A HYBRID GAUSSIAN-HMM-DEEP-LEARNING APPROACH FOR AUTOMATIC CHORD ESTIMATION WITH VERY LARGE VOCABULARY

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

ABSTRACT

We propose a hybrid Gaussian-HMM-Deep-Learning approach for automatic chord estimation with very large chord vocabulary. The Gaussian-HMM part is similar to Chordino, which is used as a segmentation engine to divide input audio into note spectrogram segments. Two types of deep learning models are proposed to classify these segments into chord labels, which are then connected as chord sequences. Two sets of evaluations are conducted with two large chord vocabularies. The first evaluation is conducted in a recent MIREX standard way. Results show that our approach has obvious advantage over the state-of-the-art large-vocabulary-with-inversions supportable ACE system in terms of large vocabularies, although is outperformed by in small vocabularies. Through analyzing and deducing system behaviors behind the results, we see interesting chord confusion patterns made by different systems, which conceivably point to a demand of more balanced and consistent annotated datasets for training and testing. The second evaluation preliminarily demonstrates our approach's superiority on a jazz chord vocabulary with 36 chord types, compared with a Chordino-like Gaussian-HMM baseline system with augmented vocabulary capacity.

1. INTRODUCTION

Automatic chord estimation (ACE) is currently undergoing a paradigm shift from Gaussian-HMM (Hidden Markov Model) approaches to deep learning approaches. Recently, there have been quite a few deep learning powered ACE approaches in the field, including a convolutional neural network (CNN) approach [10], a hybrid feedforward-recurrent neural network (DNN-RNN) approach [3], a deep belief network (DBN) approach [19], and a hybrid DBN-RNN approach [16]. Some are more purely deep learning oriented, which only apply minimal amount of feature extractions, while others consider combination of traditional signal processing techniques and deep learning.

One common point of these approaches is that they are all evaluated under major/minor vocabulary (MajMin), which is far from reflecting the complexity of chord vocabulary in pop/rock music practice. In 2013, MIREX ACE has introduced a new evaluation scheme [14] focusing on much more complicated chord vocabulary, the "SeventhsBass", which includes MajMin, three types of their seventh chords, and all of their inversions. The SeventhsBass, although also omitting some rare chords in pop/rock practice, is much closer to the reality compared with MajMin. It differentiates among triads, sevenths and their inversions because they all have different harmonic qualities. It is not only important for ACE systems to be evaluated on more complex chord vocabulary, but also to actually support that vocabulary. Unfortunately from 2013 to 2015, there have been only two systems that actually support SeventhsBass [6], others mostly do not even support chord inversions. Not being able to generate inversions is musically problematic since in some musical context they have very different harmonic qualities from their root positions. As shown in Figure 1, for example, the chord inversions serve as a diatonic or chromatic continuations of the bass line. If some of these are replaced by their root positions, the continuations are broken and thus the pieces will sound very different.

1) | G | D/F# | F | C/E | Cm/Eb |
2) | A | Bm | A/C# | D |
3) | C | G/B | Am | Am/G | F | C/E |
4) | C | F | C/E | D/F# | E/G# | F#/A# | Bm7 | C# |

Figure 1. Four chord progressions that contain bass line continuations which demand chord inversions. Progressions like 1,2 and 3 are very popular among pop/rock. Progression 4 induces a key shift from C major to F# minor.

Following the above argument, we propose an ACE system that not only supports but also be evaluated on SeventhsBass. This system uses a Chordino-like module [13] as a chord segmentation engine, and classifies chords within each segment using a deep learning model. Evaluation results show that the best system variants have obvious advantage over the state-of-the-art SeventhsBass supportable ACE system in terms of Sevenths (MajMin + maj7, min7, 7) and SeventhsBass.

Besides, we also try the proposed approach on a jazz vocabulary. The comparison target remains the same except for an augmentation of its chord vocabulary capacity. Since the standard evaluation tool [14] does not apply for this vocabulary, evaluation is done manually via comparison of weighted chord symbol recall [1] . Results show similar ranking as in the SeventhsBass' results, and still the best system significantly outscores the baseline approach.

