Deep Automatic Chord Recognition

Ayoub Ghriss
MVA - Ecole Polytechnique
ayoub.ghriss@polytechnique.org

Instructor: Emmanuel Bacry
CMAP, Ecole Polytechnique - Master MVA
emmanuel.bacry@polytechnique.fr

Contents

Introduction

Automatic chord detection is an important task in the analysis of music transcription and can contribute to applications such as key detection and structural segmentation. Despite early successes in chord detection by using pitch chroma features and Hidden Markov Models, recent attempts at further increasing the detection accuracy are only met with moderate improvement.

With the availability of more musical data, Music Information Retrieval (MIR) researchers in the last period has been queen to create methods to automatically extract meaningful descriptors from audio files. The chord transcription belongs to this category and it can be used as input feature by other MIR systems. The first step of the automatic chord recognition is the extraction of a meaningful descriptors. The classic strategies of ACR were based on the comparison of vectors of the extracted descriptor with a set of chords templates, such as the first ACR system by Fujishima[1]. This method is easily influenced by the noise of the descriptor itself and can be ineffective.

In recent years, deep learning approaches have gained significant popularity in the machine learning community as a way of building hierarchical representations from large amounts of data. Deep learning, with its potential to untangle complicated patterns should be well suited for the task of chord detection.

In this report, we investigate the application of Deep Networks for learning time-frequency representative features in the context of chord detection. Our review is based on Deep Chord Recognition[2]. To provide reliable outlook of ACR capabilities, additional references has been explored. Insatiable and curious readers can learn more about ACR in the master thesis [6] from Politecnico Di Milano.

1 Musical and historical background

1.1 Pitch class and chords

Human pitch-perception is periodic. If two notes are played following a unison interval, that corresponds to the same fundamental frequency, the perceived pitch is the same. While if two notes are in octave relation, that corresponds to a doubling of frequency, the perceived pitches have similar quality. This phenomenon is called *octave equivalence*. Human are able to perceive as equivalent pitches that are in octave relation.

Pitch classes are equivalence classes that include all the notes in octave relation. Every pitch class is enumerated with an integer scale from 1 to 12. It is possible to map the fundamental frequency of a pitch to a real number p using the formula:

$$p = 69 + 12 \log \frac{f}{f_{ref}}$$

where f_{ref} is the tuning frequency of the central A (A4,440Hz)

With *Chord* it is generally meant the simultaneous execution of three or more notes. Two notes played together form a particular type of interval called *harmonic interval*. Chords are made combining two or more harmonic intervals in specific relations. They are classified considering the number of notes that are being played and the type of the intervals.

1.2 The first ACR system

Legend has it that Fujishima in [1] developed the first Automatic Chord Recognition system based on Pitch Class Profile method.

This algorithm first transforms an input sound to a Discrete Fourier Transform (DFT) spectrum, from which a Pitch Class Profile (PCP) is derived. Then it performs pattern matching on the PCP to determine the chord type and root.

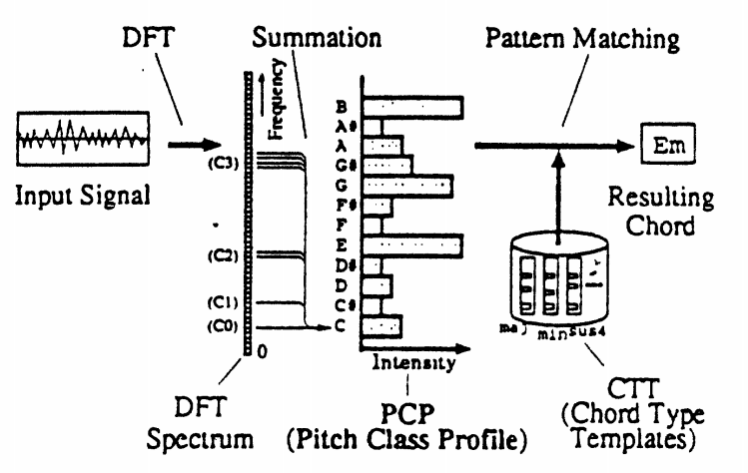


Figure 1: Fujishima's ACR System

A PCP is a twelve dimension vector which represents the intensities of the twelve semitone pitch classes. For instance, the first PCP element shows how strong C *is* in total. Once the PCP vector for a segment is obtained, the chord patterns are matched using the Nearest Neighbor method and a hand crafted score. This first ACR system would inspire the use of Constant Q Transform used in further work as in [2].

