Chord Detection with Deep Learning: paper review and application

Fatma Moalla

Ecole Normale Supérieure & Ecole Centrale Paris
fatma.moalla@student.ecp.fr

Abstract. In this project, within the Audio Signal Processing course in MVA master 2020, we will analyze and justify the use of Deep Learning structure to learn high-level feature representation of recorded chord audio. We will review the outcome of the Deep Chord Detection network, confront it with the most used techniques used today for chord recognition and perform Chord detection on our own architecture to compare the results with the paper

Keywords: Chord recognition · Convolutional Neural Networks · Deep Learning · Music Information Retrieval · ACR · autoencoders.

1 Introduction

Music Information Retrieval (MIR) has become an important task and an urgent need alongside with the classification and prediction systems. MIR systems are used in music transcription in general and the analysis of music more specifically. One of the main applications of MIR are key detection, music similarity measures, structural segmentation, and other semantic analysis tasks. The Automatic Chord Recognition(ACR) task is one the most challenging tasks in Music Information Retrieval.

An ACR system analyzes a piece of music and predicts a sequence of labels that are associated to the chord progression played in the song. Since the majority of songs in the internet does not include transcripts, automatic chord recognition could be very useful for didactic reasons for example.

From a more technical point of view, in order to perform ACR, many studies used Deep Learning and Convolutional Neural Networks to perform Music Information Retrieval[2][11] by learning features and patterns of a large amount of Data.

In this project, we will first investigate Zhou and Lerch paper, intitled "Chord Detection Using Deep Learning" [12], which uses Deep Belief Networks in order to learn high-level features after pre-processing the audio datastests. It also uses an SVM and HMMs to perform final classification. Second , we will confront, the paper with the current state of the art [10][4] methods in the Automatic Chord Recognition field. Finally, we will test our architecture on a simple datasets that we chose in order to see the outcome of the methods on a more realistic set-up.

2 General presentation of chord systems

As we mentioned before, transcribing every note by every single instrument is a very difficult task. In many music styles like pop music for example[9], the musical content is modeled in a more compact form which is a sequence of chords.

A chord is a set of notes played simultaneously. Each note can be quantified by a number called "Pitch" which depends on the fundamental frequency of a note. We can distinguish 12 equivalence classes of the different pitches which corresponds, in other words, to all the notes in one Octave. A chord can be defined by a root note which is the note upon which the chord is perceived and a type giving the harmonic structure of the chord.

For example a C major chord is defined by a root note C and a type major which indicates that the chord will also contain the major third and the perfect fifth, namely the notes E and G. In addition to the compact aspect of the notes, This new form gives information on the harmonic interval constituted by the notes in a single chord.

3 Paper analysis

Instead of training HMMs with pitch chroma features in order to perform chord recognition like it was used in earlier papers[8], Zhou and Lerch used deep learning architecture to learn high-level features for automatic chord detection. They use a bottleneck architecture to learned features and which gives oustanding performances. They also studied the influence of pre-processing and the use SVMs and HMMs as post-proceessing classifiers.

In this section, we will go through the different parts of the classification system.

3.1 Feature extaction

First, they used new input representation by applying a Constant Q Transform (CQT). This tranformation is usually an important step when building a chromagram[1] which is a low-level feature of the audio data and it is inspired by time-frequency transformation for audio. Usually, Fourier Analysis in musical applications, lies in the fact that the frequency domain bins are linearly spaced which makes them hard to interpret. However, with CQT and with an appropriate choice of the parameters, the center frequencies of the transformation directly corresponds to the ones of the musical notes. The center frequencies of the CQT transform are the following:

$$f_k = f_0 2^{\frac{k}{\alpha}}$$

with f_k is the frequency of pitch k for $k \in [0, n]$, f_0 is the minimum frequency and α is the number of bins per octave.

In fact, the CQT spectrum used in Bonvini thesis [1] is inspired by the DFT spectrum as mentioned in Zhou and Lerch paper :

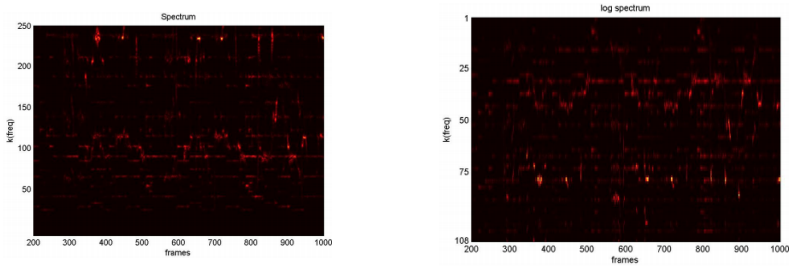
$$X[n] = \frac{1}{N[n]} \sum_{i=0}^{N[n]-1} w[i-n]x[i] \exp(\frac{-j2\pi Qi}{N[n]})$$

with w is a chosen window , $Q = (2^{\frac{1}{\alpha}} - 1)^{-1}$ the quality factor, $N[n]$ is the window length of each bin and $x[i]$ is the original signal at bin i.

