Classification of musical genres using hidden Markov models

Sebastian Dalin-Volsing

Master's thesis
2017:E20

Faculty of Science
Centre for Mathematical Sciences
Mathematical Statistics

CENTRUM SCIENTIARUM MATHEMATICARUM

Abstract

The music content online is expanding fast, and music streaming services are in need for algorithms that sort new music. Sorting music by their characteristics often comes down to considering the genre of the music. Numerous studies have been made on automatic classification of audio files using spectral analysis and machine learning methods. However, many of the completed studies have been unrealistic in terms of usefulness in real settings, choosing genres that are very dissimilar.

The aim of this master's thesis is to try a more realistic scenario, with genres of which the border between them is uncertain, such as Pop and R&B. Mel-frequency cepstral coefficients (MFCCs) were extracted from audio files and used as a multidimensional Gaussian input to a hidden Markov model (HMM) to classify the four genres Pop, Jazz, Classical and R&B. An alternative method is tested, using a more theoretical approach of music characteristics to improve classification. The maximum total accuracy obtained when tested on an external test set was 0.742 for audio data, and 0.540 for theoretical data, implying that a combination of the two methods will not result in an increase of accuracy. Different methods of evaluation and possible alternative approaches are discussed.

Keywords: Machine Learning, Genre classification, HMM, MFCC

Acknowledgements

I would like to thank my supervisor Nader Tajvidi for supporting me in the work of this thesis, as well as providing useful articles and giving me feedback. I would also like to thank my friends and family for supporting me through the process of the project.

Contents

1 Introduction

The concept of music classification is a very wide field, and since the launch of internet, it is now more important than ever. Large audio repositories have been built, and new music keeps coming. Ways of sorting new music is of great use, because as the amount of music increases, the difficulty of finding what you are looking for increases as well.

When we listen to music, we might have an idea of what genre the music belongs to by simply comparing it to other songs we have heard from the same genre. Generally speaking, it is more likely to find music one enjoys if one is searching for music within a genre they like, rather than just picking a song at random [11]. Because of this reason, digital music services such as Spotify, AppleMusic and Youtube are in need of algorithms that sort the content of new music into genres.

So how to approach this problem? There are numerous mathematical ways to classifying objects, ranging from unsupervised methods such as clustering all the way to completely deterministic models. Which ones works the best? First off one needs to figure out what defines a musical genre. Generally, a musical genre is a grouping of music that has similar musical characteristics including, but not completely defined by instrumentation, tempo, rhythm, complexity and harmonics. By considering these kinds of variables, one should be able to group songs into their respective genres.

Another thing to consider when dealing with genres is the *fit* of the label to a certain song. Is there a typical Pop-song that is then compared to other songs, or are there different definitions of Pop, all of which are equally valid? A question of matter that occurs when talking about these labels is who puts the label on the song. Whether it is the artists themselves who place their music into a genre, or an expert working as a producer, they might have different opinions about what defines a certain genre. Additionally, mapping songs to genre is not a one-for-one relation, but one song could have influences from many different genres at once, which makes classification harder. For example, a Pop-song could have jazzy elements and therefore be labeled Jazz/Pop.

A common way of creating a classifier is to create meaningful *features* that say something about the property of the class, which is then used to train a model. The way the classifier uses the features to produce an output varies depending on the chosen method, but the idea is to try to find the combinations of features that maximizes the difference between the classes. For audio classification this would mean sampling audio files, and use the sampled data to produce features for a classifier. In order to extract meaningful features from sampled audio, one typically uses some sort of *spectral analysis*. As for classification method of the audio files, there are many different methods that can be considered. For this project, however, one specific method will be used, namely *hidden Markov models*.

1.1 Previous Research

A hidden Markov model (HMM) approach was proposed in 2001, that was capable of dealing with classification tasks of Folk music from different countries [2]. The songs used were monophonic, which makes it easier to extract information about pitches, and melodic sequences. Nevertheless it was found that HMM's are powerful in expressing differences in melodies. It has also been shown that HMMs can be used for melody recognition in monophonic children songs [4].

The same year, it was found that there are ways of classifying music genres like Rock, Classical, Techno and Jazz using *Fourier transforms* of audio clips with other machine learning algorithms such as Artificial Neural Networks (ANNs) and Learning Vector Quantization (LVQ) [19]. This opened up for many different attempts of models combining the best of signal processing and machine learning.

Plenty of research has been conducted on the choice of features for genre classification [20], [5]. It has also been shown that using Mel-Frequency Cepstral Coefficients (MFCCs) as feature space increases the classification accuracy. MFCCs are usually used within speech recognition, but can also be useful for music genre classification [21], [12].