2 Feature Extraction

In [5], a comparison between time-frequency representations has been made, which included:

- Linear-scaled Short-time FT (STFT) Spectrogram
- Mel-scaled STFT spectrogram : The Mel-scale is a perceptual scale of pitches judged by listeners to be equal in distance from one another. In analogy with PCP, The reference point is assigning a perceptual pitch. Above about 500 Hz, increasingly large intervals are judged by listeners to produce equal pitch increments. As a result, four octaves on the hertz scale above 500 Hz are judged to comprise about two octaves on the mel scale.
- Constant Q Transform spectrogram

It has been shown in [5] that, overall, the Mel-Scaled STFT slightly out-performed the other representations followed by the CQT Spectrogram. However, the comparison was done fore environmental sounds and thus addressing a different question and frequencies than in our case. Generally, these two features are commonly used in ACR, including the paper [2]. We will briefly present the two transforms.

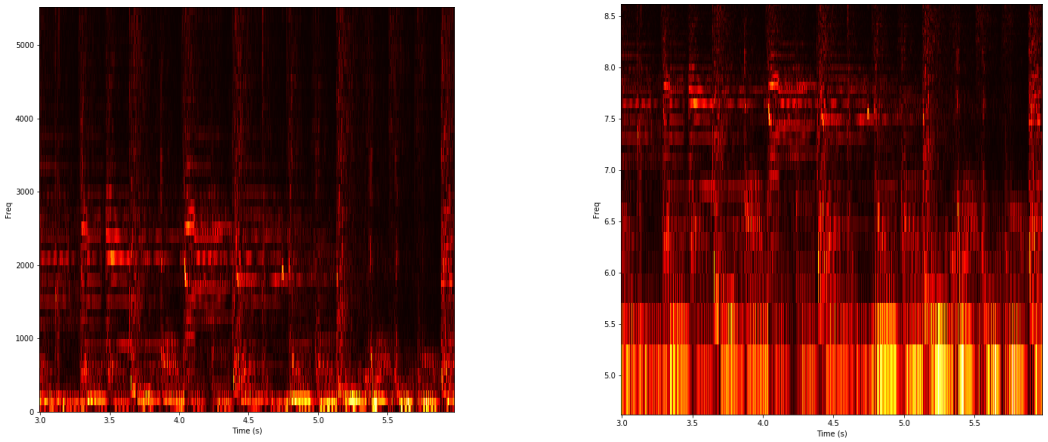
The spectrograms below have been done for on the 2^{nd} 3 seconds of the song "I saw her standing there" from the Beatles Dataset for segment of 10ms. The 10ms segment is the minimum required for human ear to recognize the tones. The sampling rate of original songs was reduced to $11025Hz$.

2.1 Short-time Fourier Transform

The STFT associates each fragment of the input sound to a DFT spectrum. Let f_s be the sampling frequency and x be a fragment of the input sound signal[1]. Then the DFT spectrum $X[n]$

$$X(n) = \sum_{n=0}^{N-1} x[n]w[n]e^{\frac{-2j\pi n}{N}-2\pi}$$

Where $w[n]$ is a windows, typically Hanning's. Since our signal x is real, we only need $X[0], \cdots, X[N/2-1]$.



(a) Linear Scale

(b) Logarithmic Scale

Figure 2: STFT Transform

2.2 Constant Q Tranform

In the usual musical applications of Fourier Analysis, the frequency domain bins are linearly spaced. This is due to the fact that the STFT is computed using a constant resolution and frequency difference.

[1] 10ms for our case

The frequencies using the equal temperament are, however, geometrically spaced. The DFT spectrum, thus, does not map efficiently to musical frequencies.