A representation of the CQT transform is done by Bonvini [1] (Figure 1) :

We can also use another transform that can give similar results such as Mel-filters with the Short-time Fourrier Transform. This feature extraction method will be used in the application section.

Finally, to complete the feature extraction step, Zhou and Lerch use a Principal Component Analysis(PCA) to reduce the dimensionality and then a standard normalization.



(a) Original spectrum with the first 256 frequency bins

(b) Log spectrum obtained after the application of the constant Q transform

Fig. 1: Example of Constant Q transform

3.2 Pre-processing

In order to take advantage of the succession of the frames, several techniques are applied such as Time splicing an filter splicing.

Time Splicing It is an operation that consists in taking into account the neighboring frames and not only a single independent frame. In other words, for a single dimension, it consists in taking the previous, current and following frames and group them into "superframes" that will be overlapped.

Filter Splicing Then, several filters are applied in the same way Convolutional Neural Networks (CNNs) work.

In fact, since CNNs use several layers of convolutions that can be modeled by applying a linear and non-linear filters in a consecutive way with a pooling (average, max-pooling)function to down-sample the feature maps, as following:

$$Y = POOL(Sigm(K * X + B))$$

with Y the output of the layer, X the input, B the bias, K is the linear kernel filter and $sigm$ is a non-linear transform.
In a similar way, the paper applies several filters.

1. **A single low pass filter** described by the following equation:

$$y_n = (1 - \alpha)y_{n-1} + \alpha x_n$$

with y_n the the output, x_n the input and α a given parameter that influence how much we filter in both directions. The bidirectional filtering has the advantage of setting the phase to zero.

2. **Two low pass filters (FIR)** with exponential decay shaped impulse responses with the following expressions:

$$y_1(n) = \sum_{k=1}^{N} a^{-k+1} x(n - N + k)$$

and

$$y_2(n) = \sum_{k=1}^{N} a^{-k+1} x(n - N - k)$$

with a an exponential base and N the filter length.

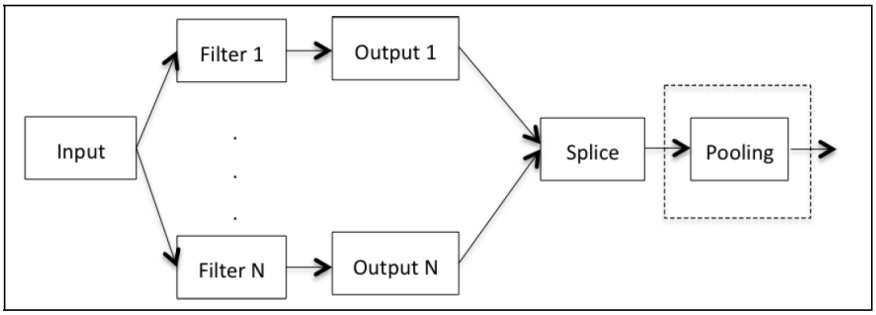


Fig. 2: Filters configurations to perform splicing

Then, the outcome of the filter is downsampled by using max-pooling for each bin.

3.3 Deep Belief Network learning and architecture

Zhou and Lerch used a Deep Belief Network [5] which consists in fully-connected layers of Restricted Boltzman Machines (RBM). Before that step, they use a Gibbs sampling to generate data on which they can pre-train the network using the greedy RBM algorithm[5]. After the pre-training, they applied Deep neural networks (DNN) with standard back-propagation to classify chord labels. The DNN is evaluated on the cross-entropy error defined as:

$$H(y', y) = -\sum_{i=1}^{n}\sum_{k=1}^{K} y_i'^{(k)} \log(y_i^{(k)})$$

with y' is the ground-truth label and y is the predicted label and K the number of classes.

