Analyses of Convolutional Neural Networks for Automatic tagging of music tracks

Master Thesis

Aravind Sankaran

Supervisors

Prof. Paolo Bientinesi

Prof. Marco Alunno

Examiners

Prof. Paolo Bientinesi

Prof. Bastian Leibe

Abstract

Describing music can be quite tricky and talking about music may simply require as much vocabulary as any technical subject. Musicians and composers usually discuss their work with jargons describing certain aesthetics of the song. As the amount of music recordings are constantly growing, finding a song that matches these aesthetic description is challenging. The currently available state-of-art algorithms are trained on datasets that have some linguistic noise, making the signal-semantic mapping unclear. In this thesis,a personal repertoire of labelled data is used to train a classifier, thereby reducing the semantic gap.

We address the automatic music tagging problem that can be solved by training with a personal repertoire. Trying to convey the meaning of music or its emotional content is not an easy task. This leaves a semantic gap between music audio and listener's choice of words to describe the aspects of music. Furthermore, some emotional and structural components of music are realized only after listening to a greater length of the song. Hence we address the problem of multi-label classification on features that approximates the whole song. Since personal repertoires are usually small, we stick to find solutions on a medium sized dataset. We use convolutional neural networks (CNNs) for localized feature extraction. These features are extracted every 29s from log-amplitude Mel-spectrogram and fed into sequence-to-one recurrent neural network for temporal summarization of the song, which are then used for classification. We finetune over CNN architectures in [9] [10] and report the categorical AUC scores and Mean average precision. We also report the variations in performance with increasing number of training labels. [TODO : findings]

Acknowledgments

Contents

Chapter 1

Introduction

Computers have been used to automate discovery and management of music in so many different ways. Automating the task of attaching a semantic meaning to a song is popularly known as *music auto-tagging*. Automatic tagging algorithms have been used to build recommendation systems that allow listeners to discover songs that match their taste. It also enables online music stores to filter their target audience. But semantic description of a song is not straightforward and there is a linguistic gap between music audio and listener's choice of words to describe it. In this thesis, we try to develop an algorithm that is least affected by linguistic noise. In section 1.1, the need for this dedicated research is explained by describing some shortcomings of the currently available solution approaches. In section 1.2, a top to bottom overview of the contents of this research is presented.

1.1 Motivation

But when an organization has specific terminologies for aesthetic aspects of a music piece, then bridging this linguistic gap would just be a matter of translation. However, current state of art methods still suffer from the following problems mentioned below.

1.1.1 Cold start problem with collaborative filtering methods

In the area of music information retrieval, great technical progress have been made to enable efficient retrieval and organization of audio content. But the problem of music recommendation is however complicated because of the sheer variety of genre,mood,acoustic scene, as well as social factors that affect listeners preference. When the usage data is available, one can use collaborative filtering to recommend the tracks on trending lists. In absence of such usage data, one resorts to content-based methods, where just the audio signal is used for generating recommendations. Although collaborative filtering techniques have shown to outperform content-based recommendations [6], they suffer from cold start problem, making it less efficient for new and unpopular songs.

1.1.2 Problems with content based methods

Content-based recommendation methods map audio signal to acoustic cues, which are then used for retrieval.

Psychoacoustic assumptions

The aspect of music that grabs attention is still an ongoing research in experimental psychology. Aesthetic judgments are strongly (although not exclusively) influenced by cultural variables, although there may be some universals that set constraints on what we initially find beautiful[no accounting for taste]. Hence, using currently available datasets[5][3] for training can make recommendation systems suffer from generalization assumptions.

Temporal summarization of audio content

The current state of art music tagging algorithms[10][7] are established by training on datasets that contain just short music excerpts. But choice of music is usually done based on the aspects of entire song. The ability to summarize the aesthetic judgement based on sequence of section-wise tags have not yet been studied. Furthermore, there arent much public datasets that summarize tags for entire song.

1.1.3 Need for adaptive glossary

Current recommendation systems including the ones that use collaborative filtering, restrict the user with the choice of tags. Moreover, it is not guaranteed that all users will perceive all the tags in the same way. A recent study in idiographic music psychology have indicated that different people use different aesthetic criteria to make judgement about music[11]. Hence there is a need to study the performance of recommendation algorithms trained on personal repertoire..

1.2 Structure of the Thesis

In the following work, only content-based methods are considered for multi-label classification. That is to say that only raw signal is used as input for classification. This requires feature extraction from input audio, temporal summarization of features followed by classification.

1.2.1 Convolutional neural network for feature extraction

The input signal preprocessed to spectrogram is mapped to the feature space by convolving hierarchically with learnable filters. (see ch.2,1) Conceptual arguments have been made to demonstrate that deep processing models are powerful extensions of hand-crafted feature extraction methods[8]. (see ch.3.1) It is also shown that deep layers learn to capture textures and patterns of continuous distribution on a spectrogram for music classification task. [explaining cNN for music class]. (see ch 3.2). we discuss the ability of these CNN models to be fine tuned on a medium sized dataset (see ch). Convolutions over log amplitude mel spectrogram and MFCC are studied (see ch)

1.2.2 Recurrent neural network for temporal summarization of features

The features extracted on every 29.1s time frame are then sent to sequence to one RNN. This leaves us with a feature of fixed dimension for audio of arbitrary length up to five minutes. (see ch). Here we make an assumption that a listener can make an aesthetic judgement within 5 minutes. Conceptual comparisons of RNN with the state of art temporal feature pooling technique in [ ] have been discussed (see) . Effectiveness of temporal summarization have been justified by comparing with the performance of section-wise merging of tags (see)

1.2.3 classification

The features are then mapped to the probability space of labels. Multilayer perceptron with binary cross entropy loss is used for training. End to end training is compared with two-stage method, separating training of features and training of classifier (see ch). In the two-stage approach, MLP is compared with SVM classifier. (see)

1.2.4 supervised training on repository with aesthetic tags

A properly labelled training set is usually required to solve the task of automatic tagging. In this thesis, a repository labelled with aesthetic judgements of songs is used. An aesthetic judgement is a subjective evaluation of a piece of music as art based on an individual set of aesthetic properties. [from everyday ..] An aesthetic judgement is assumed to rely more on higher cognitive functions , domain relevant knowledge, and a fluid, individualized process that may change across time and context. We discuss how aesthetic judgements influence preference for a song. In a recent work in experimental music psychology, strong correlation have been found between the two. But it is also shown that such preferences also vary widely between cultural contexts. (see) This also justifies the need for adaptive glossaries in recommendation systems.

1.3 Outline of the report

In chapter 2, the terminologies and mathematical formulations are elaborated. Advanced readers can skip this chapter. In chapter 3, a detailed overview of previous research, their shortcomings for the current problem along with justification for proposed models are discussed. In Chapter 4, details of the dataset, implementations and the experiment results of proposed models are discussed. In chapter 5, the results are analysed and the need for biologically motivated feature extraction techniques are discussed.

Chapter 2

Formalisms

In this chapter, acoustical characteristics of music signal that enables general MIR tasks will be introduced.We will examine the Fourier Series representations of sound waves and see how they relate to harmonics and tonal color of instruments

2.1 Representation of music signal

To retrieve information from a music signal, it is important to understand it's discriminants. In section 2.1.1, the abstractions that would help in analysing the organization of music-audio content are explained. The general aim of Music Information Retrieval (MIR) is to extract information about it's discriminants from the observed data. This observed signal is traditionally represented in the time domain. The time domain is a record of what happened to a parameter of the system versus time. Standard formats use amplitude versus time. The observed signal is then discretised by sampling and stored in digital format (see 2.1.2). This signal in the time domain is then changed to frequency domain. This is simply because our ear-brain combination is an excellent frequency domain analyser. Our brain splits the audio spectrum into many narrow bands and determines the power present in each band.[pp1] Conversion from time domain to frequency domain is done using the foundations from *Fourier theorem* (see 2.1.3). Currently used music signal representations for general MIR tasks are explained in section 2.1.4.

2.1.1 Discriminants of music signal - Harmonics and overtones

It was shown over one hundred years ago by Baron Jean Baptiste Fourier that any waveform that exists in the real world can be generated by series of sinusoids which are a function of frequencies. Hence any *stationary signal* $\mathbf{m}(t)$ (i.e signal at instantaneous time t) can be represented as a linear combination of function (f) of *fundamental frequency* ω_0(lowest frequency), and other frequencies ($\omega_i \vee i \in \{1, 2..N\}$) which are multiples of fundamental. The formalism of this function f will be elaborated in section 2.1.3. The following abstract representation is adapted for further explanations in this section,

$$\mathbf{m}(t) = f(\omega_0[t], c_i[t], k_i[t]) \qquad i \in \{1, 2, ..., N\}$$

Where c_i are the coefficients in linear combination and k_i are the multiples of the fundamental. It is important to note that all the variables of function f changes over time except for *stationary signals* and hence the index specification $[t]$. Therefore, whenever the assumption of *stationarity* is made, the signal is represented as

$$\mathbf{m}(t) = f(\omega_0, c_i, k_i) \qquad i \in \{1, 2, ..., N\}$$

For now, let us assume that signal is stationary. This can be imagined as a stroke of a musical note. When a note is played on an instrument, listeners hear the played tone as the fundamental, as well as a combination of its harmonics sounding at the same time (Hammond, 2011). Harmonics are tones that have frequencies that are integer multiples of the fundamental frequency. The fundamental and its harmonics naturally sound good together. So a *stationary* signal is harmonic if $k_i \in \mathbb{Z}$, where $\mathbb{Z}$ is a set of *integers*. The fundamental usually dominates the harmonics (i.e strength of higher harmonics are usually less). These additional frequencies with multiples k_i determines the *timber*. It is this *timber* that differentiates the same note played by different instruments.

