eastern and western pop/rock songs. They are: 20 tracks from the Chinese pop song dataset (CNPop20, or C) [3] ; 29 tracks from the JayChou dataset (JayChou29, or J) [3] ; 26 tracks from the Carole King + Queen dataset (K) dataset [4] ; 191 songs from the USPop dataset (U) [5] ; 100 tracks from the RWC dataset (R) [6] ; and 180 tracks from the TheBeatles180 (B) dataset [14]. The combination of datasets is notated by concatenating their

[1] music information retrieval exchange:`http://www.music-ir.org/mirex/wiki/MIREX\_HOME`

[2] means "not a chord", or "no chord"

[3] `http://tangkk.net/label/`

[4] `http://isophonics.net/datasets`

[5] `https://github.com/tmc323/Chord-Annotations`

[6] `https://staff.aist.go.jp/m.goto/RWC-MDB/`

letter codes. For example, a combination of all datasets is denoted as "CJKURB".

To generate the training data, all raw audios are transformed to notegram representations. The original segment-wise ground truth annotations are upsampled to become frame-wise annotations with 1-to-1 mappings to the notegram frames. Due to the absence of phase information in notegram, all data can be transposed to 12 keys to yield 12 times the original amount of data [16].

3.3 Training and Cross-validation

Two different training schemes are used. The only difference between them are the way of choosing training cases at each iteration:

- completely random (CR): a random training case is chosen.
- even chance (EC): a training case starting with a certain chord type is chosen, and each chord type has an even chance to be chosen as the start.

The EC training scheme is inspired by the skewed class sensitive training methods [5]. Considering a skewed distribution of chords in the training set [4], a random sampling scheme like CR will inevitably draw samples based on that same distribution, which causes lack of exposure of uncommon chords. The EC scheme, however, gives each uncommon chord much more exposure during the training process. Concretely, the EC scheme is formalized as follows in Algorithm 1:

Algorithm 1 EvenChanceTraining

Require: training data set - (X, y); number of chord classes - $nclass$; early stopping flag - es.

od = BalancedOrderedDict(y, $nclass$)
$iter$ = 0
while not early-stopping **do**
  if mod($iter$, $nclass$) is 0 **then**
    $coidx$ = random_shuffle(0:$nclass$-1)
  $tclist$ = $od(coidx_{mod(iter,nclass)})$
  draw a random item e from $tclist$
  update network with $(X, y)_e$
  $iter$++

The core of this procedure is the "*BalancedOrderedDict*" which generates a dictionary of *(track index, chord change position)* tuples indexed by chord classes. It is formalized in Algorithm 2, where each entry of od contains a list of *(track index, chord change position)* tuples.

It should be pointed out that, besides chord classification, the BLSTM-RNN has to also perform segmentation, which means the training samples have to contain chord segmentation boundaries for the network to learn from. As a result, we set the length of each training case to be 500 frames, which contains multiple chords. Because of this, there is still uneven distribution of common and uncommon chords during the training process. The EC scheme

Algorithm 2 BalancedOrderedDict

Require: labels of training data set - y; number of chord classes - $nclass$.

for each class i from 0 to $nclass - 1$ **do**
  initialize an empty list $od[i]$
for each track index j in y **do**
  for each frame poistion k in $y[j]$ **do**
    if k is a chord change position **then**
      append (j,k) to $od[y[j][k]]$
return od

can guarantee a uniform chord distribution at the start of each training case, but it does not try to alter the sampling of the other chords. In effect, it only boosts the exposure of uncommon chords to a certain level, but could not make the chances of common and uncommon chords totally even.

The following describes the remaining training procedures that apply throughout the experiments. We try to report the precise settings of every parameter so that the readers may reproduce the results:

- Each training case contains 500 frames of audio content with ground truth labels;
- The network update signal is computed by an Adadelta optimizer [32];
- The training is regularized with dropout [28] and early-stopping [23];
- All dropout probabilities are set to 0.5;
- All early-stopping criteria are monitored using the validation error of the CNPop20 dataset, which is not in any cross-validation set; The validation cycle is 100 iterations;
- The model with the lowest validation loss will be saved; If the current validation loss is smaller than 0.996 of the best one, the early-stopping patience will increase by 0.3 times the current number of iterations;
- Training stops when the early-stopping patience is less than the current number of iterations.