The constant Q transform was introduced in 1991 by Brown [4]. In contrast with STFT, it has geometrically spaced center frequencies $f_k = f_0 \cdot 2^{\frac{k}{b}} (k = 0, \ldots)$, where b the number of filters (bins) per octave. The bandwidth of the k-th filter as $\triangle_k^{\text{cq}} = f_{k+1} - f_k = f_k(2^{\frac{1}{b}} - 1)$. This yields a constant ratio of frequency to resolution $Q = \frac{f_k}{\triangle_k^{\text{cq}}} = (2^{\frac{1}{b}} - 1)^{-1}$. Another feature of the constant Q transform is its increasing time resolution towards higher frequencies. This resembles the situation in our auditory system. It is not only the digital computer that needs more time to perceive the frequency of a low tone but also our auditory sense.

The familiar formula of a Fourier filter at z

$$\sum_{n<N} x[n]e^{-2\pi inz/N}$$

Each component of the constant Q transform in the sequel, will be calculated as such a filter, but for suitable values for z and window length N that matches the properties discussed above.

The bandwidth of the filter is $\triangle_z^{\text{ft}} = f_s/N$, [2] independently of z. The desired bandwidth $\triangle_k = f_k/Q$ can be realized by choosing a window of length $N_k = Q\frac{f_s}{f_k}$. The frequency to resolution ratio of the filter in is $\frac{f_z}{\triangle_z^{\text{ft}}} = z$. To achieve a constant value Q for the frequency to resolution ratio of each bin: $z := Q$. Thus for integer values Q the k-th bin is the Q-th DFT-bin with window length $Q\frac{f_s}{f_k}$.

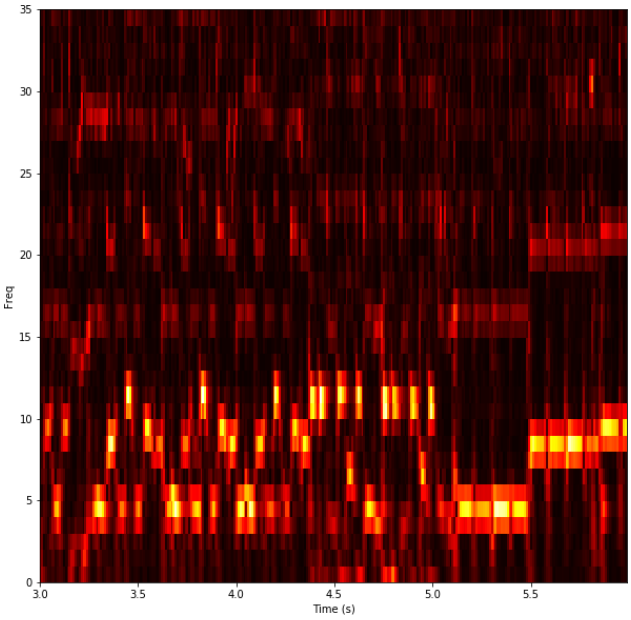


Figure 3: CQT Chroma, 36 bins

2.2.1 PCP

Now that we have seen the CQT, we can better understand the PCP used by Fujishima[1]. $PCP(p)$ is defined for the indice $p = 0, , 1 \cdots , 11$ as:

$$PCP(p) = \sum_{l \text{s.t.} M(l)=p} ||X(l)||^2$$

where $M(l)$ is a table which maps a spectrum bin index to the PCP index. Hence, the PCP transform is simply a CQT where we sum the intensities of bins with the same Pitch class.

