MIREX 2015 SUBMISSION: AUTOMATIC CHORD ESTIMATION WITH CHORD CORRECTION USING DEEP NEURAL NETWORK

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

ABSTRACT

We present an automatic chord estimation system DK1 based on NNLS-GMM-HMM design paradigm with a chord-correction post-processing step powered by a "SeventhsBass" chord confusion matrix, a pre-trained neural network and language model derived from the "JayChou29" data set. The system aims at beating the current records for sevenths chords and inversion chords, while also keeping an excellent performance in the traditional "MajMin" metrics.

1. SYSTEM OVERVIEW

The front-end of the system performs various signal processing techniques in order to compute a clean bass chromagram and treble chromagram. The back-end of the system decodes the chromagrams using a hidden-Markov-model and correct the resulting labels using high level informations mined from past results and ground truth annotations. (Figure 1)

1.1 Linear-freq to Log-freq Mapping

First a spectrogram of the input is computed using short-time-Fourier-transform (STFT), with parameters specified in Table 1. Then each spectrum is up-sampled 80 times using a precomputed cosine interpolation transform matrix and then mapped to a 252-bin log-frequency spectrum with 1/3-semitone per bin step (range from MIDI note 21 to 104).

1.2 Tuning and Standardization

The tuning is performed as indicated in [1], where the tuning in semitone 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 DFT of the time averaged log-frequency spectrogram. After tuning, a standardization process is performed to attenuate background noise and enhance harmonic content. It subtracts from the input matrix the running mean of every column and divides the result by the running standard deviation of the same input matrix.

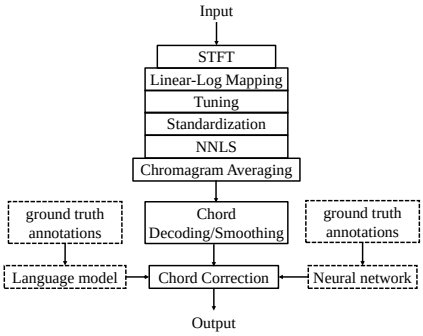

Figure 1. System Overview

Resampling rate	11025Hz
STFT hop size	512
FFT window	4096-point Hamming
FFT spectrum bins	2048
Log-freq spectrum bins	252
NNLS spectrum bins	84

Table 1. Front-end Parameters

1.3 NNLS

The output matrix from the above process is fed to a non-negative-least-square (NNLS) algorithm (Equation 2) [2], which finds for every input spectrum (Y) an optimal non-negative fit (X) of a linear combination of a set of semitone pitch profiles (P). The output is a 84-bin spectrogram with a semitone per bin step.

$$min_{X>0}||P \cdot X - Y||_2^2 \tag{2}$$

1.4 Chromagrams Computation

For better chord smoothing results, there is not any pre-segmentation scheme in this system. Especially, there is no beat-level averaging during chromagram computation. To also count in the possibility that bass signal will appear in high pitch range, the system applies a bass profile in the shape of a Rayleigh distribution as shown in Figure 2. The

	μ	σ^2
Bass - Chord Bass	1	0.1
Bass - Not Chord Bass	1	0.5
Treble - Chord Note	1	0.2
No Chord	1	0.2

Table 2. HMM-1 Parameters

chromagrams are computed by weighting the input NNLS spectrogram (84-bin) with both profiles.

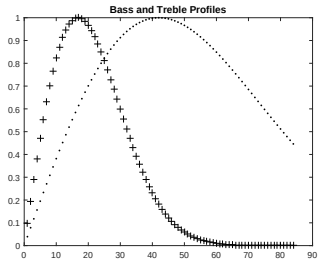

Figure 2. Bass(+) and Treble(.) Profiles

1.5 Chord Smoothing Model

The bass and treble chromagrams are then decoded by a hidden-Markov-model (HMM-1, see Figure 3). The number of its hidden states equals to the number of chords. Each hidden node generates a 24-dimension observable node, generating a 12-dimension treble chroma stacked on a 12-dimension bass chroma. Its language model is unbiased towards any type of chord transition except for giving a very high bias on self transitions. Its acoustic model is a multivariate Gaussian model with parameters specified in Table 2. Note that the parameters are slightly different from those given in [1]. They are tuned for better chord inversion recognition while not sacrificing other performances. Since the chords decoded in this stage are subject to be corrected in the next stage, thus the model in this stage is called "chord smoothing model", because the major contribution of this process is to divide the input into harmonic segments.

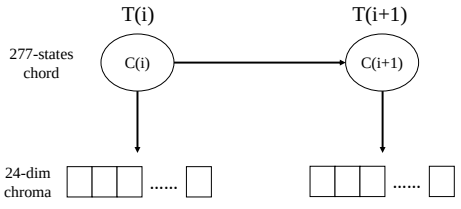

Figure 3. Chord Smoothing Model (HMM-1). Note that this HMM is used in frame scale.