The rest of this paper is organized as follows: Section 2 gives an overview of the proposed ACE system framework and its workflow; Section 3 elaborates the implementations of two deep learning based models (DBN and BLSTM-RNN); Section 4 reports both SeventhsBass and jazz vocabulary evaluation results, with a detailed discussion of chord confusion and how they affect systems' performances; Section 5 concludes the paper and puts forward some possible future considerations in ACE.

2. SYSTEM OVERVIEW

The proposed ACE approach [2] has a simple workflow as shown in Figure 2. The test data goes through a Chordino-like module for segmentation. Then each note spectrogram (referred to as "notegram" below) segment will be classified using a deep learning model. The output chord sequence is obtained by connecting the classified labels.

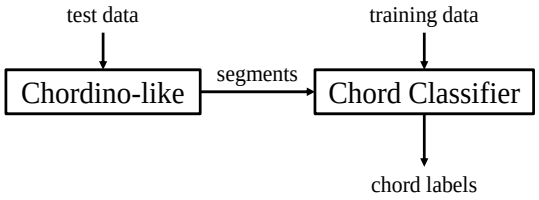

Figure 2. System overview. The audio input (test data) goes through a Chordino-like process for segmentation, then the segments are classified into chord labels.

The Chordino-like module is implemented according to the algorithmic description of Chordino [12, 13]. The audio input is first resampled at 11025 Hz, and transformed by a 4096-point Hamming window short-time-Fourier-transform (STFT) with 512 point hop size. The linear-scale spectrogram is then mapped to a log-scale spectrogram, or notegram. After standard tuning (tuned notegram) and feature scaling, note activation patterns are extracted from the notegram via non-negative-least-square (NNLS) method. A piece of chromagram is derived by bass-treble profiling of the note activation patterns. The chromagram is then decoded and segmented by a Gaussian-HMM with very high self-transition weights.

The chord classifier is implemented using deep learning models, which will be discussed in the following section. Applying different deep learning models leads to different system variants out of the proposed framework. In the following, we refer to these "variants" as "systems", and the framework as the "approach".

[1] http://www.music-ir.org/mirex/wiki/2013:Audio_Chord_Estimation

[2] the full implementation of this ACE system is accessible via: https://github.com/tangkk/tangkk-mirex-ace

3. DEEP LEARNING MODELS

We consider two types of deep learning models. They both have input at the tuned notegram level. The deep neural network will learn the rest of the transformations from tuned notegram all the way to chord label. Since there are different numbers of frames in different chord segments, in order to use a fixed-length input structure, we conducted a preliminary study and found that 6 sub-segments are good for single chord classification task. Note that the number of sub-segments should at least reflect the temporal order of bass line in order to differentiate root position from inversions. Thus we compute a 6-frame notegram for each segment as follows: at first the segment is divided into 6 equal-size sub-segments; if the total number of frames is not divisible by 6, the last frame is extended several times to make it divisible; then notegram in each sub-segment is averaged over time, resulting in one frame per sub-segment.

3.1 DBN Model

We first consider a DBN model. It contains two hidden layers, each of 800 neurons. The input layer is of 6×252-dimension (252 is the size of a notegram frame), and the output layer is a #chord-way softmax layer. The neurons of both input and output layers are of Gaussian type (real value from 0 to 1). The neurons in both hidden layers are of Bernoulli type (binary value 0 or 1).

During unsupervised pre-training, the first restricted-Boltzmann-machine (RBM) formed by the first two layers is considered as a Gaussian-Bernoulli RBM, and the second RBM formed by the two hidden layers is considered as a Bernoulli-Bernoulli RBM. The pre-training is conducted using persistent-contrastive-divergence-10 [17] (PCD-10), for 100 epochs with learning rate 0.001. During supervised fine-tuning, the network connections are updated using mini-batch stochastic gradient descent, and the updates are regularized by dropout [8] (with 0.5 dropout probability) and early-stopping. The stopping criteria is monitored by a validation set, which randomly contains 20% of the training set. The other 80% are used for computing the gradients. Due to the randomness of train/validation split, we repeatedly train 6 models. The model with the best validation score will be saved for testing.

For comparison, we also consider a feed-forward multilayer perceptron (MLP) model, whose network configuration is the same as the DBN, but trained using only the fine-tuning procedure described above.