(graph example of harmonics of two instruments)

The presence of multiple, simultaneous notes in polyphonic music renders accurate pitch tracking very difficult. However, there are many other applications, including chord recognition and music matching, that do not require explicit detection of pitches, and for these tasks several representations of the pitch and harmonic information commonly appear. Usually, there are more instruments being played simultaneously and sometimes accompanied by voices. In such cases, we hear the fundamental and overtones (chord). The overtones are any frequency above the fundamental frequency. The overtones may or may not be harmonics. So overtones are those frequencies which are not just restricted to integer multiples of fundamental. The fundamental and overtones together are called partials

EQ, ki != integers

m1(t) = f(440 ,880)

Where 440 = fundamental, 880 = first harmonic m2(t) = f(330 , 660)

Where 330 = fundamental, 660 = first harmonic

Thus the recorded signal has components of all these frequencies

m(t) = f(330,440,660,880)

Where 330 = fundamental, 440 = second partial, 660 = third partial, 880 = fourth partial

Most certainly, the signal evolves over time and hence the components of frequencies and its amplitudes will vary for each time t. The heat map representation of amplitudes, with frequency along y, time along x is called spectrogram.

Thus to discriminate a signal, we not only need the evolution of frequencies, but also information about harmonics. For instance, to discriminate the instruments from the recorded signal m(t), the classifier should infer the frequencies in the each harmonics m1(t) and m2(t). To discriminate the temporal pattern (Rhythm), we need the evolution of m1 and m2. To identify other aesthetics (warm, city) , the interaction between m1(t) and m2(t) should be inferred. To discriminate voices and other non-harmonic aspects (tempo, beat), the envelop curve of the spectrum will also be needed.

(diagram)

2.1.2 Sampling of continuous-time signal

The digital formats contain the discrete version of the signal obtained by sampling continuous-time signal. For functions that vary with time, let s(t) be a continuous function (or "signal") to be sampled, and let sampling be performed by measuring the value of the continuous function every T seconds, which is called the sampling interval or the sampling period.[1][pp2]. The sampling frequency or sampling rate, fs, is the average number of samples obtained in one second (samples per second),

thus fs = 1/T.

The optimum sampling rate is given by Nyquist-Shannon sampling theorem which says, the sampling frequency (fs) should be at least twice the highest frequency contained in the signal [pp2] Given the human hearing range lies between 20Hz - 20KHz [pp3], most of the digital audio formats use a standard sampling frequency of 44.4Khz. The signal is further down sampled depending on the kind of feature information needed for classification.

2.1.3 Time-Frequency transformations

The signal represented in the time domain is a set of ordered n-tuples of real numbers $(a_1, a_2, ..., a_N) \in \mathbb{R}^N$ in the vector space V , specifically *Euclidean n-space*. That is to say, a discrete-time signal can be represented as a linear combination of Cartesian basis vectors.

$$\mathbf{a}(t) = (a_1, a_2, ..., a_N) = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + ... + a_N\mathbf{e}_N = \sum_{i=1}^{N} a_i\mathbf{e}_i \tag{2.1}$$

where:

$\mathbf{a}$ is a discrete-time signal

$\mathbf{e}_1...\mathbf{e}_N$ are Cartesian basis vectors (Unit vectors).

Mapping from time-domain to frequency-domain is looked up on as basis transformation. We need to find a set of basis vectors $\boldsymbol{\phi}_\omega$, whose coefficients c_ω then represents the components in frequency domain.

$$\mathbf{a}(t) = \sum_{\omega=0}^{M-1} c_\omega \boldsymbol{\phi}_\omega(t) \tag{2.2}$$

for some integer $0 < M < \infty$. Then $\mathbf{c}(\phi) = (c_0, c_1, ..., c_{M-1}) \in \mathbb{C}^M$ represents the components in frequency domain. Thus our aim is to compute $\mathbf{c}(\phi)$ by defining basis vectors $\boldsymbol{\phi}_\omega$ which are functions of frequency. Computing the Fourier coefficients for periodic and aperiodic signals are discussed below.

Periodic Signals

If $\mathbf{a}(t)$ is periodic in $\mathbf{T}$, then we can apply the definition of **Exponential Fourier Series** expansion and define $\boldsymbol{\phi}$ in equation (2.2) as (See Appendix ??),

$$\boldsymbol{\phi}_k(t) = \frac{1}{\sqrt{T}} e^{ik\omega t} \tag{2.3}$$

Whose basis functions $\boldsymbol{\phi}$ now form *complete orthonormal* set [2]. That is,

$$< \boldsymbol{\phi}_i, \boldsymbol{\phi}_j >= \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \tag{2.4}$$

The fourier series finds a set of discrete coefficients of **harmonically related frequencies** $(k\omega)$. To retrieve c_k, multiply $\boldsymbol{\phi}_k$ on both sides of equation (2.2) and apply the conditions of orthonormality in equation (2.4). Thus

$$c_k =< \mathbf{a}(t), \boldsymbol{\phi}_k(t) > \tag{2.5}$$

Although periodicity assumptions are not made for general music signals, it becomes relevant to deduce rhythmic patterns.

Aperiodic Signals

It is difficult to assume periodicity for a generalized signal. We need to estimate the coefficients $\mathbf{c}$ for continuous frequency variable $\boldsymbol{\omega}$ instead of discrete harmonics $\mathbf{k}\boldsymbol{\omega}$. The Fourier series can not be applied directly and hence Fourier Transform was developed. Here we aim to find out quantity of each sinusoids is the signal $\mathbf{a}(t)$. This can be done by dividing $\mathbf{a}(t)$ by $e^{i\omega t}$ over the time domain. We use the complex exponential in place of sinusoids because we know (see Appendix ??)

$$sin(\omega t + \Phi) \propto e^{i\omega t} \tag{2.6}$$

Where Φ is the phase difference. Thus, the coefficients in the frequency domain are

$$c_\omega = \sum_{t=0}^{N-1} a(t) e^{-i\omega t} \tag{2.7}$$

This is the N-point **Discrete Fourier Transform**. For the proof of existence of such coefficients, please refer to chapter ?? in [2]. From here, $\boldsymbol{\phi}_\omega(t)$ in equation (2.2) can be defined as

$$\boldsymbol{\phi}_\omega(t) = e^{i\omega t} \tag{2.8}$$

Thus, we can compute $\mathbf{a}(t)$ as a linear combination of complex exponentials. This is also known as **Inverse Fourier Transform**.

$$\mathbf{a}(t) = \sum_{\omega=0}^{M-1} c_\omega e^{i\omega t} \tag{2.9}$$

Hence, with Fourier Transform, we can go back and forth between time and frequency domain. It is important to note that these basis vectors need **not** be *orthogonal*.

Fast Fourier Transform(FFT) is an efficient implementation of Discrete Fourier Transform(DFT) which exploits the symmetry of *sines* and *cosines*. While DFT requires $O(N^2)$ operations, FFT requires only $O(NlogN)$ [2].