For evaluation, five-fold cross-validation (CV) is performed throughout all experiments. Each fold is a combination of approximately 1/5 tracks of each dataset. Every model is trained on four folds and cross-validated on the remaining fold, resulting in a total number of five validation scores, the average of which will be the final scores to be reported in Section 4. For this research to be reproducible, all implementation details are made available online [7].

4. RESULTS AND DISCUSSIONS

Throughout this section, we use the MIREX ACE standard evaluation metric, "weighted chord symbol recall"

[7] https://github.com/tangkk/tangkk-mirex-ace

(*WCSR*), to report system performances. The "chord symbol recall" (*CSR*) is defined as follows:

$$CSR = \frac{|S \cap S^*|}{|S^*|}, \tag{4}$$

where S and S^* represents the automatic estimated segments, and ground truth annotated segments, respectively, and the intersection of S and S^* is the part where they overlap and have equal chord annotations. *WCSR* is the weighted average of all tracks' *CSR*s by the lengths of these tracks:

$$WCSR = \frac{\sum Length(Track_i) * CSR_i}{\sum Length(Track_i)}, \tag{5}$$

where the subscript i denotes the i^{th} track. Likewise, the *WCSR* of a specific chord type is:

$$WCSR_C = \frac{\sum Length(C_i) * CSR_i}{\sum Length(C_i)}, \tag{6}$$

where the subscript i denotes the i^{th} instance of chord C within the data set.

To measure the balanced performance of a system, we report "average chord quality accuracy" (*ACQA*) [6]:

$$ACQA = \frac{\sum WCSR_C}{\#\,of\ chords}. \tag{7}$$

which sums up the *WCSR*s of all chord types in the vocabulary. Systems that over-fit a few chord types or neglect uncommon chords tend to get lower *ACQA*s, while those well balanced systems will have higher *ACQA*s.

The original scores in this section are computed using the MusOOEvaluator [8] .

4.1 Sevenths, Inversions and ACQA

Table 1 shows the comparison between CR and EC training schemes on some uncommon (*non-majmin*) [4] chords' *WCSR*s as well as the *ACQA*. The six chord types in the table are chosen because they have relatively more weights in pop/rock songs than the more long-tail ones such as *min/5* and *min/b3*. Note that *maj/5* and *maj/3* are also included in two other large vocabularies proposed by Mauch [20] and Cho [6].

	maj7	7	min7	maj/5	maj/3	7/b7	ACQA
CR	7.3	6.6	24.1	4.5	24.5	0.0	10.8
EC	14.6	9.9	30.9	12.0	32.4	7.8	13.2

Table 1. Comparison between CR and EC: seventh chords, inversions and ACQA scores; Dataset: JKURB

The results show that EC outscores CR in all categories, some of which by very large amount such as *maj/5* and *maj/3*. Although not all chord types' results are shown, the *ACQA* results suggest that the EC training scheme could lead to a much more balanced LVACE system under a skewed class distribution.

[8] https://github.com/jpauwels/MusOOEvaluator

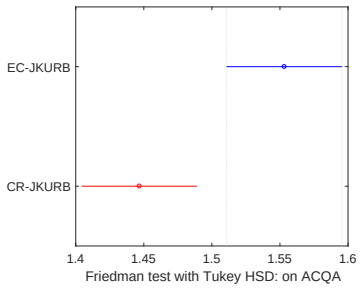

Figure 2. Multiple comparison test on ACQAs

We perform a Friedman test on the track-wise *ACQA* results of both systems. After that we use the Tukey HSD (honest significant difference) to perform a multiple comparison test on the Friedman test's statistics with a significance level of 0.05. The results as shown in Figure 2 confirm that EC is significantly better than CR in *ACQA*.

4.2 Major, Minor and WCSR

The EC trained system has a more balanced performance than the CR's, however, it scarifies common chords' *WCSR*s. Table 2 shows the comparison between CR and EC on some common (*majmin*) [4] chords' *WCSR*s as well as on the overall SeventhsBass *WCSR*.

	maj	min	WCSR
CR	74.2	52.2	52.0
EC	67.8	51.4	50.6

Table 2. Comparison between CR and EC: major, minor and WCSR scores; Dataset: JKURB

Although the two schemes have very close scores on *min*, there is a large difference in *maj*. Due to the dominantly large weight of *maj* chords in the JKURB dataset combination, it eventually leads to CR's much higher *WCSR*, despite EC performs better in most of the other chord types. CR's much higher *maj WCSR* is not unexpected: since it draws each training case at random, the probability that each chord type gets "seen" by the neural net is subject to the distribution of chord types in the training dataset, and therefore the *maj* chords are "learned" much more than the other chords.