3.2 BLSTM-RNN Model

Historically, long-short-term-memory (LSTM) [9] unit is introduced to try to solve the gradient vanishing problem [2] when training a recurrent neural network with a long sequence of examples.

3.2.1 LSTM Unit

Instead of having only one input port, an LSTM unit has four input ports. As shown in Figure 3, three of them are used for gating purpose, and the other is used for normal purpose. Each gate computes an output gating signal from the weighted sum of its inputs using a non-linear activation function. The gating signal computed by input gate, output gate and forget gate will interact with both the LSTM unit's input value and the LSTM cell value through simple multiplications, resulting in the LSTM unit's output value. Input gate regulates the amount of input feeding into the cell; forget gate regulates the current cell value by the previous cell value; and output gate regulates the amount of output by interacting with the current cell value. Since all functions involved in an LSTM unit are differentiable or partially differentiable, all connections can be trained using the same back-propagation-through-time (BPTT) [7] technique as used in training a normal RNN.

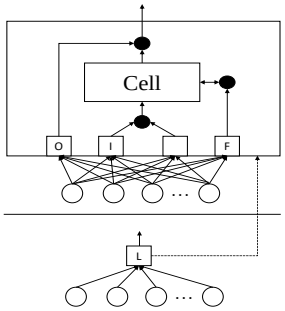

Figure 3. LSTM unit. O = output gate; I = input gate; F = forget gate; Black dots indicate multiplication operations

3.2.2 BLSTM-RNN

We then consider a BLSTM-RNN model [7] as shown in Figure 4. It has both forward and backward LSTM layers, each of which has 800 LSTM units. Before the #chord-way softmax output layer, it performs mean pooling to summarize results from all frames. During training, the RNN is always unrolled to 6 frames, and the weights are updated via BPTT using AdaDelta algorithm [18], regularized with dropout (with 0.5 dropout probability) and early-stopping, monitored by a validation set chosen in the same way as in DBN's case. Due to the randomness of train/validation split, we repeatedly train 6 models. The model with the best validation score will be saved for testing.

4. EVALUATION

For SeventhsBass ACE implementation, four datasets of 266 tracks in total are used in training. They contain both eastern and western pop/rock songs. They are: 1, JayChou29 dataset [5]; 2, a Chinese pop song dataset (CNPop20) [3]; 3, Carole King + Queen dataset

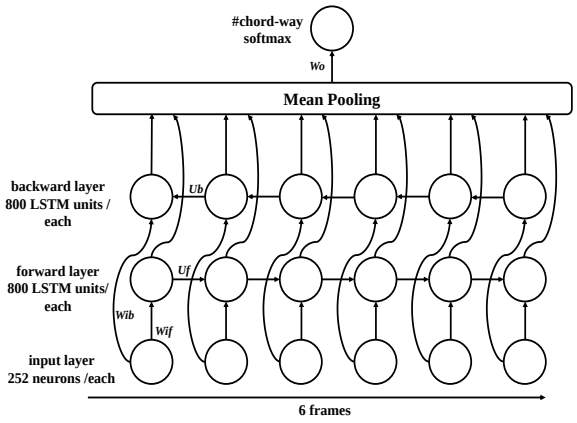

Figure 4. Bidirectional-long-short-term-memory recurrent neural network (BLSTM-RNN) used in the proposed approach

(KingQueen26) [4]; 4, 191 songs from USPop dataset (U) [5]. In order to see the effect of data size, all models will be incrementally trained on: 1, JayChou29 and CNPop20 (CJ); 2, CJ + KingQueen26 (CJK); 3, all four datasets (CJKU).

For Jazz ACE implementation, 99 pieces of jazz chord comping + soloing dataset extracted from a jazz guitar book [15] (JazzGuitar99) are used as training/validation dataset, and 7 pieces from Gary Burton's online course [1] (GaryBurton7) are used as test dataset. JazzGuitar99's annotations are taken directly from the book, and GaryBurton7's annotations are taken from the leadsheets provided along with the course. The jazz chord vocabulary contains 36 types [6]. Note that inversions are not considered in this preliminary jazz ACE study because: 1. there are very few inversion notations in the currently used datasets; 2. it results in huge number of classes based on these 36 types.