2.1.4 STFT, Mel-Spectrogram, Chromogram

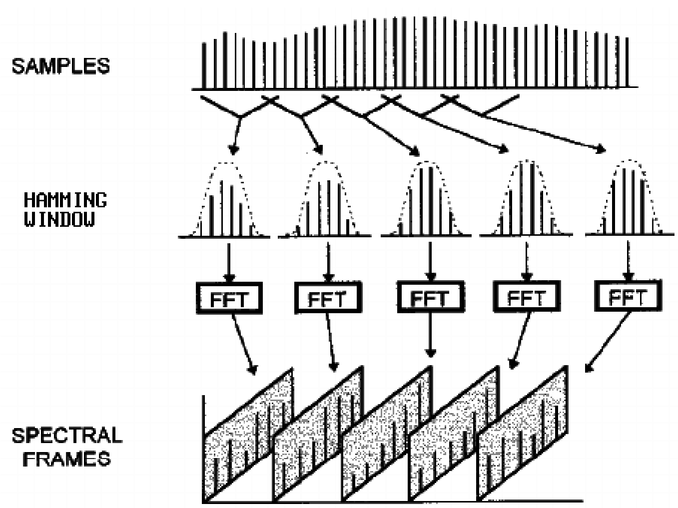
It is useful to perform FFT locally over short segments. This is simply because FFT becomes very expensive for larger N.

$$KNlog(N) < (KN)log(KN)$$

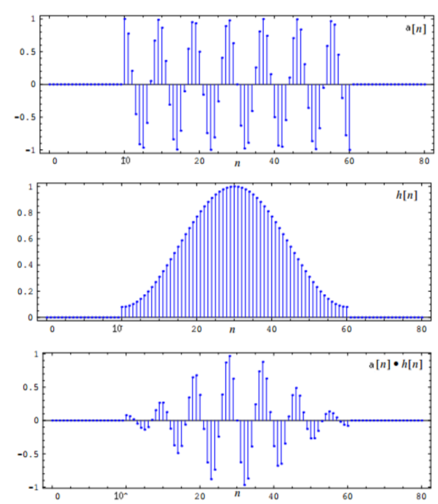
The full length signal is divided into short segments, and FFT is computed separately for each segment. This is known as **Short Time Fourier Transform (STFT)**. Usually the dimension of the frequency components are reduced by using bins. Every frequency component is assigned to it's nearest bin. This however causes **spectral leakage** when we divide the signal into rectangular windows. That is, components at the end of the segment can leak to the adjacent segment. This is avoided by modifying the original signal by applying some window function. The most common window function is the **Hamming Window** defined as,

$$h[n] = 0.54 - 0.46cos(\frac{2\pi n}{N-1}) \tag{2.10}$$

Where $n \in 0, 1, ..., N-1$. The signal approaches zero near $n = 0$ and $n = N - 1$, but reaches peak near $n = N/2$ [12]. To overcome the information loss at the ends of the window, signal is divided into segments that are partly *overlapping* with eachother. Figure (2.1 (a)) shows the extraction of spectral frames of a spectrogram.



(a) Windowing is applied on overlapping segments followed by FFT



(b) Application of Hamming Window on a segment of input signal

Figure 2.1: (a) Shows STFT Pipeline. (b) Shows the application of Window function

The discrete STFT (*slow*) for p^{th} frame of signal $\mathbf{a}$ is obtained as,

$$\mathbf{C}(p, \omega) = \sum_{n=p.s}^{p.s+F} \mathbf{a}(n)\mathbf{h}(n - p.s)e^{-i\omega(n-p.s)} \tag{2.11}$$

Where:

P: is the number of spectral frames; $p \in [0, 1..P-1]$

M: is the dimension of discrete frequency space ; $\omega \in \mathbb{R}^M$

F: is the frame length

s: is stride (or) hop-length for the next segment

$\mathbf{a} \in \mathbb{R}^N$; $n \in [0, 1...N-1]$

$\mathbf{h} \in \mathbb{R}^F$

$\omega \in \mathbb{R}^M$

$\mathbf{C}$: is Fourier Coefficient Matrix ; $\mathbf{C} : \mathbb{R}^{F.P} \to \mathbb{C}^{M \times P}$

Equation (2.11) can be seen as a **convolution** over the signal **a** with **W** which has finite support over the set $\{0, 1.., F\}$ (more details in section ??)

$$\boxed{\mathbf{C}(p, \omega) = \mathbf{a}(n) \star \mathbf{W}_\omega(n-\tau)} \tag{2.12}$$

Where:

$\tau = p.s$

$\mathbf{W}_\omega(n-\tau) = \mathbf{h}(n-\tau)e^{-i\omega(n-\tau)}$

It is important to note that the coefficients c_ω may be complex valued. They are functions of the amplitude of corresponding sinusoidal component (see Appendix ??). But, to obtain useful metrics, we need to extract some physical quantity from the coefficients. This is where **Parseval's theorem** is used, which relates and time and frequency domain components in DFT as follows [2] :

$$\|\mathbf{c}\|^2 \propto \|\mathbf{a}\|^2 \tag{2.13}$$

If **a** represents amplitude in the time-domain, then as a consequence of Hook's law on energy equation (see Appendix ??), we know that

$$Energy \propto amplitude^2 \tag{2.14}$$

Relating equation 2.11 and 2.12, it can be inferred that **square** of the Fourier coefficients is proportional to the energy distributed in the corresponding frequencies. This is called the **Power Spectrum (E)**. It is often motivating to use this representation because *loudness* is proportional to *energy*.

$$\mathbf{E} = \mathbf{C} \odot \mathbf{C} \tag{2.15}$$

As mentioned earlier, the frequencies in the considered range are grouped into bins. It is useful to do so, not only to reduce dimension but also due to the aliasing effect of human auditory system. This is motivated by the human cochlea (an organ in the ear) which vibrates at different spots depending on the frequency of the incoming sounds. Depending on how frequencies are grouped, two different class of spectrograms are discussed.

Mel Spectrogram

The *mel-scale* was developed to express measured frequency in terms of psychological metrics (i.e perceived pitch). The mel-scale was developed by experimenting with the human ears interpretation of a pitch. The experiment showed that the pitch is linearly perceived in the frequency range 0-1000 Hz. Above 1000 Hz, the scale becomes logarithmic. There are several formulae to convert Hertz to mel. A popularly used formula is noted in [1]

$$\omega_m = 2595 log_{10}\left(1 + \frac{\omega}{700}\right) \tag{2.16}$$

Where ω is the frequency in Hertz. In a mel spectrogram, the frequencies and converted to mels and then grouped into mel-spaced bins. This is done by multiplying the spectrum with some **filter bank ($\mathbf{M}_{\omega_m}$)**. For details about computation of mel-filter banks, refer [4]. Each filter bank is centered at a specific frequency. Hence, to compute R mel bins, we need R mel-filter banks.

$$\mathbf{Mel}(p, \omega_m) = \sum_{\omega=0}^{M} \mathbf{Y}(p, \omega)\mathbf{M}_{\omega_m}(\omega) \tag{2.17}$$

Where:

$\mathbf{Y} = f(\mathbf{C})$

ω_m = mel frequency

When the function f is an defined by equation (2.15), we get **mel power spectrogram**

We can re-write equation (2.17) as,

$$\mathbf{Mel}(p, \omega_m) = \sum_{k=p.M}^{p.M+K} \mathbf{U}(k)\mathbf{M}_{\omega_m}(k - p.M) \tag{2.18}$$

Where:

P: is the number of spectral frames; $p \in [0, 1..P - 1]$

M: is the dimension of discrete frequency space ; $\omega = k - p.M \in \mathbb{R}^M$

$K = M.P$ and $k \in [0, 1...K]$

$\mathbf{U}(k) = \mathbf{Y}(i, j)$; $i = floor(\frac{k}{M})$; $j = k - floor(\frac{Mk}{M-1})$

$\mathbf{Y} \in \mathbb{R}^{M \times P}$

$\mathbf{U} \in \mathbb{R}^{M.P}$

$\omega_m \in \mathbb{R}^R$

Mel: is Mel Spectrum Matrix ; $\mathbf{Mel} : \mathbb{R}^{M.P} \to \mathbb{R}^{R \times P}$

Hence, we can represent mel-spectrogram as **M-strided convolution** over *flattened* **Y** with mel filters $\mathbf{M}_{\omega_m}$ (i.e, the frequency axis of **C** is contracted with each mel-filte)r,

$$\boxed{\mathbf{Mel}(p, \omega_m) = \mathbf{U}(k) \star \mathbf{M}_{\omega_m}(k - p.M)} \tag{2.19}$$

Chromagram

This representation takes advantage of the periodic perception of pitch. Two pitches are perceived similar in "color" if they differ by one or several octaves apart. Chromagram groups such periodic perceptions into same coefficient (chroma). All pitches that belong to the same chroma are said to be from same pitch class. [TODO: How is this computed?]