1.6 Chord Vocabulary

The chord vocabulary of the system is the most complex "SeventhsBass" set. Thus the system recognizes all 12 roots with chord types maj, min, maj7, 7, min7, maj/3, maj/5, min/b3, min/5, maj7/3, maj7/5, maj7/7, 7/3, 7/5, 7/b7, min7/b3, min7/5 min7/b7 and the N chord, perfectly conforming with the current MIREX evaluation standard. [3]

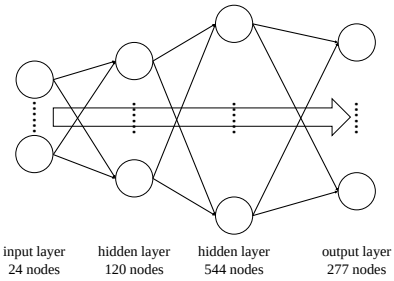

Figure 4. A 4-layer Neural Network

2. CHORD CORRECTION

After the chord progression has been decoded by HMM-1, it then goes through a chord correction process, in which every harmonic segment found by the smoothing model will be examined and labeled again. Four chord correction schemes have been implemented.

2.1 Heuristic Based Correction

This scheme tries to correct confusions between "X:{maj,min}" and "X:{maj7,7,min7}", as well as between "A:{maj,min}" and "B:{maj/3,maj/5,min/b3,min/5}". These are confusions that we think might be usually come across by the system.

2.2 Chord Confusion Matrix Based Correction

In this scheme, we first run a normal pass without chord correction to get a chord confusion matrix in "SeventhsBass" scale (totally 277 entries). We rearrange the matrix so that it can be indexed by the "confused" chords, or the "mis-recognized as" chords. The correction process tries to reestimate the scores of the top 10 most confused chords, and reassign the best label for every harmonic segments.

2.3 Neural Network Based Correction

In this scheme, we train a 4-layer neural network (as illustrated in Figure 4) to fit the JayChou29 data set. The input size is 24, corresponding to 24-dimension bass-treble chroma. The output size is 277, corresponding to the probabilities of all the chords in "SeventhsBass" category. The network is trained with 1000 iterations of full-batch gradient descent, with weight decay factor $\lambda = 1$. The trained neural network is used to recompute the chord labels for every harmonic segments.

2.4 Probabilistic Model Based Correction

In this scheme we combine the neural network trained in the previous scheme with a language model extracted from the same set of labeled data. We train a bi-gram language

model using JayChou29 data set, resulting in a 277×277 matrix corresponding to frequencies of chord transitions. The trained neural network is taken as the acoustic model. It outputs the probabilities of each of the 277 chords given a bass-treble chroma. We call a sequence of such chord probabilities a "chordogram", and a single such 277-dimension vector a "chordo". The two models are integrated under a hidden-Markov-model (HMM-2, see Figure 5) with 277 hidden states. In HMM-2, each hidden node generates a 277-dimension observable node. The emission probabilities are model as a 277-dim Gaussian with $\mu = 1$ and $\sigma^2 = 0.1$. The transition matrix is equal to the language model. The prior probabilities are set as uniform distribution.

Each segment of the chromagram are averaged and normalized as a 24-dimension chroma. After processed by the neural network, the chromagram becomes a chordogram, which is then decoded by the HMM-2. The new labels genrated by HMM-2 will be taken as the chord progression output.

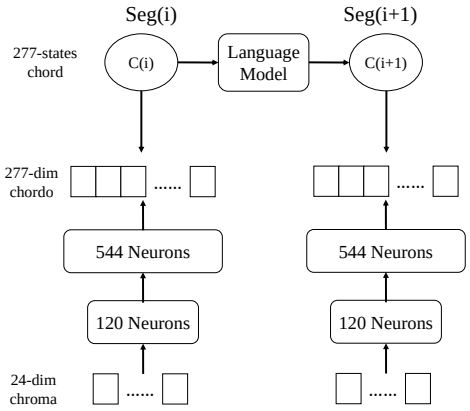

Figure 5. Probabilistic Model Based Chord Correction (HMM-2). Note that this HMM is used in segment scale.

2.5 Deep Belief Network Based Correction

In this scheme, we turn the feed-forward neural network in to a deep belief network, which composes of two stacked restricted Boltzmann Machine (RBM) and on top of which a feed-forward network. They together form an architecture usually referred to as a deep belief network. To train this network, we first pre-train the two RBMs, which requires unlabeled data. When treating the whole network as a feed-forward network, the pre-training decides a reasonable good initial weights settings. Then we do discriminative fine-tuning using back-propagation as if the whole network is just a feed forward network using labeled data.

3. REFERENCES

[1] 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.

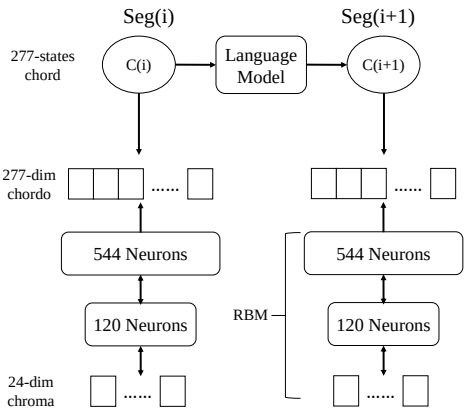

Figure 6. Deep Belief Network Based Correction. Note that the original multi-layer neural network is replaced by a restricted Boltzmann Machine .

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

[3] 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.