The architecture used has two different aspects. One common architecture that consists in having the same number of neurons, 1024 in this paper[12], in each layer. The second aspect is a bottleneck architecture with 256 neurons in the middle layer, 512 neurons in neighboring layers. The latter architecture reassembles to auto-encoders architectures and is described in the following figure 3:

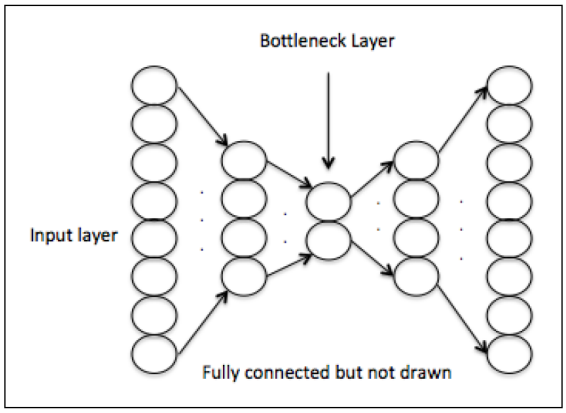


Fig. 3: Bottleneck architecture

The final output layer for both architectures is a softmax layer as follows:

$$softmax(Y_l) = \frac{\exp(Y_l)}{\sum_{k=1}^{N} \exp(Y_k)}$$

with Y_l the output of layer l.

This output is considered as the likelihood of each chord label. It is then used as intermediate features to train HMMs and SVMs in order to predict the final chord class which is described in the following section.

3.4 Post-processing

SVMs The Support Vector Machines (SVMs) is used to train the output the DBN as inputs features and to perform a labelling frame by frame before a smoothing technique.

HMMs Similarly Hidden Markov Models (HMMs) can be used for final classification. In many previous papers, HMMs[7] are the main backbone of the chord recognition solution.

In fact, chords are modeled as hidden states that can be estimated from features extracted from the audio signal. The hidden units are in fact latent variables that can be observed consecutively. The output of the softmax layer is considered therefore as emission probabilities for this HMM. They use also bigrams of chord transition to model the transition probabilities. A final Viterbi algorithm is then used to get the optimal chord labels in a consecutive way.

4 Comparison with the state of the art

In order to evaluate the model, Zhaou and Lerch used a combination of 4 datasets, they also considered 24+1 chord classes instead of 12 as following:

$$labels \subset \{N\} \cup \{maj, min\}XS$$

with S the set of the 12 pitches and N is the unknown label, maj for major and min for minor.

The use also a new classification metric, the Weighted Chord Symbol Recall (WCSR), defined as,

$$WCSR = \frac{1}{N} \sum_{k=1}^{n} C_k$$

, with n the number of audios, N is the number of frames in total, C_k is the number of correctly-classified frames in the k^{th} audio.

The best WCSR obtained is **0.919** with the bottleneck architecture, using spliced filters, with α factor for the filtering equal to 0.75, by using the 24+1 chord labels instead of the 12 pitch classes and finally by using the HMM for the final classification (post-processing).

The results of the latter algorithm are outstanding compared to standard methods that uses chroma features and only a HMM structure to perform prediction. In fact, Lee and Slaney achieved[7] a Recognition rate of **80%** on a simpler training data using few instruments.

Other papers, like Korzeniowski and Widmer work [6] achieve a WCSR of **0.821** on a richer dataset that the one used in Zhou and Lerch paper[12]. In fact, they use a probabilistic model that includes three components of a chord recognition system: the acoustic model, the duration model, and the language model. They developed RNN-based duration models and language models that outperforms HMMs, but they could not outperform the DBN results.

Thee results shown by Zhou and Lerch, might be ambitious, due to the fact that the training set is a limited and constituted only of songs and not on other instrumental recordings. Korzeniowski and Widmer work[6] seems to be more robust as it uses a richer dataset and compares the WCSR with other state of the art models and standard models.

5 Application on customized dataset

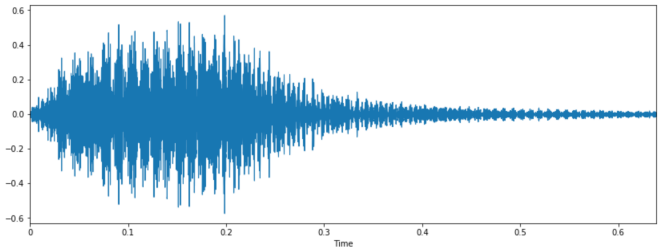
In this section, we want to perform Chord Recognition using our own Deep Learning structure and by changing the Feature extraction technique.