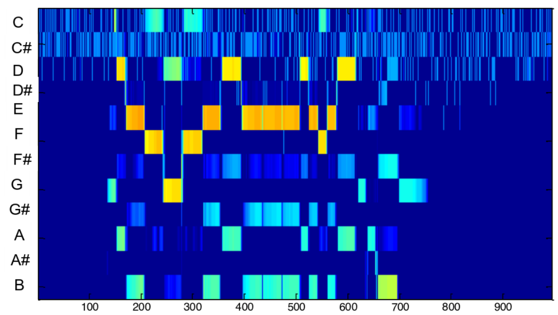


Figure 2.2: Chromagram of Western Pitch Scale

2.2 Dimensionality Reduction

The objective of dimensionality reduction is to retain only the desirable characteristics of the representations presented in section 2.1. This is done because the representation ($\mathbf{R}$) can be large for longer audio tracks (because number of frames P depends on length of the audio). Reduction over a large frame at once can lead to loss of temporal information (i.e, change of variables across frames). Hence, dimensionality reduction is done hierarchically, sometimes by stacking combination of these techniques. We generalize the operations on input signal $\mathbf{a}$ as follows,

$$\mathbf{R} = Rep(\mathbf{a}) \qquad Rep : \mathbb{R}^{N} \rightarrow \mathbb{R}^{R \times P}$$
$$\mathbf{X} = f(\mathbf{R}) \qquad f : \mathbb{R}^{R \times P} \rightarrow \mathbb{R}^{S \times Q}$$
$$\mathbf{Y} = D(\mathbf{X}) \qquad D : \mathbb{R}^{S \times Q} \rightarrow \mathbb{R}^{T \times W}$$

The representation operations defined in the previous section can be a part of the function *Rep*. Since dimension reductions can be stacked, f represents the previous reductions applied. Q and W are number of frames as a result of hierarchical windowing operation shown in fig (2.3). For the first reduction however, f does not exist and hence represented by conditional arrow (dotted). The output of reduction $\mathbf{Y} \in \mathbb{R}^{T \times W}$ will then be the reduced representation ($T < S$ or $W < Q$). Depending on how the function D is defined, we will classify the techniques into broad categories.

- Reduction by basis transformation
- Reduction by neural networks
- Reduction by clustering

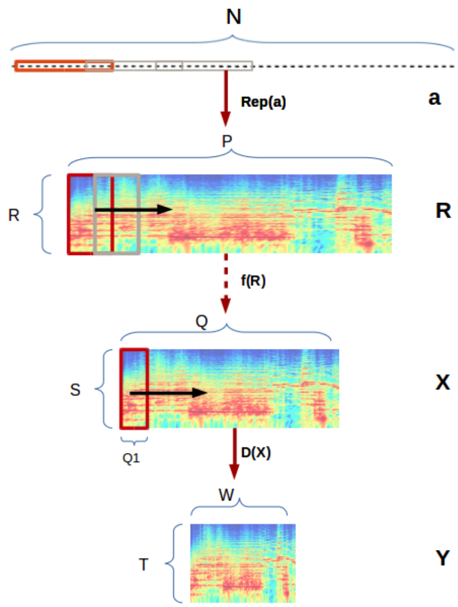


Figure 2.3: Dimensionality Reduction Pipeline

2.2.1 Basis Transformations - PCA, MFCC

The basis vectors are functions of the properties we want to encode. In equation (2.2), we used basis transformation to represent the signal in frequency domain. Now we want to use the same concept, but for dimensionality reduction. In general terms, this is done by *changing to a reduced basis*. That is, we need to find a change of coordinates matrix that will map the input to a basis system with lesser coordinates.

The input frame $\mathbf{X}_w$ is first operated with some window function $\mathbf{w}$,

$$\mathbf{z}_w = \{\mathbf{X}_w\mathbf{w} \quad | \quad \mathbf{X}_w \in \mathbb{R}^{S\times J}, \mathbf{w} \in \mathbb{R}^J, \mathbf{z}_w \in \mathbb{R}^S\}$$

Now we have to compute $\mathbf{y}_w \in \mathbb{R}^T$ which has least representation error with $\mathbf{z}_w \in \mathbb{R}^S$ such that $T < S$. Let us say, $\mathbf{z}_w$ can be written as a linear combination of some basis $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, .., \mathbf{v}_T] \in \mathbb{R}^{S\times T}$ with coordinates $\mathbf{y}_w = [y_1, y_2, ..., y_T]$. Then $\mathbf{y}_w$ can be calculated by solving,

$$\mathbf{z}_w = \sum_{i=1}^{T} y_i\mathbf{v}_i = \mathbf{V}\mathbf{y}_w$$

$$\mathbf{y}_w = \mathbf{V}^{-1}\mathbf{z}_w$$

Thus, the operator D in equation (??) is,

$$\mathbf{Y} = D(\mathbf{X}, \mathbf{V})$$

Therefore, dimension reduction through basis transformation are those class of techniques where D is a function of some invertible change of coordinates matrix $\mathbf{V}$. Now, depending on how $\mathbf{V}$ is defined, some methods are discussed.

Principal Component Analysis (PCA)

The frequencies in the adjacent bins can be highly correlated and therefore contain redundant information. Principal component Analysis is a procedure to transform large number of correlated variables into smaller number of uncorrelated variables. This means, $\mathbf{y}_w$ should be approximated only with basis vectors that account for large variance. Hence, in PCA based techniques, dimension reduction is done through basis vectors of covariance matrix ($\mathbf{\Sigma}$). The coordinates of the resulting basis system are known as *principal components*. The steps of this abstraction are enumerated,

(a) The rows of $\mathbf{X}$ are centred by their mean and covariance is computed as,

$$\mathbf{\Sigma} = \mathbf{E}[(\mathbf{X} - \mathbf{E}[\mathbf{X}])(\mathbf{X} - \mathbf{E}[\mathbf{X}])^T] = \frac{1}{Q}\hat{\mathbf{X}}\hat{\mathbf{X}}^T \quad \in \mathbb{S}^{S\times S}$$

(b) The eigen values and eigen vectors of $\mathbf{\Sigma}$ are computed. At this point, we use the Orthogonal Eigenvector Decomposition Theorem and infer that eigen vectors of symmetric matrix ($\mathbf{\Sigma}$) form an orthogonal basis in $\mathbb{R}^S$.

$$\mathbf{\Sigma} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \qquad \mathbf{V} \in \mathbb{O}^{S\times S}, \quad \mathbf{\Lambda} \in \mathbb{D}^{S\times S}$$

(c) The eigen values represent the magnitude of variance for each frequency. Hence, eigenvectors corresponding to large eigen values gives the coordinates corresponding to greater variance. So eigen vectors corresponding to top T eigen values are retained, while ignoring coordinates of lower variance. The resulting change of coordinates matrix is then $\hat{\mathbf{V}} \in \mathbb{O}^{S\times T}$

(d) Since $\hat{\mathbf{V}}$ is orthogonal, $\hat{\mathbf{V}}^{-1} = \hat{\mathbf{V}}^T$, and we can compute $\mathbf{y}_w = \hat{\mathbf{V}}^T\mathbf{z}_w$

Algorithm 1 $\mathbf{Y} = \text{PCA}(\mathbf{X})$

Input : $\mathbf{X} \in \mathbb{R}^{S\times Q}$
Output : $\mathbf{Y} \in \mathbb{R}^{T\times W}$

```
W = Q / Q_s
for i ∈ {0, 1, .., W} do
    X_s = X[i.Q_s : (i+1).Q_s]                ▷ X_s ∈ R^{S×Q_s}
    Σ = (1/Q_s) X_s X_s^T                      ▷ Σ ∈ S^{S×S}
    V, λ = EIG(Σ, T)                           ▷ λ ∈ R^T , V ∈ O^{S×T}
    Y^T[i] ← λ
end for
```

2.2.2 Feature learning with stacked convolutions

Feature engineering Vs Feature Learning

Algorithm 2 $\mathbf{Y}$ = PCA WHITENING($\mathbf{X}$)

Input : $\mathbf{X} \in \mathbb{R}^{S \times Q}$
Output : $\mathbf{Y} \in \mathbb{R}^{T \times Q}$ $\triangleright\ W = Q$
$\mathbf{\Sigma} = \frac{1}{Q}\mathbf{X}\mathbf{X}^T$ $\triangleright\ \mathbf{\Sigma} \in \mathbb{S}^{S \times S}$
$\mathbf{V}, \mathbf{\Lambda} = EIG(\mathbf{\Sigma}, T)$ $\triangleright\ \mathbf{\Lambda} \in \mathbb{D}^{T \times T}$, $\mathbf{V} \in \mathbb{O}^{S \times T}$
$\hat{\mathbf{X}} = ZERO\_MEAN(\mathbf{X})$
$\mathbf{Y} \leftarrow \mathbf{\Lambda}^{-1}\mathbf{V}^T\hat{\mathbf{X}}$

Algorithm 3 $\mathbf{Y}$ = MFCC($\mathbf{a}$)

Input : $\mathbf{a} \in \mathbb{R}^{N}$
Output : $\mathbf{Y} \in \mathbb{R}^{S \times P}$ $\triangleright\ T = S,\ W = Q = T$
$\mathbf{C} = STFT(\mathbf{a})$ $\triangleright\ \mathbf{C} \in \mathbb{C}^{M \times P}$
$\mathbf{Y}_r = MODULUS(\mathbf{C})$ $\triangleright\ \mathbf{Y}_r \in \mathbb{R}^{M \times P}$
$\mathbf{R} = MEL(\mathbf{Y}_r)$ $\triangleright\ \mathbf{R} \in \mathbb{R}^{R \times P},\ \mathbf{X} = \mathbf{R}$
$\mathbf{R} \leftarrow ln(\mathbf{R})$
$\mathbf{V} \leftarrow COSINE\_BASIS(R, S)$ $\triangleright\ \mathbf{V} \in \mathbb{R}^{R \times S}$
$\mathbf{Y} \leftarrow \mathbf{V}^T\mathbf{R}$