All training data are to be used at their tuned notegram level, which does not contain phase information. Assuming well temperament, we can augment all training data by pitch shifting their notegrams to all 12 keys with zero padding. Adjusting the chord labels accordingly, this results in 12 times of training data.

4.1 SeventhsBass Vocabulary Systems Evaluation

SeventhsBass evaluation is conducted in a MIREX standard way. We use TheBeatles180 (B) as the test set and run end-to-end automatic chord transcriptions from raw audio to chord progression for every track within. The metric score is computed in a weighted chord symbol recall (WCSR) way using the MIREX ACE evaluation tool [14]. All systems are compared with each other and compared with Chordino. Chordino is the only other suitable system

[3] containing 20 songs from both male and female singer-songwriters from Chinese cultural backgrounds including mainland China, Hong Kong and Taiwan

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

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

[6] They are: maj, min, min6, 6, maj7, maj7#5, maj7#11, maj7b5, min7, minmaj7, min7b5, min7#5, 7, 7b5, 7b9, 7#9, 7#5#9, 7#5b9, 7b5b9, 7#5, 7sus4, aug7, dim7, maj9, min9, 9, 9#11, min11, min11b5, 11, min13, maj13, 13, 13b9, 69 and N

	Mm	MmB	S	**SB**
Chordino	**74.30**	**71.40**	52.99	50.60
CJ-MLP	67.25	62.27	55.15	50.86
CJ-DBN	70.68	66.52	**58.23**	**54.71**
CJ-BLSTM	69.09	64.51	56.47	52.74
CJK-MLP	65.18	63.12	53.82	52.00
CJK-DBN	67.44	65.56	55.64	54.03
CJK-BLSTM	70.46	68.56	**59.11**	**57.50**
CJKU-MLP	67.95	65.87	55.98	54.09
CJKU-DBN	68.53	66.49	56.19	54.37
CJKU-BLSTM	72.62	70.47	**59.37**	**57.47**

Table 1. WCSRs of four main MIREX ACE vocabulary (Mm = MajMin, MmB = MajMinBass, S = Sevenths, SB = SeventhsBass; CJ = JayChou29 + CNPop20; CJK = CJ + KingQueen26; CJKU = CJK + USPop191)

for comparison because this is the only publicly available system which also supports SeventhsBass vocabulary [4].

Here we argue for the validity of our evaluation methodology. Note that our systems are trained with combination of C, J, K, U, and tested on B. Some may challenge that since these two sets may be drawn from two different chord populations (NOT in terms of chord types, but chord rendering styles), thus the test results may not reflect the true system performance. It is true that they contain different distributions of chord rendering styles, especially in terms of the "dominant sevenths" chord, as also reflected in the results and discussions in Section 4.1.2. But as we will see in the results, in general, the C,J,K,U-trained systems generalize very well on B. In fact, since there could be countless of possible chord rendering styles of each chord, it is difficult to "make sure" that two datasets are drawn from the "same" population, not to mention that it is even more difficult to define the possible "properties" of such "population". An average k-fold cross-validation score could be a better indication of system performance in terms of the combined CJKUB training/test set, but neither is this a standard benchmarking method, nor can this score be directly compared with an expert system such as Chordino.

4.1.1 Overall Results of SeventhsBass

The WCSRs of four main MIREX ACE vocabularies are shown in Table 1. In MajMin and MajMinBass, Chordino still does the best among all systems. But in Sevenths and SeventhsBass (the main focus in this paper), all our systems perform better than Chordino, with CJ-DBN, CJK-BLSTM and CJKU-BLSTM performing best.