In a study by Karpov & Subramanian in 2002, the 12 first MFCCs was used as features for a HMM. They also found that it was possible to enhance the information from the MFCCs by including a delta and acceleration coefficient for each coefficient. This gives the model a memory about past states, which improves the accuracy. The classification was made on four genres Techno/-Trance, Celtic, Classical and Rock, and they got an accuracy of 92.4% for the Techno genre, but only 72.4% for the Celtic. The reasons behind the big differences in accuracies were explained by the Techno class being so different from the others in terms of music. Celtic was misclassified for both Rock and Classical.

In later years, a study was made using Gaussian Mixture Models and HMM mixture models, to compute similarity measures between musical pieces [16]. Another study showed that ANNs outperform less complex models such as k-Nearest Neighbors and k-Means Clustering [7]. This was done using solely MFCCs, and it was shown to work fairly well to discriminate between Classical, Jazz, Metal and Pop.

Much of the research that has been made is evaluated on the GTZAN dataset[1] which consists of 1000 sample 30-second long clips of songs from 10 different genres. These genres are blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae and rock. In order to compare models to each other, it is very common to test a newly acquired model on this data set to see how it fares against the state-of-the-art models. In the last 3 years, the state-of-the-art model has been subject to change plenty of times. In 2014, new research was made on musical genre classification using new types of classifiers such as Joint Sparse Low-Rank Representation (JSLRR) [14]. Another type of newer method applied to this problem is the fuzzy-means clustering, which has been proven successful both as a classification approach [1], but also as a pre-classification step [15]. Alexandridis et al. used the fuzzy means clustering together with a radial basis functions network (RBF network) to classify MFCC features combined with spectral features. Poria et al. used the fuzzy means clustering as a rough clustering step, followed by a hard classifier using a combination of long-term features, short-term features and beat features and obtained a classification accuracy of 97.1% on the GTZAN data set.

[1] Available here.

It has been shown that the feature space can be expanded from only using timbre features as in the previous research, to a combination of intensity, pitch, timbre, tonality and rhythm [8]. Huang et al. proved in 2014 that a combination of such features can obtain a total classification accuracy of 97.2% on the GTZAN testing set. This was done by using a form of pairwise dimension reduction technique, where the genres where compared one-and-one, using an algorithm called SAHS (Self-adaptive harmony search). It was also shown that rhythm played a primary role in discriminating between genres.

There has also been research in combining acoustical features from songs with visual features obtained from spectrograms [13]. These visual features were proven to be successful in increasing performance of the classifier, by locally extracting features from sub-windows of the spectrogram.

Table 1: Comparison of the accuracy of the methods used in previous research (GTZAN data set).

Reference	**Features**	**Classifier**	**Accuracy**
[1]	MFCC + Spectral Features	Fuzzy + RBF network	74.79%
[14]	MFCC + Chroma	JSLRR	89.40%
[13]	Acoustic + Visual	SVM	89.9%
[15]	Long-term + Short-term + Beat-features	Fuzzy + SVM	97.1%
[8]	Intensity + Pitch + Timbre + Tonality + Rhythm	SAHS + Pairwise	97.2%

1.2 Question at hand

In this project, the aim is to use the MFCCs as a feature basis, together with the delta and acceleration coefficients. The idea is to perform a classification with a more realistic setting in terms of genres than what was used in Karpov & Subramanian's study, using Pop, Jazz, R&B and Classical. In reality, it is very hard to separate Pop from for example R&B, since there are not any clear borders between the genres. To visualize this, a preliminary study based on music theory was made on the four genres. This was to map out what could possibly be used as separation, and if the MFCC-feature space could be extended using a more music-theoretical approach.

2 Theory

In this section we will cover the theory required to understand the classification problems.

2.1 Music Theory

The music we are all used to hearing on the radio, is often referred to as western music. This is defined by a 12-step chromatic scale of pitches, each step called a *semitone*. A set of twelve tones is called an *octave*.

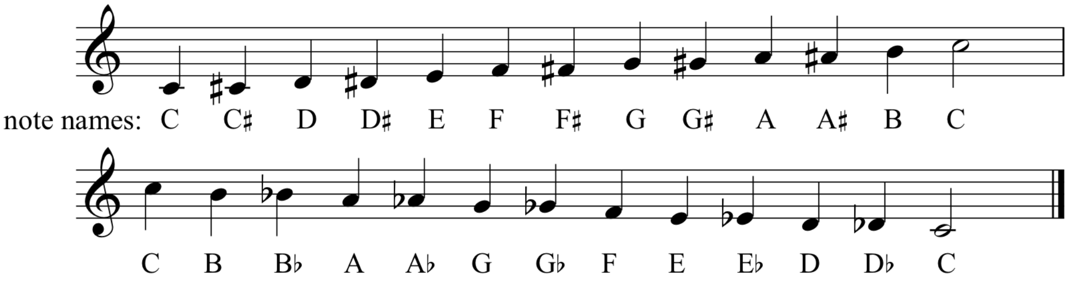

note names: C C♯ D D♯ E F F♯ G G♯ A A♯ B C
C B B♭ A A♭ G G♭ F E E♭ D D♭ C

Figure 1: Chromatic scale in C. This scale consists of all twelve semitones in an octave. This particular scale is rarely used in music composition.