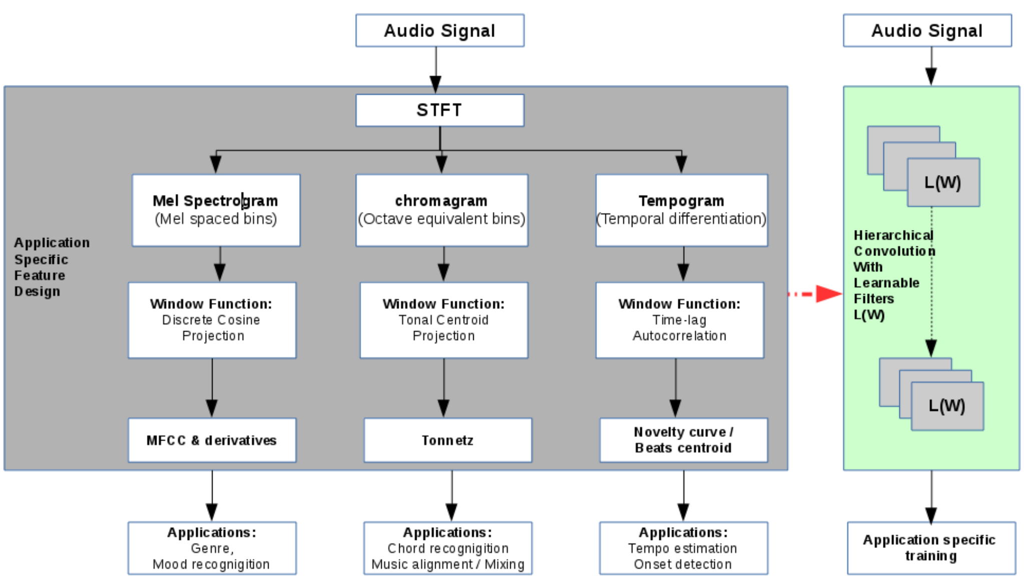


Figure 2.4: Motivation for deep architectures

2.3 Temporal pooling

2.3.1 Clustering

2.3.2 Recurrent Neural Networks

2.4 Classifier Training

Chapter 3

Literature Dynamics and Model Selection

Using content-based music information for solving several music information retrieval tasks is not new, but a decade long research efforts have been put. Hunting for the right model for our task and to justify it to be superior to the rest requires thorough understanding of evolution of such techniques. In section 3.1, the dynamics of the literature that has lead to the use of deep learning techniques for MIR tasks have been discussed. In section 3.2, the inferences from state of art techniques have been used to short list models for the experiments.

3.1 Literature Review

A number of surveys (e.g. [8, 22, 29]) amply document what is a decades-long research effort at the intersection of music, machine learning and signal processing. In a broader sense, all techniques have a two-stage architecture: first, features are extracted from music audio signals to transform them into a more meaningful representation. These features are then used as input to a classifier, which is trained to perform the task at hand. This dedicated analysis for music features emerged due to the fact that music signals possess specific acoustic and structural characteristics that distinguish them from spoken language or other non musical signals.

In a general sense, the goal of classification tasks in music informatics is to attach a semantic meaning to the content. The underlying issue is ultimately one of *organization* and *variance* of the features.

Feature organization : The better organized a feature is to answer some question, the simpler it is to assign or infer semantic meaning. Thus a feature representation should explicitly reflect a desired semantic organization. That is, the information about the discriminants (refered in 2.1.1) is not lost.
Feature variance : A feature representation is said to be *noisy* when variance in the data is misleading or uninformative, and *robust* when it predictably encodes these invariant attributes.

Complicated classifying methods are necessary only to compensate for any noise and hence a *robust* feature representation is important.

In subsection 3.1.1, some of the early works indicating the need for better feature organization are discussed. In the remaining subsections, adoption of feature learning techniques for multi-label classification task are elaborated. The general motivation of all the works from section 3.1.2 - 3.1.4, was to use feature learning to obtain *robust* and *organized* features. (In section 2.2.2, how feature learning can increase *robustness* was explained) All the models (except [],[],[]) were experimented on Magna Tag a Tune dataset [] (MTT) with about 29K clips which are 29.1s long.

3.1.1 From classifier to feature emphasis

Looking back to our history before 2010, there is a clear trend in MIR of applying increasingly more powerful machine learning algorithms to the same feature representations to solve a given task. There are also ample surveys with evidence suggesting that appropriate feature representations significantly reduce the need for complex semantic interpretation methods [2]. Evidence from genre classification and chord recognition task have been discussed below.

Audio music genre classification using different classifiers and feature selection methods. Proceedings of the International Conference on Pattern Recognition, Hong-Kong, China,2006

Fixing the features, ten different classifiers were compared, namely: Fisher (Fisher classifier), LDC (Linear classifier assuming normal densities with equal covariance matrices), QDC (Quadratic classifier assuming normal densities), UDC (Quadratic classifier assuming normal uncorrelated densities), NBC (Nave Bayes Classifier), PDC (Parzen Density Based Classifier), KNN (Knearest neighbor with optimal k computed using leave one out cross validation), KNN1 (1 nearest neighbor), KNN3 (3 nearest neighbor), KNN5. It is seen that a ceiling performance of 80% accuracy on GTZAN dataset was obtained by using combination of classifiers and squeezing every last percentage from the same features. This suggested the need for robust feature representation for further improvements.

Exploring common variations in state of the art chord recognition systems. In Proc. SMC, 2010.

The significance of robust feature representations was demonstrated by using appropriate filtering of chroma features to increase system performance even for the simplest classifier. An overall reduction of performance variation across all classifiers was also shown[9].

3.1.2 From hand-crafting to feature learning

Feature learning consists of exploiting the structure of the *data distribution* to construct a new representation of the input.Although MFCC are hand-crafted, they pose a tough competition, which makes researchers not to ignore them completely. Recalling that feature extractors can be used in hierarchy (see section ??), there is a chance that MFCCs can outperform for some combination of feature learning at higher levels. It is also worthy to consider MFCCs because of computational efficiency. Aggregating hand-crafted features for music tagging was introduced in [25]. Several subsequent works rely on a Bag of frames approach - where a collection of features are computed for each frame and then statistically aggregated. Typical features are designed to represent physical or perceived aspects of sound and include MFCCs, MFCC derivatives and spectral centroids.

Although MFCC related features work great for speech recognition tasks, it falls short in performance for MIR tasks[ ]. This is especially because long range temporal structure is crucial in music (MFCC derivatives only encode short term temporal information). In [ ], better performance was achieved by using different scales of PCA whitened frames, and achieves state of art result for multi label classification on MTT dataset till date.

[2011] Multi label class : temporal pooling auc 86 [mfcc]

The pipe line of their algorithm is shown below . The formalism of the notations used are consistent with explanations in chapter 2. The PCA whitened mel-power spectrogram is compared with MFCC features on Magna tag a tune dataset. It was shown that the former achieve a performance of AUC 0.87 out performing MFCCs which was 0.77.

Signal ($\mathbf{a}$) is sampled at 22.1 KHz. Then STFT with window length 1024 and stride 512 is computed with FFT algorithm. This is followed by conversion to mel power-spectrogram with 128 bins, followed by PCA Whitening which selects the top 120 variant frequencies. Another transformation is done by stacking a single layer perceptron ($L(\mathbf{W})$ means the weight matrix $\mathbf{W}$ is learned by training a neural network (see section ??)). The temporal pooling is done by summarizing every 2.3s frame with suitable functions (see [ ] for details). The matrix $\mathbf{W}_1$ learns the optimal features for pooling. The resulting feature is then classified by two layer perceptron with 1000 hidden units with sigmoid (σ) activations.

Algorithm 4 $Pred = \text{MODEL}(\mathbf{a})$

Input : $\mathbf{a} \in \mathbb{R}^N$
Output : $Pred \in \mathbb{R}^L$

1: $\mathbf{C} = STFT(\mathbf{a})$ ▷ $\mathbf{C} \in \mathbb{C}^{M \times P}$
2: $\mathbf{Y}_r = \mathbf{C} \odot \mathbf{C}$ ▷ $\mathbf{Y}_r \in \mathbb{R}^{M \times P}$
3: $\mathbf{R} = MEL(\mathbf{Y}_r)$ ▷ $\mathbf{R} \in \mathbb{R}^{128 \times P}$
4: $\mathbf{X}_1 = PCA\_WHITEN(\mathbf{R})$ ▷ $\mathbf{X}_1 \in \mathbb{R}^{120 \times P}$
5: $\mathbf{X}_2 = L(\mathbf{W}_1)\mathbf{X}_1$ ▷ $\mathbf{W}_1 \in \mathbb{R}^{S \times 120}, \mathbf{X}_2 \in \mathbb{R}^{S \times P}$
6: $\mathbf{y} = POOL(\mathbf{X}_2)$ ▷ $\mathbf{y} \in \mathbb{R}^{S.W}$
7: $Pred = \sigma(L(\mathbf{W}_3)\sigma(L(\mathbf{W}_2)\mathbf{y}))$ ▷ $\mathbf{W}_2 \in \mathbb{R}^{1000 \times S.W}, \mathbf{W}_3 \in \mathbb{R}^{L \times 1000}$

It is important to note that this algorithm is does not work on audio of arbitrary length because of their design of temporal pooling (because fixed sized features are needed for classification).