3 Features processing

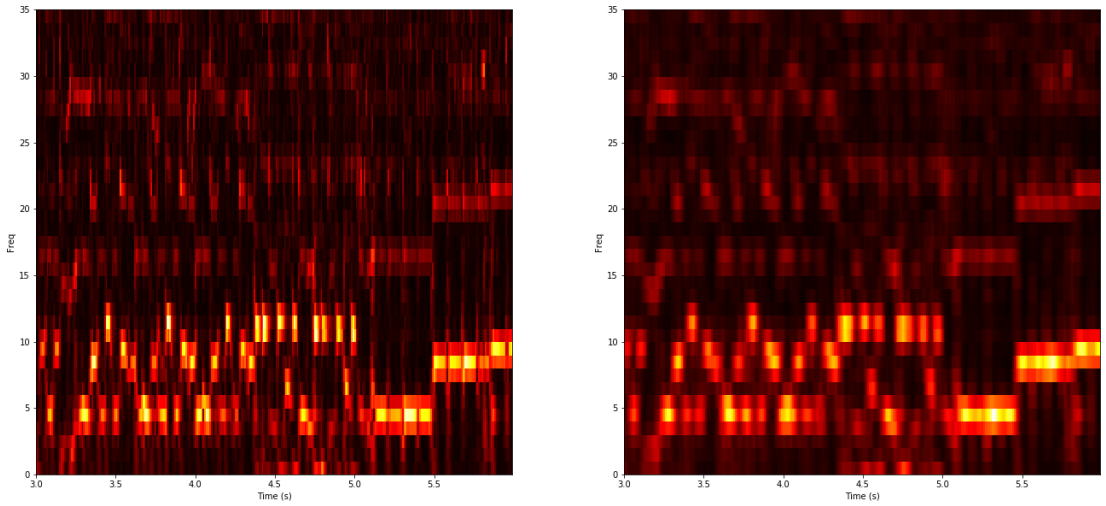
Once the time-frequency representation has been obtaining, further processing can improve the results. Such as decorrelating the features through a PCA and time splicing. The concept of time splicing by adding data from neighbor frames in [2] is simply a centered moving average of length $L = 3$, thus can be seen as low pass filter.

3.1 Denoising

The chromagram obtained from can be very noisy when the audio signal includes different instruments performing at the same time, also, to tackle the fragmentation issue due to the frame division. Besides

[2] f_s the sampling rate

denoising, the application of time domain filters has also the desirable effect of addressing temporal correlation resulting in a smoothed chromagram representation.



(a) CQT Chroma (b) Denoised Chroma

Figure 4: Centred Low-Pass filter

3.1.1 Low-Pass Filter

The simplest kind of filter used to denoise the chromagram is the low pass filter. In the case of [2], the filter is computed by a weighted moving average calculation. With this process, each element of each chroma vector is substituted by the weighted mean of the elements coming from the previous and/or following chroma vectors. In [2] two types (causal and anti-causal) of the filter has been applied separately. The weighted centered version for length L is:

$$I_{low}(b,t) = \frac{1}{L} \sum_{r=-\frac{L-1}{2}}^{\frac{L-1}{2}} \gamma^{|r|}.I(b,t+r)$$

3.1.2 Median filtering

Another commonly used denoising method is the median filter. It is non-linear filtering technique that allows deleting the outliers while preserving the edges. In particular, the median of a sequence of numbers is the middle element of the sorted sequence.

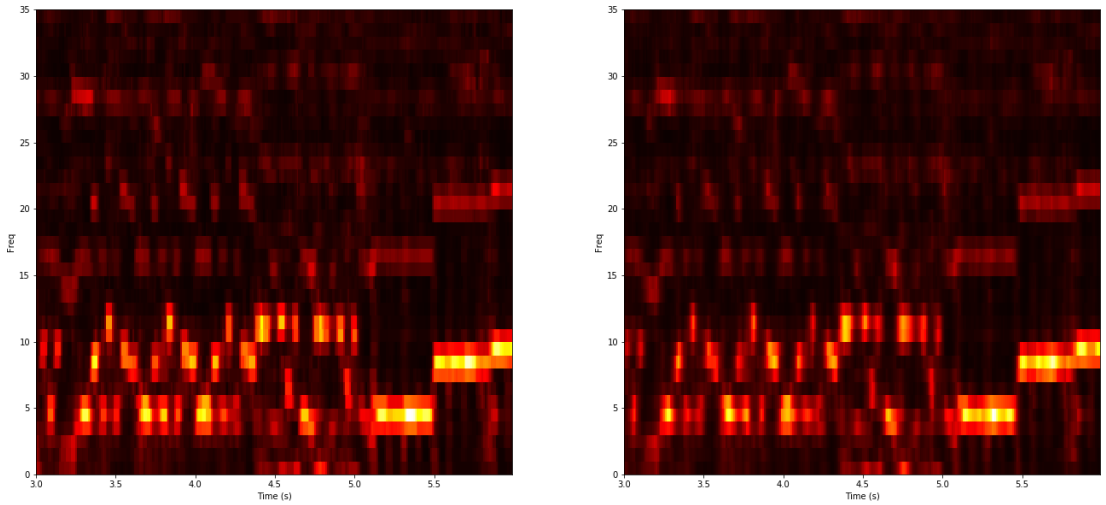
$$I_{med}(b,t) = median_{r\in[-\frac{L-1}{2},\frac{L-1}{2}]}\{I(b,t+r)\}$$