Let's take CJ-DBN as representative for the moment. It seems that it performs better at recognizing seventh chords but worse at inversions compared with Chordino, but this is not a correct deduction from the table. Note that Sevenths is a collapse of chords, regardless of root positions or inversions, to their maj, min, maj7, min7 or 7 forms; and MajMinBass is a collapse of chords, regardless of tetrads or triads, to their maj, min, maj/3, maj/5, min/b3 or min/5 forms. Considering SeventhsBass as all chords in their original forms, the score boost from SeventhsBass to Sevenths indicates the amount of confusion between root positions and inversions (let's call it "bass confusion"); the score boost from SeventhsBass to MajMinBass indicates

	maj	min	maj/3	maj/5	min/b3	min/5
maj (r)	0.66	0.03	0.00	0.02	0.00	0.00
min (r)	0.10	0.60	0.00	0.01	0.00	0.00
maj/3 (r)	0.35	0.13	0.19	0.00	0.00	0.00
maj/5 (r)	0.50	0.08	0.00	0.23	0.00	0.00
min/b3 (r)	0.36	0.30	0.01	0.06	0.00	0.00
min/5 (r)	0.19	0.55	0.04	0.04	0.00	0.00

Table 3. Chordino's bass confusion matrix.

	maj	min	maj/3	maj/5	min/b3	min/5
maj (r)	0.72	0.06	0.03	0.03	0.00	0.00
min (r)	0.15	0.63	0.02	0.02	0.00	0.00
maj/3 (r)	0.34	0.28	0.23	0.01	0.00	0.02
maj/5 (r)	0.49	0.11	0.03	0.19	0.01	0.00
min/b3 (r)	0.39	0.20	0.06	0.07	0.01	0.00
min/5 (r)	0.28	0.28	0.11	0.06	0.00	0.06

Table 4. CJ-DBN's bass confusion matrix.

the amount of confusion between tetrads and triads (let's call it "seventh confusion"); and the score boost from SeventhsBass to MajMin approximately sums up two types of confusion. It should be noted that there are yet other types of confusion, such as confusion of roots, or of maj and min, which could not be regarded as correct in any ways under the current evaluation method.

Following this deduction, the Sevenths result actually indicates that CJ-DBN still scores much better than Chordino if bypassing bass confusion; while the MajMinBass result indicates that CJ-DBN scores much lower than Chordino if bypassing seventh confusion. Therefore compared with Chordino, CJ-DBN has a better chance of bass confusion, but less chance of seventh confusion. Notice that in CJ-DBN, the difference between MmB and SB is much larger than that between S and SB, which means the net amount of seventh confusion is much more than that of bass confusion. The same is also true in Chordino. Therefore in both systems, there are much higher chances of making seventh confusion than bass confusion.

As for the intra-comparison among all proposed systems, three observations are noticeable: 1, DBN has advantage over MLP, and this advantage decreases with the increase of training data size; 2, BLSTM-RNN has obvious advantage over DBN with big enough training data size; 3, the investment of more data yields diminishing return. The first point is mainly due to the intensive unsupervised pre-training in DBN. The second point demonstrates that the proposed BLSTM-RNN model has better capability in modeling a single chord than the proposed DBN model. BLSTM-RNN is good at modeling temporal dependency, but DBN is good at modeling spacial dependency. An input feature with 6 frames of time dependent notegrams should be more suitable for temporal modeling, thus a plausible reason behind the second observation. The third observation may possibly point to a ground truth annotation consistency problem [11], which will be explained in next subsection.

4.1.2 Details of SeventhsBass

A deeper look at the per chord-type WCSR of SeventhsBass may reveal more details behind the overall scores.