[2012] Multi scale : spec : auc 89.8 (still not deep learning, feature learning is not task specific)

The result reported by this model is the current state-of-art on MTT dataset (AUC 0.898). Here, reducing the mel spectrogram to different sizes is done in parallel by gaussian pyramids. The resulting features are then concatenated. This was mainly done to see the relevance of rhythmic structure from longer time-scales. PCA whitened frames in the mel-spectrogram are subjected to unsupervised learning with K-Means to get the bag of words features (see 2.3.1). It has been shown that learning features at larger timescales in addition to short time scales improves performance. This also suggests the existence of periodicity at longer timescales emerging from rhythmic structure, repeated motifs and musical form.

Algorithm 5 $Pred = \text{MODEL}(\mathbf{a})$

Input : $\mathbf{a} \in \mathbb{R}^N$
Output : $Pred \in \mathbb{R}^L$

$\mathbf{C} = STFT(\mathbf{a})$ ▷ $\mathbf{C} \in \mathbb{C}^{M \times P}$
$\mathbf{Y}_r = \mathbf{C} \odot \mathbf{C}$ ▷ $\mathbf{Y}_r \in \mathbb{R}^{M \times P}$
$\mathbf{R} = MEL(\mathbf{Y}_r)$ ▷ $\mathbf{R} \in \mathbb{R}^{R \times P}$
for $i \in \{1, .., W\}$ **do**
    $\mathbf{X}_1 \leftarrow GAUSSIAN\_PYRAMID(\mathbf{R}, i)$ ▷ $\mathbf{X}_1 \in \mathbb{R}^{R \times Q1_i}$
    $\mathbf{X}_2 \leftarrow PCA\_WHITEN(\mathbf{X}_1)$ ▷ $\mathbf{X}_2 \in \mathbb{R}^{S1 \times Q1_i}$
    $\mathbf{X}_3 \leftarrow BAG\_OF\_WORDS(\mathbf{X}_2, J)$ ▷ $\mathbf{X}_3 \in \mathbb{R}^{S2 \times Q2_i}$
    $\mathbf{Y}[i] \leftarrow MAX\_POOL(\mathbf{X}_3)$ ▷ $\mathbf{Y}[i] \in \mathbb{R}^{S2}, \mathbf{Y} \in \mathbb{R}^{S2 \times W}$
end for
$\mathbf{y} = FLATTEN(\mathbf{Y})$ ▷ $\mathbf{y} \in \mathbb{R}^{S2.W}$
$Pred = \sigma(L(\mathbf{W}_3)ReLU(L(\mathbf{W}_2)\mathbf{y}))$ ▷ $\mathbf{W}_2 \in \mathbb{R}^{1000 \times S2.W}, \mathbf{W}_3 \in \mathbb{R}^{L \times 1000}$

The take-away is that modelling relation between features at rhythmic intervals does help.

3.1.3 Transfer Learning by supervised pre-training

Stacked feature learning techniques typically require large amounts of training data to work well. The following publication propose to exploit models trained on larger datasets like Million Song Dataset and use those weights as initialization for classification on smaller datasets. This is called supervised pre-training and it is essential to have a source task that requires a very rich feature representation, so as to ensure that the information content of this representation is likely to be useful for other tasks

[2014] Transfer learning : auc 88.0

This model achieves AUC 0.88 on MTT dataset. The training on MTT dataset (29k clips) was done by initializing weights of a model trained on MSD dataset (1000K clips). The workflow for source and target are shown below,

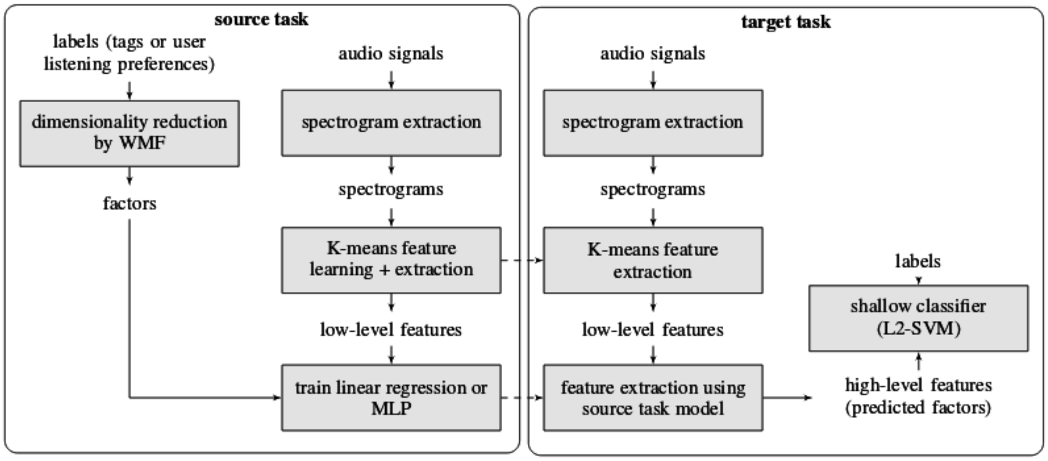


Figure 3.1: Schematic overview of the workflow of transfer learning

Source task: The low-level features from audio spectrograms are learned through unsupervised learning by spherical K-Means. A multi layer perceptron is then stacked to obtain final prediction. So the output from the penultimate layer of MLP are treated as transferable features. To tackle problems created by redundant and sparse labels, dimensionality reduction is done in the label space using PCA. The model is then trained to predict the reduced label representation.

Target task Next, the trained models are used to extract features from other datasets, which are then passed to train shallow classifiers for different but related target tasks. This workflow is visualized in fig[ ] Dashed arrows indicate transfer of the learned feature extractors from the source task to the target task.

It has been shown that features learned in this fashion work well for auxiliary audio classification tasks on different datasets, consistently outperforming a purely unsupervised feature learning approach.

3.1.4 Convolutional Neural Networks

It can be seen that deep signal processing structures can be realized by stacking multiple shallow architectures. As feature learning was proving to be more efficient than hand crafted features, stacking learnable layers over one another became a hot area of research. Following the success of convolutional neural networks in computer vision [ ] and speech recognition [], experiments were also done for music auto-tagging [] [] []. The idea was to replace the application specific dimension reductions with hierarchy of learnable convolution filters (see **??**)

End to end learning of music audio

As shown in chapter 2, all operations including FFT can be defined in terms of convolutions. In this research they investigate whether it is possible to apply feature learning directly to raw audio signal.

The signal was convolved with 3 layers of 1D convolutions followed by two fully connected layers for predicting the tag. Thus, the feature and the classifier was trained in a single pipeline and this was called end to end learning. They compared the end to end learning approach with convolutions from mel-spectrogram on MTT dataset (i.e, retaining STFT). It was found that, discarding STFT hurt the performance. CNN from mel-spectrogram achieved 0.8815 AUC, but on including convolutions on audio signal AUC dropped to 0.8487.

Algorithm 6 CNN(raw audio) [**0.84**]

Input : $\mathbf{a} \in \mathbb{R}^N$
Output : $Pred \in \mathbb{R}^L$

$\mathbf{C}_1 = Log(\mathbf{a} \star \mathbf{w}^{(256)}_{(256)})$
$\mathbf{C}_2 = MaxPool(ReLU(\mathbf{C}_1 \star \mathbf{w}^{(1)}_{(32)}))$
$\mathbf{C}_3 = MaxPool(ReLU(\mathbf{C}_2 \star \mathbf{w}^{(1)}_{(32)}))$
$\mathbf{y} = FLATTEN(\mathbf{C}_3)$
$Pred = \sigma(L(\mathbf{W}_2)ReLU(L(\mathbf{W}_1)\mathbf{y}))$

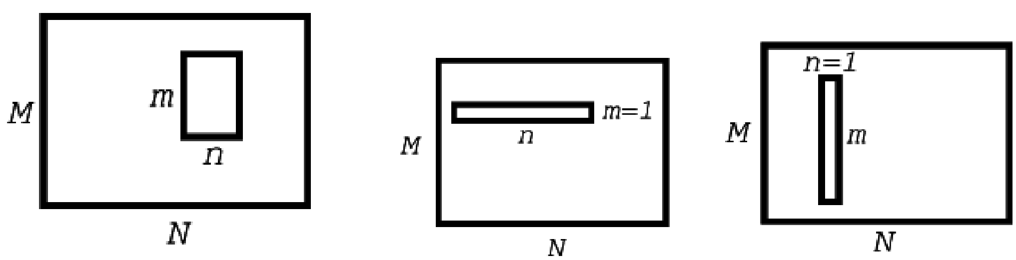
Algorithm 7 CNN(Mel-Spectrogram) [**0.88**]