These semitones are represented in the frequency domain by a particular frequency. The octaves are defined so that every semitone has a frequency equal to a multiplier of the *tonic* (the first tone of the scale). This multiplier ranges from 1 to 2, where 2 is the frequency multiplier for the tone an octave away from the tonic.

In Figure 1 we can see the chromatic scale, which is the sequence of all twelve semitones within an octave. The chromatic scale is rarely used when composing songs, but instead one uses a heptatonic scale, which only consists of seven out of these twelve tones. There are several different heptatonic scales (*modes*), and the most commonly used ones are *major* and *minor*. The difference between these modes are the tones that build the scale. For example the major and minor scale in C is shown in Figure 2. The scales can be compared by looking at the steps between the tones. A single step is considered *whole* if it is two semitones long, and *half* for one semitone. The big difference between the major and minor scale is the third tone, being 4 semitones from the tonic in the major scale, but only 3 semitones for the minor scale. This is often referred to as the *triad*, since it is the third tone of the scale.

Every song can be put into one of these scales, based on the tones used to build the melodies. However, since the tonic is not the same for all songs, they need to be adjusted. A song's scale is defined not only by the mode, but also by the *key signature*, which is a set of ♭'s and ♯'s put together with the clef at the beginning of the score, to denote what tone the scale starts at. In Figure 2, the key signature is C, since it has no extra symbols.

A tone is often also assigned a number corresponding to the octave of the tone. The difference between C2 and C3 is one octave, and one can therefore easily understand that the frequency of

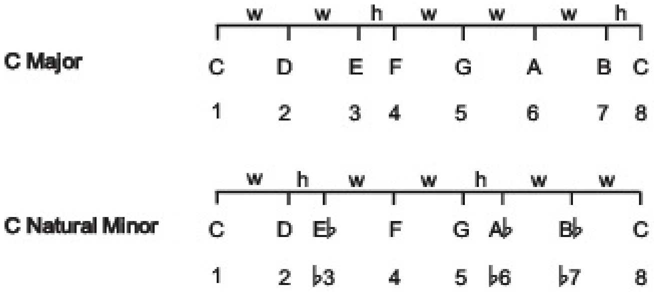


Figure 2: Comparison of the major and minor scale in C. A *w* stands for a whole step and *h* stands for a half step.

C2 is half of C3. The most common tone used for tuning is A4, which is 440 Hz in standard concert tuning.[2]

In order to transcribe a song to sheet music, one also needs to know about *chords*. A chord is a collection of tones (usually three or more), that together make up a harmony. Chords are denoted as one or more letters with superscripted or subscripted numbers. Some common chords are major (C), minor (Cm), major seventh (C^7) and minor seventh (Cm^7), here expressed as their corresponding chord in key signature C.

Another way of denoting chords irrespective of key signature is roman numeral analysis, where each chord gets denoted in relation to the tonic. In the key of C, the chord C would get denoted as I. F would be denoted as IV, since it is the fourth chord of the scale. Minor chords are denoted with lower case roman numerals, such as i or vi.

As for melodies, the chords of a song are dependent on the mode of the song. In order to be able to compare songs with different modes, we introduce the concept *relative minors*, which means that a major key and a minor key with the same key signature has the same scale. The relative minor to C is Am, which means a song in Am and a song in C should have the same set of basis chords to use, even though the tonics are different. We take advantage of this when transposing songs.

2.1.1 Genres

What kinds of differences do we expect to see in the different genres? The definitions of the genres that are considered in this analysis (Pop, Jazz, Classical and R&B) is quite ambiguous. Usually, one can hear the difference between for example Pop music and Classical music, but what is the underlying properties of the genres that provide the basis for the difference?

[2] One may choose to tune their instrument in another way, using for example 442 Hz as A4, which slightly alters the scale. A good example of this is the various philharmonic orchestras around the world, which use between 440-445 Hz for A4.