3.1.3 Geometric averaging

The geometric mean is another type of non-linear filters that is commonly used in digital image processing to blur an image. It usually performs well in removing Gaussian type noise than the simple average computed by the low-pass filter. The resulted chromagram after geometric mean filtering is the following:

$$I_{gm}(b,t) = [\prod_{r=-\frac{L-1}{2}}^{\frac{L-1}{2}} I(b,t+r)]^{\frac{1}{L}}$$

The Figure 5 shows the two filters applied to the CQT 3. The geometric filter gave smoother and better localized intensities.



(a) Median Filter, L = 6

(b) Geometric Filter, L = 3

Figure 5: Centred Low-Pass filter

4 Learning Architecture

For a while, the classical method of exploiting the chroma features has been the Hidden Markov Models as the one explicited in [7]. The HMM have known particular success in speech recognition. Even after the popularity of the Neural Networks, HMM are still used to model the sequential dependence between predictions.

When it comes to Deep architectures, it is usually a matter of personal taste. In the various papers referenced, we find Multi-Layer Perceptron : a multi fully-connected layers with non-linear activations, Recurrent Neural Networks, and Deep Belief Networks.

The task of the deep networks was to fit the chroma features data and reduce the dimensionality. The output was then plugged into classifiers such as Logistic Regression, Support Vector Machine and HMM.

4.1 Deep Belief Networks

DBNs They have been introduced by Geoffrey Hinton and provided different algorithms in several papers to solved the problem of the training of the multi-hidden layer neural networks. In particular, he proposed a greedy algorithm that trains one layer at the time in a unsupervised manner.

This is possible because they are defined as probabilistic generative models made by several layers of Restricted Boltzmann Machines (RBMs). The generative models such as the HMM, in contrast with the discriminative models such as the standard neural networks, are able to provide a joint probability distribution over labels and observable data.

4.2 Architecture

In [2], two deep architectures has been used:

- Common : A Multi-Layer Perceptron of 6 fully-connted layers, each with 2014 neurons.
- Bottleneck : A Multi-Layer Perceptron of 6 fully-connted layers, each with 2014 neurons. Except the central with 256 and its neighbors with 512 neurons

With a small dataset, RBM greedy algorithms were used to pre-train the networks on Gibbs sampled data. The last layer is a $softmax$ layer that was plugged to train HMM and SVM classifiers. The models were tested with different combinations of filtering, classifiers and networks.