5.1 Datatset

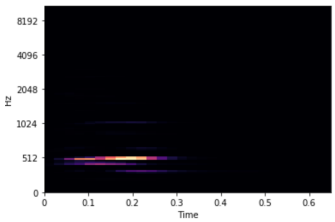
For this application , we considered a single-instrumental noisy recordings of Violin and 10 types of chords from Montefiore institute dataset[3] : AMajor, Aminor, Bminor, CMajor, DMajor, Dminor, EMajor, Eminor, FMajor, GMajor. For each chord, we have 10 recordings and the main objective is to perform automatic deep chord detection.

5.2 Preprocessing and methodology

We started by using the 100 samples of Violin recordings. We used a simple feature extraction with the 'librosa' package in python, which gets mel-filter spectogram on the raw recordings, as shown in the figure below for a recording of an "Aminor" chord.
Then, we reshaped all the feature vectors to the same shape by using a zero-padding. Then we split the data to 80% train and 20% test.



(a) Waveplot of Aminor



(b) Spectrogram of Aminor

Fig. 4: Feature extraction for the customized solution

For the training phase, we use Keras framework and we used : 2 convolutional layers with max-pooling at the end of each layer , "ReLU" activation and 1 dropout layer with 0.4 dropout and then two dense layer and a final softmax activation layer.

5.3 Evaluation

In order to perform this training, we use a cross-entropy error like in the paper[12] that we analyzed before. And we use 3 main metrics to evaluate the performance: Recall, Precision and F1-score. Customizing the score in order to get WCSR, was very hard, as recordings were very short ($\sim 2s$). That's the reason why we considered the f1-score as a metric that quantifies better the performance of our implementation.

We also performed Data augmentation by speeding audio recordings and shifting each pitch by a semi-tone. The following table compares the results between the training on the original dataset and the augmented one along with the missclassified pitches. As, we can see the results are

	F1_score(validation)	Missclassified chord labels (test)
Plain data	0.969	'CMajor' (3) and 'Eminor' (7)
Augmented (pitch_shift+speed)	**0.986**	'AMajor' (1) and 'CMajor' (3)

surprisingly high maybe because of the limited dataset that we use or because of an overfitting problem that we couldn't detect. We also notice that the class "CMajor" has missclassification errors, this is due maybe to a random phenomenon while splitting the dataset. We can also notice the importance of the data augmentation in improving the results.

6 Conclusion

In this project, we presented in details the feature extraction, Deep Belief Networks structures and the final classification methods of Zhou and Lerch's paper[12] "Chord Detection Using Deep Learning". We confronted the results of paper with the recent papers that use different techniques and we concluded that, the experimentation of the paper is optimisitic as it considers a standard song dataset taht contains fex samples. In order to confront the result of the paper with simpler datasets and in a more realistic setup, we implemented a Deep Neural Network solution on a simple Violin recorded dataset and noticed an improvement in performance regarding the F1-score but it seems to be biased by the length of the dataset.

Bibliography

[1] Bonvini, A. (2013). Automatic chord recognition using deep learning techniques.

[2] Choi, K., Fazekas, G., Cho, K., and Sandler, M. (2017). A tutorial on deep learning for music information retrieval.

[3] GROUP, M. R. (2019). The chords dataset.

[4] Hinton, G., Deng, l., Yu, D., Dahl, G., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., and Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29:82–97.

[5] Hinton, G. E., Osindero, S., and Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554.

[6] Korzeniowski, F. and Widmer, G. (2018). Improved chord recognition by combining duration and harmonic language models. In *ISMIR*.

[7] Lee, K. and Slaney, M. (2006). Automatic chord recognition from audio using a hmm with supervised learning. pages 133–137.

[8] Ni, Y., Mcvicar, M., Santos-Rodríguez, R., and Bie, T. (2012). Using hyper-genre training to explore genre information for automatic chord estimation.

[9] Oudre, L., Grenier, Y., and Févotte, C. (2011). Chord recognition by fitting rescaled chroma vectors to chord templates. *IEEE Transactions on Audio, Speech, and Language Processing*, 19:2222–2233.

[10] Rida, I., Hérault, R., and Gasso, G. (2018). An efficient supervised dictionary learning method for audio signal recognition.

[11] Salamon, J. and Bello, J. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, PP.

[12] Zhou, X. and Lerch, A. (2015). Chord detection using deep learning.