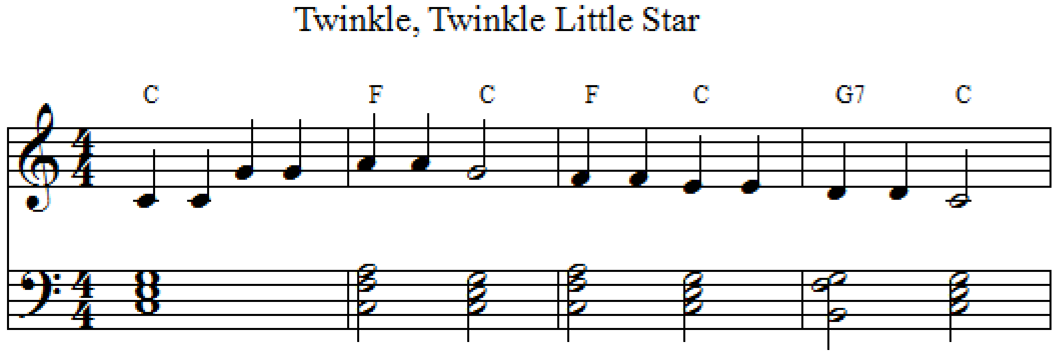


Figure 3: Chords of the first part of Twinkle Twinkle Little Star. One can follow the melody in the upper part, and see the chords as collection of tones in the bottom part of the score. The chords are often translated as can be seen on the top of the melody. The key signature is C, and the mode is major.

Pop music is a very wide genre, appealing to the larger mass of listeners. Pop music is often said to have simple chord progressions, usually I-V-vi-IV or vi-IV-I-V.[3] Usually Pop songs are repetitive, with recurring choruses with catchy melodies. The instrumentation varies heavily, from acoustic guitars, to electronic music.

Jazz music is mostly known for its complex chord progressions, using seventh chords, extended chords and borrowed chords frequently. A common chord progression is the ii^7-V^7-$\text{I}^\triangle$. Instrumentation is commonly piano, with a set of brass instruments such as trumpets, trombones and saxophones in addition to the drums, bass guitar and electric guitar.

Classical music is very melodic, and usually orchestrated, or played on a solo instrument such as piano. A common chord progression seen in classical music is the *descending-fifth*-progression, which means the chords always move five steps down the scale. An example of this could be vi-II-V-I.

R&B-music is very much like Pop a very wide genre, except it features rap as well as sung melodies. R&B music are usually heavy on the beats and bass, with electronic melodies played from a music producing software. It is unclear if there are any specific chord progressions that characterizes this genre.

2.2 Spectral analysis

In order to access information about the audio track, one needs to use spectral analysis. Using spectral analysis, one can decompose the song into its frequency components, finding the power of different frequencies in different points in time. By combining this method with what is known about music, one can now start to estimate the frequencies, and since we have a map from frequencies to their respective tones, we can estimate scales and melodies. However, analyzing a

[3] More commonly known as the four-chord progression.

song spectrally can be hard, due to difference in instrumentation.

A tone is often described to have three different properties:

- *pitch*; the frequency of the respective tone. A4 is 440 Hz in standard concert tuning for example.
- *amplitude*; the strength of the tone, which differentiates tones that are played from tones that are not.
- *timbre*; quality of the tone. This property is useful for finding differences in instrumentation between songs.

All three of these properties are useful for genre classification of songs, but they are not equally useful. Amplitude and pitch are often not specific for genres, since the key signature and scale of a song are more or less random. However, these properties are useful when considering changes in chords for example, which is used in chord estimation. Also, when combined, these two properties can say something about the general volume of the track, as well as the volume and balance of the track in the different frequency bands.[4] The last property, timbre, can be used for genre classification to a certain extent, knowing that some instruments are more heavily featured in certain genres. An example of this is the piano, which is common in classical music, whereas in R&B, they are not.

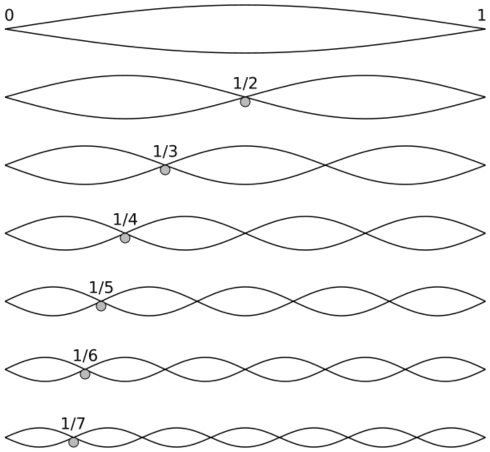


Figure 4: Harmonic series of a tone. When a tone is initiated, standing waves are created, which amplifies the tones at frequency multipliers of the fundamental frequency. These frequency multipliers are called the harmonic series (overtone series).