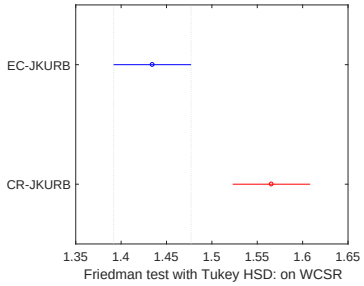

Figure 3. Multiple comparison test on WCSRs

We perform a Friedman test on the track-wise *WCSR* results of both systems. After that we use the Tukey HSD to perform a multiple comparison test on the Friedman test's statistics with a significance level of 0.05. The results as shown in Figure 3 confirm that CR is significantly better than EC in *WCSR*.

4.3 On Different Datasets

For more convincing comparison results, the same experiment is run 4 times using different dataset combinations. Table 3 shows the results of JK, JKU, JKUR and JKURB. We only report the *WCSR* and *ACQA* for brevity.

	CR-WCSR	EC-WCSR	CR-ACQA	EC-ACQA
JK	46.4	46.4	13.5	15.5
JKU	50.4	49.1	11.2	13.5
JKUR	50.1	49.6	12.8	14.5
JKURB	52.0	50.6	10.8	13.2

Table 3. Comparison between CR and EC: WCSR and ACQA on different datasets.

In all these experiments, the EC systems get higher *ACQA*s, but lower or equal *WCSR*s, than the CR systems. It is sufficient to say that EC is better at training a balanced performing LVACE system under skewed class distribution, while CR is better at training an LVACE system with higher overall performance.

For both training schemes, the increment of training data will lead to the increase of *WCSR*, but the same thing does not happen in *ACQA*. Assuming that every dataset contains a certain amount of noise (i.e., mis-labeled or mis-segmented chord regions), this observation could be tentatively explained as follows. *WCSR* is mostly relying on the quality of *majmin* chord labels, which are on average easier to be labeled. Therefore the increment of data will also increase the *WCSR* score. *ACQA*, however, is mostly relying on the quality of *non-majmin* chord labels, which are on average more difficult to be labeled. Therefore the increment of data could not guarantee the increase of *ACQA* score, since it is hard to guarantee the proportion of *non-majmin* noise in the incremental data is smaller than those of the original data.

5. CONCLUSIONS

This paper presents a BLSTM-RNN based LVACE system, trained using a skewed class oriented "even chance" scheme. This scheme is compared with a more intuitive "completely random" scheme that chooses training case randomly at each iteration. Evaluation results demonstrate that the EC training scheme is superior in both the uncommon (*non-majmin*) chords' *WCSR*s and the *ACQA*, at the expense of the common (*majmin*) chords' *WCSR*s and the overall *WCSR*.

A successful LVACE system is marked by both high *WCSR* and high *ACQA*, because human chord recognition experts are able to achieve both of them. The EC training scheme is a technique to improve a system's *ACQA*, thus it is a valuable approach to consider when we design an LVACE system in the future.

The fundamental driving force of LVACE research should be the ground-truth data and their qualities, especially the qualities of the uncommon or long-tail chords. As we see in the discussion above, *ACQA* is very vulnerable to uncommon chords' quality. Therefore, it might be possible that in the future as we gradually increase the amount of ground-truth data, we could use *ACQA* in a way to perform sanity check on the quality of the incremental data.

6. REFERENCES

[1] Juan Pablo Bello and Jeremy Pickens. A robust mid-level representation for harmonic content in music signals. In *Proceedings of the 6th International Society for Music Information Retrieval Conference, ISMIR*, volume 5, pages 304–311, 2005.

[2] Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[3] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Audio chord recognition with recurrent neural networks. In *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR*, pages 335–340, 2013.