SeventhsBass	M/5	M/3	M	M7/5	M7/3	M7/7	M7	7/5	7/3	7/b7	7	m/5	m/b3	m	m7/5	m7/b3	m7/b7	m7
B%	2.0	1.0	63.3	0.0	0.2	0.3	0.8	0.1	0.1	0.4	8.3	0.6	0.4	15.0	0.0	0.1	0.4	2.4
Chordino	19.9	17.1	54.4	0.0	0.0	0.0	55.6	0.0	0.0	5.7	41.0	0.0	0.0	54.3	0.0	0.0	0.0	51.0
CJ-MLP	15.8	19.8	58.2	0.0	0.0	0.0	30.0	0.0	0.0	9.5	11.5	3.7	0.7	54.2	0.0	0.0	0.2	19.9
CJ-DBN	19.2	21.7	63.0	0.0	0.0	0.0	35.5	0.0	0.0	20.8	9.0	5.6	0.7	59.8	0.0	0.0	0.0	21.6
CJ-BLSTM	15.3	22.4	60.4	0.0	0.0	0.0	34.8	0.0	0.0	13.1	10.2	10.2	1.0	59.0	0.0	0.0	0.0	28.0
CJK-MLP	5.6	14.2	62.7	0.0	0.0	0.0	30.7	0.0	0.0	2.7	10.0	1.7	0.0	46.7	0.0	0.0	0.0	22.8
CJK-DBN	7.6	19.2	64.2	0.0	0.0	0.0	37.7	0.0	0.0	5.0	13.5	1.9	1.6	51.5	0.0	0.0	0.0	27.0
CJK-BLSTM	6.4	12.0	70.5	0.0	0.0	0.0	37.8	0.0	0.0	10.2	8.5	9.8	1.9	48.7	0.0	0.0	0.3	32.1
CJKU-MLP	11.8	18.3	63.8	0.0	0.0	0.0	18.4	0.0	0.0	3.7	19.4	0.5	0.4	52.7	0.0	0.0	0.6	20.9
CJKU-DBN	8.2	16.2	64.3	0.0	0.0	0.0	19.5	0.0	0.0	1.2	20.4	1.9	2.1	52.9	0.0	0.0	0.0	20.2
CJKU-BLSTM	22.4	16.1	66.6	0.0	0.0	0.0	33.2	0.0	0.0	8.9	23.9	2.8	3.2	59.0	0.0	0.0	0.3	26.6

Table 2. WCSRs of every SeventhsBass category. (M=maj, m=min). %B shows the constitution of chord in test set.

Table 2 shows the categorical breakdowns of the SeventhsBass' WCSRs. Our systems' advantages in M and m are as expected. As the training data contains huge amount of their examples, deep learning models can take full advantages and draw clear boundaries between M v.s.non-M and m v.s. non-m. Table 3 and 4 show a comparison of bass confusion in Chordino and CJ-DBN [7] , which not only reflects CJ-DBN's advantages in M and m, but also confirms our previous deduction that CJ-DBN makes slightly more bass confusion than Chordino.

The results of M/5, M/3 and 7/b7 deserve further investigation. The "CJ-" systems generally perform better than Chordino in these categories. This could be due to both C and J contain a large number of consistent annotations of these chords. In the meantime we observe their scores generally drop with introduction of K and U, seemingly in exchange for more score boost from M. This seems contradictory: since all three chord types (M, M/3 and M/5) have clear distinctions by definition, thus given a neural network with enough modeling capacity and properly trained (which we assume is the case), more ground truth data should yield better classification boundaries. But instead the introduction of K and U also introduces chaotic classification behaviors regarding, M/3, M/5, 7/b7 and M. Thus we have to believe that these results conceivably point to a ground truth annotation consistency problem [11], where, for example, some similarly rendering M/3 chords in different datasets are annotated differently (as M, M/3, M/5 or others), so that when trained on a combined dataset, the classifier is getting confused about the boundaries between those similar chords. Assuming more inversions are "mis-annotated" [8] as root positions than vice versa (which might unfortunately be true), if such inconsistencies abound, classifications will be bias towards the dominating root position chords.

The most noticeable drawback of our systems is the poor performance of all sevenths chords (M7, 7 and m7) compared with Chordino. Chordino has a very nice and balanced chord confusion matrix. Shown in Table 5, almost every chord type has less than 50% confusion with other types. As for our approach, taking CJK-BLSTM as example, the main problem is that both M7 and 7 chords are easily confused with maj, and m7 is easily confused

[7] The numbers in the table are normalized durations. Reference labels are indicated by "(r)"

[8] technically not necessarily a "miss" but let's just use this expression for convenience in this context

	maj	min	maj7	min7	7
maj (r)	0.66	0.03	0.11	0.03	0.13
min (r)	0.10	0.60	0.03	0.20	0.03
maj7 (r)	0.22	0.08	0.62	0.02	0.01
min7 (r)	0.12	0.20	0.01	0.56	0.08
7 (r)	0.30	0.08	0.06	0.06	0.47

Table 5. Chordino's seventh confusion matrix.

	maj	min	maj7	min7	7
maj (r)	0.82	0.05	0.03	0.02	0.03
min (r)	0.21	0.52	0.01	0.17	0.02
maj7 (r)	0.42	0.07	0.39	0.03	0.01
min7 (r)	0.21	0.31	0.02	0.34	0.03
7 (r)	0.67	0.12	0.01	0.05	0.10