Timbre works by different instruments having different *overtone* structure. The overtones are multipliers of the original tone that resonate due to string vibrations (string instruments), or air

[4] Also known as equalization. This is the process of filtering the audio track to enhance or suppress certain frequencies. Music producers tend to equalize differently depending on the genre of the song.

vibrations (wind instruments). When a tone is initiated, the corresponding system starts to vibrate at the *fundamental frequency*[5] of the tone. In addition, standing waves are created at multipliers of the fundamental frequency, which results in a number of audible tones, at set locations from the fundamental frequency. This is illustrated in Figure 4, where the standing wave of a string instrument can be seen as multipliers of the fundamental frequency. The locations of these frequencies are described by the *harmonic series*, which is the sequence of tones that accompany the fundamental frequency to make up a sound. The harmonic series can be seen in Figure 5.

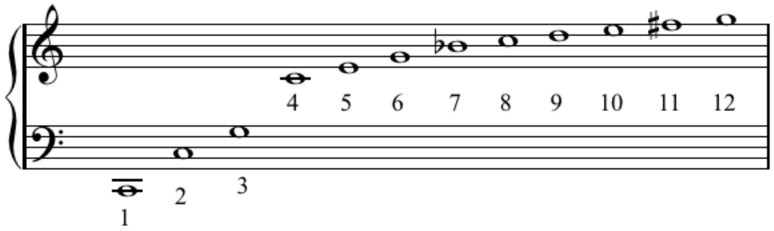
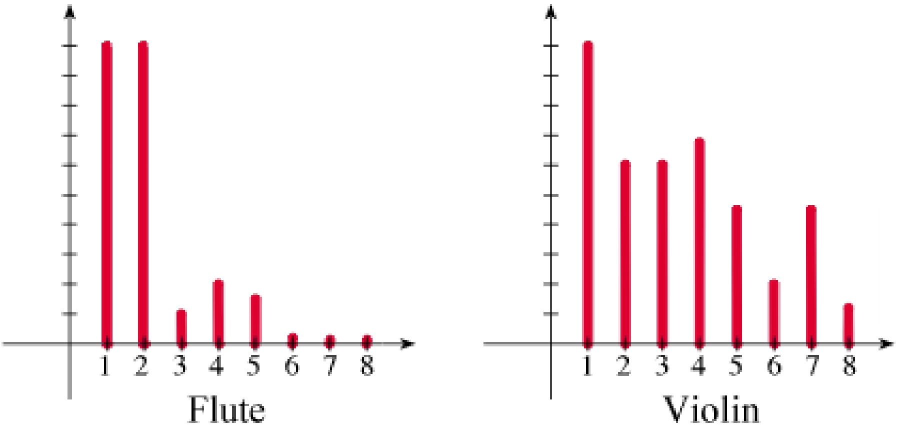


Figure 5: First twelve tones of the harmonic series (illustrated in the key of C, starting at C2). One can note that the first overtone (second tone of the series) is the octave at $2f_0$ (C3), followed by the second overtone at $3f_0$, which is the fifth tone of the scale (in this case G3).



Adapted from James Stewart, *Essential Calculus: Early Transcendentals*

Figure 6: Difference in timbre for a tone played on a flute and a violin. It is noticeable that there are big differences in overtone structure between the two instruments. The flute has two distinct overtones and the rest are quite weak, whereas the violin has a more complex overtone structure.

What differs the instruments from each other is the amplitude of the different overtones. For example the spectral content of a flute is very different from a violin in terms of overtones. In Figure 6, we can view the amplitude of the different overtones of the same fundamental fre-

[5] For example 440 Hz for the tone A4.

quency. The flute has two strong first overtones, and quite weak overtones following, which results in a pure sound. The violin has a more complex structure, with one distinct overtone and several underlying overtones, which make up a rich sound. This difference, as well as obvious differences in sound such as striking a key on the piano, blowing into a flute and then breathing in between the notes, or hitting a drum together account for a good basis for classification.

In order to access this information from the audio signal, we use the *power spectral density* (PSD). This resembles the distribution of variance across the different frequencies of the signal. For a discrete data signal[6] $x(t)$, an estimate of the PSD is computed as:

$$\hat{R}_x(f) = \frac{1}{n}|\mathcal{X}(f)|^2,$$

where

$$\mathcal{X}(f) = \sum_{t=0}^{n-1} x(t)e^{i2\pi ft}$$

is the Fourier transform of the data vector $x(t)$. This estimate is called the periodogram.