Input : $\mathbf{a} \in \mathbb{R}^N$
Output : $Pred \in \mathbb{R}^L$

$\mathbf{R} = MEL(||STFT(\mathbf{a})||^2)$
$\mathbf{C}_1 = MaxPool(ReLU(\mathbf{R} \star \mathbf{w}^{(1)}_{(32)}))$
$\mathbf{C}_2 = MaxPool(ReLU(\mathbf{C}_1 \star \mathbf{w}^{(1)}_{(32)}))$
$\mathbf{y} = FLATTEN(\mathbf{C}_2)$
$Pred = \sigma(L(\mathbf{W}_2)ReLU(L(\mathbf{W}_1)\mathbf{y}))$

Experimenting musically motivated CNNs

In the previous section, only 1D convolution with filter sizes directly motivated by hand-crafted methods were tested for comparison. But usually, the convolution operation allows flexibility in choosing the filter sizes. In this research, the authors discuss how convolution filters with different shapes can fit specific musical concepts.



(a) Rectangular filter (b) Time filter (c) Frequency filter

Figure 3.2: Different Filter sizes

Time filters can learn temporal cues (Onset, BPM and other rhythmic patterns), while *frequency filters* can differentiate timbre and note. *Rectangular filters* can learn short time sub-bands (Bass, kick, drums) [ ]. However, because of hierarchical nature of deep networks, any filter should be theoretically capable of picking up the relevant cues. It was shown in their experiments that *rectangular filters* or combination of time and frequency filters performed better than using just time / frequency filter. These experiments were however done for a genre classification task.

Automatic tagging using deep convolutional neural network

Different CNN architectures were tested and the proposed model achieves close to state of art performance on MTT dataset (0.894 AUC). The audio samples were down-sampled to 12 KHz and convolutions were started from mel spectrogram (96 bins). They also compared MFCCs, convolutions over STFT and convolutions over Mel-spectrogram and report that the latter performs significantly better.

Model	**AUC**
STFT $\rightarrow$ CNN	0.846
STFT $\rightarrow$ MEL $\rightarrow$ CNN	**0.894**
STFT $\rightarrow$ MEL $\rightarrow$ MFCC	0.862

Also, to exploit the advantage of PCA Whitening proven in [] [], Batch Normalization[] of frequency components is done. That is, data is centred to the batch mean and divided by batch variance. In Batch normalization however, the basis is not switched but the data is *learned* to be scaled and shifted.

Algorithm 8 $\hat{\mathbf{X}}$ = BATCHNORM($\mathbf{X}$)

Input : $\mathbf{X} \in \mathbb{R}^{B\times S\times Q}$, $\triangleright$ B is batch size
Output : $\hat{\mathbf{X}} \in \mathbb{R}^{B\times S\times Q}$
Parameters to learn : γ (Scale), β (Shift)
1: $\boldsymbol{\mu}, \boldsymbol{\sigma}^2 = FREQUENCY\_MEAN\_VARIANCE(\mathbf{X})$ $\triangleright$ $\boldsymbol{\mu}, \boldsymbol{\sigma}^2 \in \mathbb{R}^S$
2: **for** $i \in \{1, .., B\}$ **do**
3: **for** $j \in \{1, .., Q\}$ **do**
4: $\mathbf{X}[i, :, j] \leftarrow \frac{\mathbf{X}[i,:,j]-\boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2-\epsilon}}$
5: **end for**
6: **end for**
7: $\hat{\mathbf{X}} = \gamma\mathbf{X} + \beta$

The five layer proposed CNN architecture is shown below. The filters $\mathbf{W}$ in each layer are the weights that will be learned. $Spatial\_Bn$ is similar to the normalization algorithm mentioned above, except that the normalization is done along the 1st axis of tensors $\mathbf{C}$. $MaxPool_{i,j}$ is a dimensionality reduction done by pooling (i, j) elements along S and Q directions respectively. Elu is an element-wise non-linearity operation described in section (??)

Algorithm 9 $\mathbf{y} = CHOI\_CNN(\mathbf{R})$

Input : $\mathbf{R} \in \mathbb{R}^{1\times96\times1366}$

Output : $\mathbf{y} \in \mathbb{R}^{1024}$

$\mathbf{R}_n = BatchNorm(\mathbf{R})$

$\mathbf{C}_1 = \mathbf{R}_n \star \mathbf{W1}^{(1,1)}_{(32)}$ $\quad\triangleright\ \mathbf{W1} \in \mathbb{R}^{32\times3\times3}, \mathbf{C}_1 \in \mathbb{R}^{32\times S1\times Q1}$

$\mathbf{C}_1 \leftarrow MaxPool_{(2,4)}(Elu(Spatial\_Bn(\mathbf{C}_1)))$ $\quad\triangleright\ \mathbf{C}_1 \in \mathbb{R}^{32\times T1\times W1}$

$\mathbf{C}_2 = \mathbf{C}_1 \star \mathbf{W2}^{(1,1)}_{(128)}$ $\quad\triangleright\ \mathbf{W2} \in \mathbb{R}^{128\times3\times3}, \mathbf{C}_2 \in \mathbb{R}^{128\times S2\times Q2}$

$\mathbf{C}_2 \leftarrow MaxPool_{(2,4)}(Elu(Spatial\_Bn(\mathbf{C}_2)))$ $\quad\triangleright\ \mathbf{C}_2 \in \mathbb{R}^{128\times T2\times W2}$

$\mathbf{C}_3 = \mathbf{C}_2 \star \mathbf{W3}^{(1,1)}_{(128)}$ $\quad\triangleright\ \mathbf{W3} \in \mathbb{R}^{128\times3\times3}, \mathbf{C}_3 \in \mathbb{R}^{128\times S3\times Q3}$

$\mathbf{C}_3 \leftarrow MaxPool_{(2,4)}(Elu(Spatial\_Bn(\mathbf{C}_3)))$ $\quad\triangleright\ \mathbf{C}_3 \in \mathbb{R}^{128\times T3\times W3}$

$\mathbf{C}_4 = \mathbf{C}_3 \star \mathbf{W4}^{(1,1)}_{(192)}$ $\quad\triangleright\ \mathbf{W4} \in \mathbb{R}^{192\times3\times3}, \mathbf{C}_4 \in \mathbb{R}^{192\times S4\times Q4}$

$\mathbf{C}_4 \leftarrow MaxPool_{(2,4)}(Elu(Spatial\_Bn(\mathbf{C}_4)))$ $\quad\triangleright\ \mathbf{C}_4 \in \mathbb{R}^{192\times T4\times W4}$

$\mathbf{C}_5 = \mathbf{C}_4 \star \mathbf{W5}^{(1,1)}_{(256)}$ $\quad\triangleright\ \mathbf{W5} \in \mathbb{R}^{256\times3\times3}, \mathbf{C}_5 \in \mathbb{R}^{256\times S5\times Q5}$

$\mathbf{C}_5 \leftarrow Elu(Spatial\_Bn(\mathbf{C}_5))$

$\mathbf{y} = Flatten(\mathbf{C}_5)$ $\quad\triangleright\ \mathbf{y} \in \mathbb{R}^{1024}$

The features from convolutions then pass through a fully connected layer of size equalling number of tags. The authors have then trained this model on MSD dataset and made the weights publicly available.

3.2 Model Selection

In section 3.1, it was stated that when the feature is well *organized* and encodes the *variance* in the data, it becomes easier to attach a semantic meaning. Feature learning can increase robustness, but to learn an organized representation is not guaranteed. That is to say, the extracted feature should encode the information about it's discriminants related to the task. Our brain differentiates sounds with energy changes, and this is approximated by MFCCs or Principal components (ref 2.2.1) through proportionate energy variance from a mel-spaced spectrogram (ref 2.1.4). But in section 2.1.1, it was argued that the difference between music and speech is that, a music signal is composed of several superimposed *rhythmic traces*. It was not clear if the classifier could decompose the rhythms from engineered features and hence there was this movement from feature engineering to feature learning. The results from the work [], where fully unsupervised technique is adopted for feature learning, shows that hand-crafted features does lose some information necessary for classification. But these features were extracted from mel-spaced frequency spectrogram that does not exploit the harmonic encodings. That is, we still do not know if the learning algorithms extract the rhythms, thereby questioning the optimality of *feature organization*. However in general, it could be seen that feature learning performs better than MFCCs for multi-label classification task [auto tag, temporal].