[4] John Ashley Burgoyne, Jonathan Wild, and Ichiro Fujinaga. An expert ground truth set for audio chord recognition and music analysis. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR*, volume 11, pages 633–638, 2011.

[5] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Editorial: special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1):1–6, 2004.

[6] Taemin Cho. *Improved techniques for automatic chord recognition from music audio signals*. PhD thesis, New York University, 2014.

[7] Taemin Cho, Ron J Weiss, and Juan Pablo Bello. Exploring common variations in state of the art chord recognition systems. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 1–8, 2010.

[8] Junqi Deng and Yu-Kwong Kwok. Automatic chord estimation on SeventhsBass chord vocabulary using deep neural network. In *Proceedings of the 41th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[9] Junqi Deng and Yu-Kwong Kwok. A hybrid gaussian-HMM-deep-learning approach for automatic chord estimation with very large vocabulary. In *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR*, 2016.

[10] Karin Dressler and Sebastian Streich. Tuning frequency estimation using circular statistics. In *Proceedings of the 8th International Society for Music Information Retrieval Conference, ISMIR 2007*, pages 357–360, 2007.

[11] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[12] Takuya Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proceedings of the 25th International Computer Music Conference*, volume 1999, pages 464–467, 1999.

[13] Alex Graves. Supervised sequence labelling. 2012.

[14] Christopher Harte. *Towards automatic extraction of harmony information from music signals*. PhD thesis, Department of Electronic Engineering, Queen Mary, University of London, 2010.

[15] Eric Humphrey. *An Exploration of Deep Learning in Content-based Music Informatics.* PhD thesis, New York University, 2015.

[16] Eric J Humphrey and Juan P Bello. Rethinking automatic chord recognition with convolutional neural networks. In *Proceedings of the 11th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 357–362. IEEE, 2012.

[17] Michael I Jordan. Attractor dynamics and parallellism in a connectionist sequential machine. In *Proceedings of the Eighth Annual Meeting of the Cognitive Science Society*, pages 531–546. Lawrence Erlbaum Associates, 1986.

[18] Maksim Khadkevich and Maurizio Omologo. Use of hidden Markov models and factored language models for automatic chord recognition. In *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009*, pages 561–566, 2009.

[19] Namunu C Maddage, Changsheng Xu, Mohan S Kankanhalli, and Xi Shao. Content-based music structure analysis with applications to music semantics understanding. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 112–119. ACM, 2004.

[20] Matthias Mauch. *Automatic chord transcription from audio using computational models of musical context*. PhD thesis, School of Electronic Engineering and Computer Science Queen Mary, University of London, 2010.

[21] Matthias Mauch and Simon Dixon. Approximate note transcription for the improved identification of difficult chords. In *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR*, pages 135–140, 2010.

[22] Yizhao Ni, Matt McVicar, Raul Santos-Rodriguez, and Tijl De Bie. An end-to-end machine learning system for harmonic analysis of music. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1771–1783, 2012.

[23] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 1998.

[24] David E Rumelhart, James L McClelland, PDP Research Group, et al. *Parallel distributed processing*, volume 1. IEEE, 1988.

[25] Matti P Ryynänen and Anssi P Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, 2008.

[26] Alexander Sheh and Daniel PW Ellis. Chord segmentation and recognition using EM-trained hidden Markov models. In *Proceedings of the 4th International Society for Music Information Retrieval Conference, ISMIR*, pages 185–191. International Symposium on Music Information Retrieval, 2003.

[27] Siddharth Sigtia, Nicolas Boulanger-Lewandowski, and Simon Dixon. Audio chord recognition with a hybrid recurrent neural network. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[28] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[29] Borching Su and Shyh-Kang Jeng. Multi-timbre chord classification using wavelet transform and self-organized map neural networks. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Citeseer, 2001.

[30] Jan Weil and Jean-Louis Durrieu. An HMM-based audio chord detection system: Attenuating the main melody. *The Music Information Retrieval Exchange*, 2008.

[31] Adrian Weller, Daniel Ellis, and Tony Jebara. Structured prediction models for chord transcription of music audio. In *International Conference on Machine Learning and Applications, ICMLA*, pages 590–595. IEEE, 2009.

[32] Matthew D Zeiler. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[33] Xinquan Zhou and Alexander Lerch. Chored detection using deep learning. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR*, volume 53, 2015.