Often the periodogram is not enough to capture all properties of the signal, since it is very noisy. Normally what one does is to multiply the data vector with a window function $w(t)$ in every step of the Fourier transform, where this window function can have many different appearances.

2.3 MFCC

A common way of extracting information from an audio signal is to consider the Mel-Frequency Cepstral Coefficients (MFCCs). These coefficients are widely used in speech recognition, and music modelling, since they can be used as a feature basis for classification purposes.

One of the reasons why MFCCs is so popular is the scaling of the spectrum into the mel-frequency plane. This was first implemented in 1937 by Stevens, Volkmann and Newman, where they found that humans interpret the chromatic scale as linear, when it really is exponential [17]. Therefore, by putting a logarithmic function on the frequencies, we can obtain a new scale which is more similar to what we perceive. This scale is now more commonly known as the Mel-scale.

The process of creating MFCC features goes as follows [12]:

1. Divide the signal into small frames of $\sim$ 25 ms so that stationarity can be assumed.
2. Take the discrete Fourier transform of the frame.
3. Take the logarithm of the amplitude spectrum.
4. Mel-scaling and smoothing of spectrum to emphasize meaningful frequencies.
5. Discrete cosine transformation.

[6] Since the data is sampled, we are always dealing with discrete data. There are ways of dealing with continuous data as well, but in this report we will only consider discrete data vectors.

The signal x_n is first filtered using a pre-emphasis coefficient α by the following equation:

$$\tilde{x_n} = x_n - \alpha x_{n-1}$$

Then the signal is chunked into smaller segments (frames) and a windowed periodogram spectrum $\hat{R}_x(f,t)$ is computed for each of the segments. The spectrum is then filtered by a triangular filterbank which extracts information about certain frequencies, which are uniformly spaced on the mel-scale. The transfer function between frequency and mel-scale is the following:

$$m = 2595\log_{10}(1+\frac{f}{700}) = 1127\ln(1+\frac{f}{700}),$$

and inversely:

$$f = 700(10^{\frac{m}{2595}} - 1) = 700(e^{\frac{m}{1127}} - 1),$$

The transfer function can be seen in Figure 7.

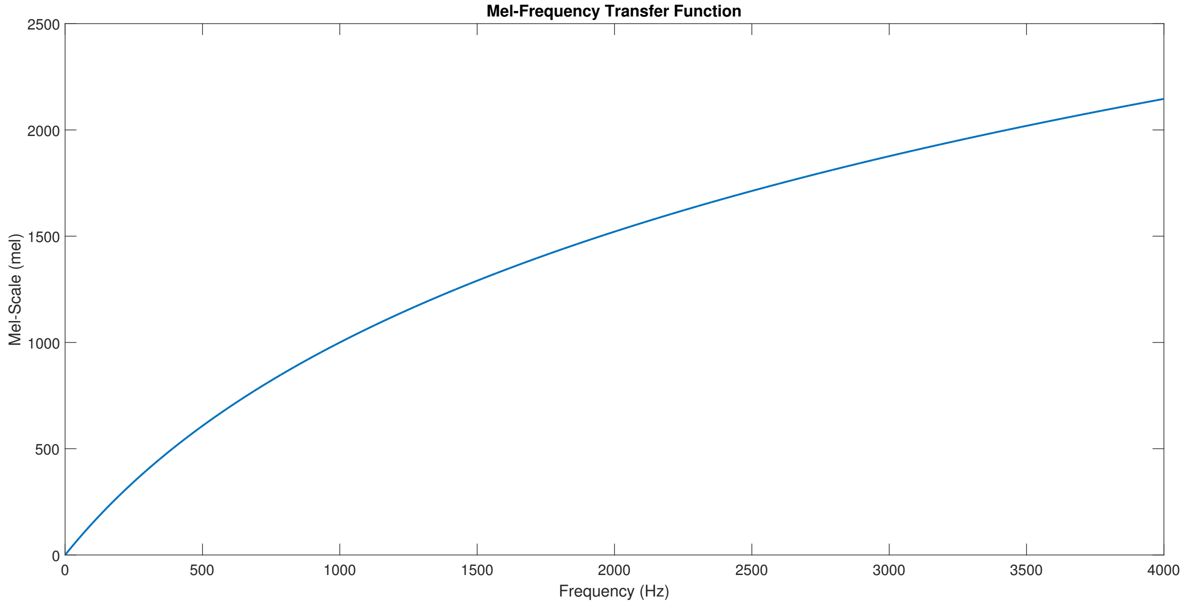


Figure 7: Transfer function between frequency and mel-scale. For low frequencies, the relationship is approximately linear, but that becomes logarithmical over a certain threshold. The mel-scale is closer to how people perceive pitch.