Table 6. CJK-BLSTM's seventh confusion matrix

with min (Table 6). The most undesirable case is the confusion between 7 and maj . The main reason behind this, as we try to analyze, is the different distribution of 7s in the training datasets and the test dataset. The Beatles' albums contain a lot of chord progressions that involve 7s, where the bass lines are moving by arpeggio or running as broken chords, but in CJK, there are very few such examples. CJK contains 7s that are mostly bass line static. Thus CJK-BLSTM does not have enough chance to learn 7 in dynamic bass line population, resulting in these poor results. This analysis is to some degree confirmed by the much better scores of 7 after adding dataset U, which contains a lot more 7 chord renderings in dynamic style.

For Sevenths' inversions other than "7/b7", since there are not many examples in all datasets, it is not meaningful for further discussion. Actually, their WCSRs are all relatively low. This fact might in some sense invalidate the necessity to recognize more complicated inversions, but does not invalidate the need to capture inversions in general.

4.2 Jazz Chord Vocabulary Systems Evaluation

Following the MIREX ACE convention, system performance on jazz chord vocabulary should also be evaluated based on WCSR. The WCSR score computing procedure in its fairest/strictest sense should count each chord as it is

	μ	σ^2
Bass - Chord Bass	1	0.1
Treble - Chord Note	1	0.2
Neither bass nor treble	0	0.2
"N" Chord	1	0.2

Table 7. Gaussian model of Jazz-Chordino

systems	WCSR	SQ
Jazz-Chordino	57.99	81.68
Jazz-MLP	61.81	76.18
Jazz-DBN	62.33	80.73
Jazz-BLSTM	**66.41**	80.78

Table 8. WCSRs and SQ (segmentation quality) of jazz chord vocabulary.

without applying any sort of mapping scheme, as happens to SeventhsBass. In the following we evaluate each system in this way. The baseline is an augmented Chordino with jazz vocabulary extension (Jazz-Chordino). The augmentation is done within its Gaussian-HMM engine by applying the jazz chord dictionary to the Gaussian model, whose setting is described in Table 7.

The jazz vocabulary systems have the same system framework as the SeventhsBass systems, but their deep learning models are trained using JazzGuitar99 dataset. All systems are tested using GaryBurton7 dataset [9]. Results are shown in Table 8. Jazz-BLSTM system performs the best, and outperforms Jazz-Chordino by about 10 points. The ranking is very similar to SeventhsBass', but the results are in a sense more convincing, since the test set is not dominated by chords like major and minor. In fact the composition of chords in GaryBurton7 is relatively balanced, though rare chords are still rare. Therefore in this set of results we see clearly the advantage of hybrid Gaussian-HMM-Deep-Learning approach over a pure Gaussian-HMM approach for very large chord vocabulary.

Meanwhile, notice that the SQ of these systems are all relatively high, and these are achieved in pure jazz test audio. All systems use Jazz-Chordino's Gaussian-HMM as segmentation engine. The differences between SQ scores are caused by different merging of consecutive chord boundaries in different systems. Obviously the success of Jazz-BLSTM is based on the success of the Gaussian-HMM segmentation at the beginning; then based on the robust segmentation it performs classifications without taking care of chord progression context. This task is comfortable to deal with by a fixed-length input deep learning model. The advantage may not be obvious under a small chord vocabulary, but is obvious under a large chord vocabulary.

5. CONCLUSION

In this paper we propose a hybrid Gaussian-HMM-Deep-Learning approach towards SeventhsBass and jazz vocabulary automatic chord estimation. Based on a Chordino-like segmentation engine, the approach applies two types of deep learning models, i.e., DBN and BLSTM-RNN, for chord classifications.

For SeventhsBass implementation, we train several models of each type using four datasets in an incremental way. The systems are tested using another dataset, and compared with Chordino. Results show that the best system variant, CJKU-BLSTM obviously outperforms Chordino in both Sevenths and SeventhsBass, but is slightly outperformed by Chordino in MajMin and MajMinBass. We find that our system tends to make more bass confusion but less seventh confusion compared with Chordino. The major success of our systems is in triads, while the major drawbacks are in sevenths chords. The trends within the results along incremental training data sizes may indicate a possible data annotation inconsistency issue that conceivably leads to diminishing return effect.