Music tagging problem is further complicated by the complexity of semantic assignments that reflect user preference. To get the right discriminants, the training method should be properly defined in the first place. In section 1.1.2, the problems with content based methods that stem from training assumptions were pointed out. One of which was the generalization assumption resulting from training on large datasets. This leaves us with the question, if the models trained by supervised learning on larger datasets [auto tagging, 1D] can be used for smaller datasets. Secondly, the psychoacoustic assumptions resulting by training on short excerpts of music rather than whole song cause vagueness. This is because, the currently available large datasets only contain short clips and the current training algorithms generalize the tags for the whole song by merging tags from short sections of the song. Therefore, methods that hold better feature organization for songs of arbitrary length are also explored.

3.2.1 Transfer learning Vs MFCC

To check if models trained on large datasets can be exploited for smaller datasets, *transfer learning* from the model which achieves state of art performance with CNN [auto tagging] is compared with MFCC features. (It makes perfect sense to compare the state of art unsupervised feature learning algorithm [] as well, but in this thesis I stick to analysing CNNs). This will also show if features learned (stacked convolutions - ref. 2.2.2) through supervised training on large dataset attain better *feature organization* than MFCCs. It is important to note that, better *feature organization* simply does not mean that classifier identifies the *rhythmic traces*.

3.2.2 K-Means vs RNN

To summarize tags for songs of arbitrary length, most of the current algorithms classify short sections of the spectrogram separately and finally merge tags across different sections [end to end]. It is also possible to stack a *decision tree* above the feature extractor to improve the performance, but that would not tell anything about the optimality of *feature organization*. Hence for temporal pooling, only methods that directly work on content information are considered. Algorithm in [multi scale] is designed to handle songs of arbitrary length and it was seen that bag of words features trained using K-Means algorithm (see 2.3.1) proved efficient while testing on 29.1s excerpts from MTT dataset. However, it is not clear if these features are optimal choice for identifying rhythms. It is also not known if the efficiency of K-Means will be retrained when tested on songs longer than 30s. Hence, this algorithm is compared with temporal summazization using Recurrent Neural Network (see 2.3.2). Supervised training with RNN might force the classifier to look for rhythmic content.

Chapter 4

Experiments and Results

The aim of this thesis is to find the optimal algorithm for content-based multi-label classification of music tracks, that can be solved with minimal training data. In Chapter 3, the state of art models were reviewed and in section 3.2, the short-comings of these algorithms in addressing the problems discussed in Chapter 1 (see 1.1.2) were pointed out. In this chapter, the experiments that will lead to finding the best algorithm using the components short-listed in 3.2 will be described.

4.1 Dataset and Evaluation

4.2 Experiments

4.3 Summary of Results

Appendix A

A.1 Basis Transformation

Here we discuss only transformation from standard basis or Cartesian basis.

The standard basis for $\mathbb{R}^N$ is the ordered sequence $\mathbf{E}_n = [\mathbf{e}_1, \mathbf{e}_2, .., \mathbf{e}_n]$, where $\mathbf{e}_i$ is a vector with 1 in i^{th} place and 0 elsewhere. Any vector $\mathbf{x} = [x_1, x_2, ..., x_n] \in \mathbb{R}^N$ can be represented as a linear combination of $\mathbf{E}_n$ as,

$$\mathbf{x} = \sum_{i=1}^{N} x_i \mathbf{e}_i = \mathbf{E}_n \mathbf{x}$$

Basis transformation from standard basis is defined as representing the same vector $\mathbf{x}$ with the new co-ordinates $[y_1, y_2, ..., y_m]$ in basis $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, .., \mathbf{v}_m] \in \mathbf{R}^{N \times M}$.

$$\mathbf{x} = \sum_{i=1}^{M} y_i \mathbf{v}_i = \mathbf{V}\mathbf{y} \qquad \mathbf{y} \in \mathbb{R}^M$$

$\mathbf{V}$ is also known as **change of coordinates matrix** (also stated as any matrix whose columns form a basis). If $\mathbf{V}$ is orthogonal, then $\mathbf{V}^{-1} = \mathbf{V}^T$ and hence $\mathbf{y} = \mathbf{V}^T\mathbf{x}$

A.2 Convolution

Only discrete convolutions with finite support are discussed below,

A.2.1 1D Convolution

Convolution of a vector $\mathbf{f}$ with filter $\mathbf{w}_k$ of stride s is defined as,

$$\mathbf{C}(k,i) = \sum_{n=i.s}^{i.s+F} \mathbf{f}(n)\mathbf{w}_k(n - i.s) \qquad \mathbf{f} \in \mathbb{R}^N, \mathbf{w}_k \in \mathbb{R}^F, \mathbf{C} \in \mathbb{R}^{K \times I} \tag{A.1}$$

$$\mathbf{C}(k,i) = \mathbf{f}(n) \star \mathbf{w}_k(n - i.s)$$

Where:

K is the number of filters. $k \in 0, 1 ... K - 1$

I is the number of contractions. $i \in 0, 1 ... I - 1$

Short Hand Notation : 1D Convolution of $\mathbf{f}$ with filter $\mathbf{w}_k$ with stride s

$$\boxed{\mathbf{C}(k, :) = \mathbf{f} \star \mathbf{w}_k^{(s)}}$$

A.2.2 2D Convolution

Convolution of a matrix $\mathbf{F}$ with filter $\mathbf{W}_k$ of row-stride s and column-stride t is defined as,

$$\mathbf{C}(k, j, i) = \sum_{n=i.s}^{i.s+F} \sum_{m=j.t}^{j.t+G} \mathbf{F}(m, n) : \mathbf{W}_k(m - j.t, n - i.s) \qquad \mathbf{F} \in \mathbb{R}^{M \times N}, \mathbf{W}_k \in \mathbb{R}^{G \times F}, \mathbf{C} \in \mathbb{R}^{K \times J \times I}$$
$$\mathbf{C}(k, j, i) = \mathbf{F}(m, n) \star \mathbf{W}_k(m - j.t, n - i.s) \tag{A.2}$$

Short Hand Notation : 2D Convolution of $\mathbf{F}$ with filter $\mathbf{W}_k$ with row-stride s and column-stride t

$$\boxed{\mathbf{C}(k, :, ;) = \mathbf{F} \star \mathbf{W}_k^{(s,t)}}$$

Bibliography

Proceedings

[5] Thierry Bertin-mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. "The million song dataset". In: *In Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR*. 2011.

[7] Sander Dieleman and Benjamin Schrauwen. "Multiscale Approaches To Music Audio Feature Learning". In: *ISMIR*. 2013.

[9] Keunwoo Choi, George Fazekas, and Mark Sandler. "Automatic tagging using deep convolutional neural networks". In: *International Society of Music Information Retrieval Conference. ISMIR*. 2016.

Articles

[3] Edith Law, Kris West, Michael Mandel, Mert Bay, and J. Stephen Downie. "Evaluation of algorithms using games: The case of music tagging". In: (2009), pp. 387–392.

[4] S. K. Kopparapu and M. Laxminarayana. "Choice of Mel filter bank in computing MFCC of a resampled speech". In: (2010), pp. 121–124. DOI: `10.1109/ISSPA.2010.5605491`.

[6] M. Slaney. "Web-Scale Multimedia Analysis: Does Content Matter?" In: *IEEE MultiMedia* 18.2 (2011), pp. 12–15. ISSN: 1070-986X. DOI: `10.1109/MMUL.2011.34`.

[8] Humphrey Eric J., Juan P. Bello, and LeCun Yann. "Feature learning and deep architectures: new directions for music informatics". In: *Journal of Intelligent Information Systems* 41.3 (2013), pp. 461–481. ISSN: 1573-7675. DOI: `10.1007/s10844-013-0248-5`.

[11] Patrik N. Juslin, Laura S. Sakka, Gonalo T. Barradas, and Simon Liljestrm. "No Accounting for Taste? Idiographic Models of Aesthetic Judgmentin Music". In: *Psychology of Aesthetics, Creativity, and the Arts* 10.2 (2016), pp. 157–170. DOI: `10.1037/aca0000034`.

Pre-Prints

[10] Keunwoo Choi, Gyorgy Fazekas, Mark Sandler, and Kyunghyun Cho. *Convolutional Recurrent Neural Networks for Music Classification*. Version 3. Dec. 21, 2016. arXiv: `1609.04243`.

Books

[1] D. O'Shaughnessy. *Speech communication: human and machine*. Addison-Wesley series in electrical engineering. Addison-Wesley Pub. Co., 1987. ISBN: 9780201165203.

[2] R.L. Allen and D. Mills. *Signal Analysis: Time, Frequency, Scale, and Structure*. Wiley, 2004. ISBN: 9780471660361.

Misc

[12] Lecture Notes. *Spectral Leakage and Windowing*. https://mil.ufl.edu/nechyba/www/__eel3135.s2003/lectures/lecture19/spectral_leakage.pdf. Online; accessed 20 March 2017.