Using the extracted information from the specific *mels* (transformed frequencies), we can now compute the MFCCs by using the log filterbank amplitudes $\{r_j\}$ and performing a Discrete Cosine Transform (DCT) [22]:

$$c_i = \sqrt{\frac{2}{M}} \sum_{j=1}^{M} r_j \cos(\frac{\pi i}{M}(j-0.5)),$$

where M is the number of filterbank channels. The coefficients c_i are then *liftered* (filtered cepstrally) by the following relation:

$$\tilde{c}_i = (1 + \frac{L}{2} \sin \frac{\pi(i-1)}{L})c_i,$$

for $i \in \{1, 2, \dots, N\}$, where L is a cepstral lifter parameter, N is the total number of coefficients, and i is the index for the particular coefficient.

The obtained MFCCs are now used as features for a certain time frame of 25 ms. These coefficients can be enhanced by introducing a delta-value $\Delta_i(t)$ for each coefficient $\tilde{c}_i(t)$, as well as an acceleration value $a_i(t)$ [10]. These are calculated by:

$$\Delta_i(t) = \tilde{c}_i(t) - \tilde{c}_i(t-1)$$

and

$$a_i(t) = \Delta_i(t) - \Delta_i(t-1),$$

where t is the considered time frame of 25 ms. The first and last two values are then removed from the feature space to have the coefficient vector of all time bins $\tilde{\mathbf{C}}_i$ of equal length as the delta and acceleration vectors $\mathbf{\Delta}_i$ and $\mathbf{a}_i$.

The final feature vector for each time frame results in:

$$\mathbf{o}_t = \begin{pmatrix} c_1(t) \\ c_2(t) \\ \vdots \\ c_N(t) \\ \Delta_1(t) \\ \Delta_2(t) \\ \vdots \\ \Delta_N(t) \\ a_1(t) \\ a_2(t) \\ \vdots \\ a_N(t) \end{pmatrix},$$

where N is the number of coefficients used.

2.4 Spectral features

Another common way of extracting more basic information from an audio signal is to compute the measures **spectral centroid** (SC), **spectral rolloff** (SR), and **spectral flux** (SF). These measures are defined as follows:

$$\text{SC} = \frac{\sum_{f=0}^{N-1} f \hat{R}_x(f)}{\sum_{f=0}^{N-1} \hat{R}_x(f)},$$

where f is the frequency corresponding to the calculated spectrum in $\hat{R}_x(f)$. Spectral centroid can be seen as the average frequency weighted with the estimated power of the corresponding frequency.

Spectral rolloff is the frequency at which 85% of the total power is below that frequency. This is calculated using a cumulative sum of the spectrum $\hat{R}_x(f)$.

Spectral flux is defined as the difference in spectrum for that frame with the frame before.

These three measures were tried alongside of the MFCCs, to see if it could improve the final classification once the parameters for the MFCC had been chosen.

2.5 HMM

Finally, the actual modelling was done using hidden Markov models (HMMs), which is a powerful tool to deal with sequences of observations. Hidden Markov models use hidden *states* to describe the properties of the sequence in a given time. These states each have their own probability distribution associated to them, giving the model the ability to switch distributions as the sequence progresses.

A discrete hidden Markov model can be fully defined by the number of hidden states n, the static state transition probability matrix $\mathbf{P}$, the observation symbol probability distribution $\mathbf{B}$ and the initial state distribution $\boldsymbol{\pi}$. The matrices $\mathbf{P}$, $\mathbf{B}$ and $\boldsymbol{\pi}$ are defined as:

$$\begin{aligned}
&\mathbf{P} = \{p_{ij}\}, \quad p_{ij} = \mathbb{P}(\mathcal{X}_{t+1} = j | \mathcal{X}_t = i), \quad i, j \in \{1, 2, \ldots, n\} \\
&\mathbf{B} = \{b_i(k)\}, \quad b_i(k) = \mathbb{P}(o_t = v_k | \mathcal{X}_t = i) \\
&\boldsymbol{\pi} = \{\pi_i\}, \quad \pi_i = \mathbb{P}(\mathcal{X}_0 = i),
\end{aligned}$$

where n is the number of states in the model, $P = \{1, \ldots, n\}$ is the space of possible states, and V_k, $k \in \{1, \ldots, K\}$ is the space of possible outputs. For a continuous observation space the possible outputs are determined by the probability distribution given by the state.