For jazz vocabulary implementation, we train one model for each type using JazzGuitar99 dataset, test them using GaryBurton7 dataset, and compare them with a Chordino-like system augmented with jazz chord vocabulary (Jazz-Chordino). Results show a similar system ranking as in SeventhsBass' results, with high segmentation qualities. The best system, Jazz-BLSTM, outscores Jazz-Chordino obviously. Given that GaryBurton7 is a relatively chord balanced dataset, the results demonstrate more clearly the advantage of hybrid Gaussian-HMM-Deep-Learning approach over pure Gaussian-HMM approach, which might not be so obvious with much smaller chord vocabulary.

Generally speaking, Chordino is an elegant music knowledge driven expert system that generally recognizes chords very well. But at times it fails also because of its simplicity, which fails to capture chords rendered in abnormal ways. On the other hand, our approach is data driven. The success or non-success of it depends highly on the chord balancing, distribution and population of training data. While performances on some dominating chords benefit much from the data, other performances suffer a lot from data insufficiency or inconsistency.

There are a few concerns to be addressed. The first concern is about the manually engineered segmentation engine. The Gaussian-HMM segmentation engine is good indeed, but for scientific interest, we are also very curious about whether by doing a deep training on huge amount of data can one system learn that transformation. Preliminary researches are ongoing, but none of our attempts have achieved that level yet. We believe this can be achieved gradually by deeper models and more data. A separate training for segmentation only might be beneficial. The second concern is about datasets. A better training based system asks for more ground truth annotations, especially those of skew classes, so as to train a more balanced system and to avoid the main contribution of performance being dominated by a few classes. Generally more data will lead to more examples of skew classes, but due to annotation inconsistency issue, simply "more data" may not be the final solution at all, which leaves much more works to be done in this area. Finally there is a concern of vocabulary size (seems contradictory to the previous concern), which asks for gradually exploring ACE systems' capabilities on more complex vocabularies as it is the way to approach the ultimate goal of ACE, which is to match human expert's ability of doing chord recognition.

[9] Composition of chords in GaryBurton7: maj:0.09; min7:0.13; 7:0.22; min7b5:0.12; 7b9:0.06; min:0.1; maj:0.14; others:0.14.

6. REFERENCES

[1] Gary Burton Jazz Improvisation Course. https://www.coursera.org/learn/jazz-improvisation/. Accessed: 2016-02-16.

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

[3] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Audio chord recognition with recurrent neural networks. In *ISMIR*, pages 335–340, 2013.

[4] J Ashley Burgoyne, W Bas de Haas, and Johan Pauwels. On comparative statistics for labelling tasks: What can we learn from mirex ace 2013. In *Proceedings of the 15th Conference of the International Society for Music Information Retrieval (ISMIR 2014)*, pages 525–530, 2014.

[5] Junqi Deng and Yu-Kwong Kwok. MIREX 2015 submission: Automatic chord estimation with chord correction using neural network, 2015.

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

[7] Alex Graves. *Supervised sequence labelling*. Springer, 2012.

[8] Geoffrey Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 9(1):926, 2010.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

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

[11] Eric J Humphrey and Juan P Bello. Four timely insights on automatic chord estimation. In *Proceedings of the 16th Conference of the International Society for Music Information Retrieval (ISMIR 2015)*, 2015.

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

[13] Matthias Mauch and Simon Dixon. Approximate note transcription for the improved identification of difficult chords. In *ISMIR*, pages 135–140, 2010.

[14] Johan Pauwels and Geoffroy Peeters. Evaluating automatically estimated chord sequences. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 749–753. IEEE, 2013.

[15] Jeff Schroedl. *Hal Leonard Guitar Method - Jazz Guitar: Hal Leonard Guitar Method Stylistic Supplement Bk/online audio*. Hal Leonard, 2003.

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

[17] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071. ACM, 2008.

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

[19] Xinquan Zhou and Alexander Lerch. Chord detection using deep learning. In *Proceedings of the 16th ISMIR Conference*, volume 53, 2015.