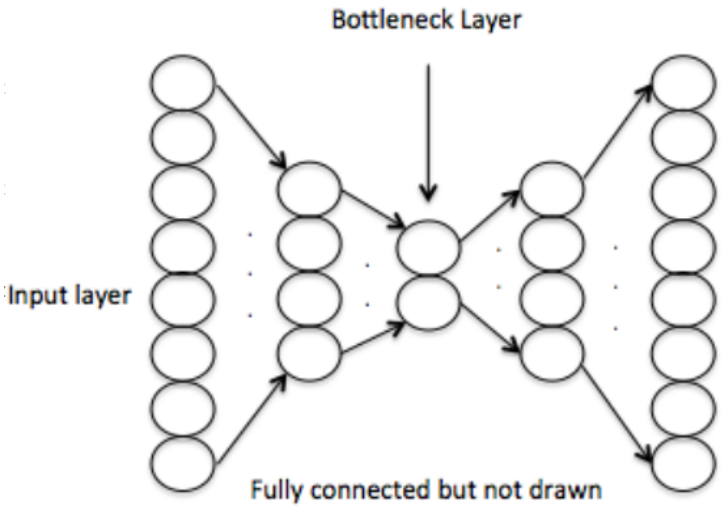


Figure 6: Bottleneck Architecture

4.3 Evaluation

The common precision metric in Deep ACR papers is defined as:

$$Precision = \frac{Total\ duration\ of\ correct\ chrods}{Duration\ of\ the\ song}$$

Best precision was achieved for the bottleneck architecture with spliced filtering. The common architecture did better on the training set, thus showing that DBN are less queen to over-fitting than as simple MLP.

In most of the Deep ACR papers, the best precision achieved was around 90%, this is to be compared with the average 70% score in methods based on pattern matching or direct application of HMM or Logistic Regression such as in [7].
In [10], a comparison between the RNN and DBN has been performed. The RNN showed a better score during training step, but was outperformed by DBN during validation. DBN manifest a sort of resiliency to over-fitting, while RNN were able to stick to their past. The RNN behavior can be particularly beneficial for well structured and "orthodox" music.

Convilutional Neural Networks have been explored for single pitch recognition applied to MIDI files in [11]. Since the MIDI framework is supposed to less challenging, it is difficult to make a comparison with the results of DBN.

[2].

Conclusion

In the previous sections, we have presented few results achieved in the recent efforts of elaborating an Automatic Chord Recognition system. While the deep models gave better results than using uniquely standard generative models, it is important to mention the scarcity of labeled music datasets compared to the applications of Deep Learning in the image recognition field.

In the available online ACR models, the ACR results are far from being convincing especially when it comes to the "attack time" detection. The performance also depends on the instruments used, thus pointing out that perhaps the aforementioned structure is best suited to the music datasets used for training. Therefore, the presented models still need to be put on diverse and unified testing framework to provide a reliable comparison.

Refrences

[1] Takuya Fujishima, Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music. *In proceedings of international computer music association (ICMC), 1999, pages 464-467, 1999.*

[2] Xinquan Zhou & Alexander Lerch, Chord Detection Using Deep Learning. *Proceedings of the 16th ISMIR Conference, Málaga, Spain, October 26-30, 2015*

[3] Matthias Mauch & Katy Noland & Simon Dixon (2009) Using Musical Structure to Enhance Automatic Chord Transcription *10th International Society for Music Information Retrieval Conference.*

[4] J. C. Brown, Calculation of a Constant Q Spectral Transform," *The Journal of the Acoustical Society of America , 1991.*

[5] Muhammad Huzaifah, Comparison of Time-Frequency Representations for Environmental Sound Classification using Convolutional Neural Networks. *arXiv:1706.07156*

[6] Alessandro Bonvini, Automatic Chord Recognition Using Deep Learning Techniques *Master Thesis, Politenico Di Milano.*

[7] Kyogu Lee & Malcolm Slaney, Automatic Chord Recognition from Audio Using a Supervised HMM Trained with Audio-from-Symbolic Data.

[8] Allen Huang & Raymond Wu, Deep Learning for Music.

[9] G.E. Hinton & R.R. Salakhutdinov, Reducing the Dimensionality of Data with Neural Networks,*Science, 28 July 2006, Vol. 313. no. 5786, pp. 504 - 507.*

[10] Nicolas Boulanger-Lewandowski & Yoshua Bengio & Pascal Vincent. Audio Chord Recogntion with Recurent Neural Networks.

[11] Anis Rojb & , Vladyslav Sarnatskyi & Vadym Ovcharenko & Mariia Tkachenko & Sergii Stirenko & Yuri Gordienko, Music Transcription by Deep Learning with Data and "Artificial Semantic" Augmentation (2017), *arXiv:1712.03228*