A *Markov chain* is defined as a sequence of random variables $\{\mathcal{X}_t\}$, $t \in \{0, 1, 2, \ldots\}$ where, given the present state at time t, $\mathcal{X}_t$, the past and the present are independent. This is due to the Markov property, which says that the probability of the next state is only dependent on the previous state. This is more formally defined as:

$$\mathbb{P}(\mathcal{X}_{t+1} \in A | \mathcal{X}_0 = x_0, \ldots, \mathcal{X}_t = x_t) = \mathbb{P}(\mathcal{X}_{t+1} \in A | \mathcal{X}_t = x_t),$$

for a measurable set A in the probability space.

The sequence of random variables is often dependent on a *transition matrix* $\mathbf{P}$, which contains the information about the transition probabilities between the states i and j, where $i, j \in \{1, 2, \ldots, n\}$. The matrix is defined as:

$$\mathbf{P} = \{p_{ij}\}, \quad i, j \in \{1, 2, \ldots, n\},$$

where $p_{ij} = \mathbb{P}(\mathcal{X}_{t+1} = j | \mathcal{X}_t = i)$. The rows of **P** indicate the probability distributions of the states. These probability distributions are in this specific case assumed to be discrete, but they work equally well for continuous distributions.

2.5.1 Example of Markov chains

Imagine a situation where the weather can be modelled using Markov chains. In a very simplified example, today's weather only depend on the weather on the day before. There are three different states: 1 = sunny, 2 = raining and 3 = cloudy. The sequence of days: $\{1, 1, 3, 2\}$ explains that it was sunny the first two days, but on the third day it was cloudy, and on the fourth day it rained. We create a transition matrix **P** for the states:

$$\mathbf{P} = \begin{array}{c} \\ \text{Sunny} \\ \text{Rainy} \\ \text{Cloudy} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Sunny} & \text{Rainy} & \text{Cloudy} \end{array} \\ \begin{bmatrix} 0.7 & 0.1 & 0.2 \\ 0.5 & 0.15 & 0.35 \\ 0.2 & 0.6 & 0.2 \end{bmatrix} \end{array}$$

We can see that if it is sunny today, it is a probability of $p_{1,2} = 0.1$ of it being rainy tomorrow. When calculating the probability for the sequence $\{1, 1, 3, 2\}$, we use the transition matrix to multiply the probabilities (assuming Markov property):

$$\mathbb{P}(\{1, 1, 3, 2\}) = p_{1,1} \cdot p_{1,3} \cdot p_{3,2} = 0.7 \cdot 0.2 \cdot 0.6 = 0.084.$$

2.5.2 Example of Hidden Markov chains

We can extend the above scenario to hidden Markov models by considering an ice cream salesman. If we assume the ice cream sales per day is dependent only on the weather, we can write a probability matrix **B** for the number of ice creams sold during that day.

$$\mathbf{B} = \begin{array}{c} \\ \text{Sunny} \\ \text{Rainy} \\ \text{Cloudy} \end{array} \begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{bmatrix} 0 & 0.1 & 0.9 \\ 0.7 & 0.2 & 0.1 \\ 0.3 & 0.4 & 0.3 \end{bmatrix} \end{array}$$

Imagine now that a person observes the number of ice creams sold every day, but has no idea what the weather is. For that person, this is a hidden Markov model. Often we want to use the observed data (sold ice creams) to determine the sequence of states (weather).

2.5.3 Continuous setting

Considering music, the Markov property does not hold completely. It's a known fact that music relies heavily on recurring melodies, choruses, and repeating beats. Moreover, the observations obtained by the MFCCs are continuous, and multi-dimensional. Having said this, one can still use Markov chains in modelling. Using the properties of HMMs combined with the calculated

MFCCs, one can make the assumption that the observed values come from a normal distribution, and can therefore replace the observation probability distribution matrix **B** with the following:

$$
\begin{aligned}
&\mathbf{P} = \{p_{ij}\}, \quad p_{ij} = \mathbb{P}(\mathcal{X}_{t+1} = j | \mathcal{X}_t = i), \quad i, j \in \{1, 2, \ldots, n\} \\
&\mathbf{B} = \{b_i(\mathbf{o}_t)\} = \mathbb{P}(\mathbf{o}_t | \mathcal{X}_t = i) \sim \mathbb{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \\
&\boldsymbol{\pi} = \{\pi_i\}, \quad \pi_i = \mathbb{P}(\mathcal{X}_0 = i),
\end{aligned}
$$

where now the observation vector $\mathbf{o}_t$ in each time instance t is assumed to be normally distributed with a mean vector $\boldsymbol{\mu}_i$, and a covariance matrix $\boldsymbol{\Sigma}_i$, $i \in \{1, 2, \ldots, n\}$. The size of $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ depend only on the size of the feature space. In Figure 8 we can see an example of a four-state HMM with different one-dimensional distributions. The different distributions have different means and variances, depending on the training procedure.

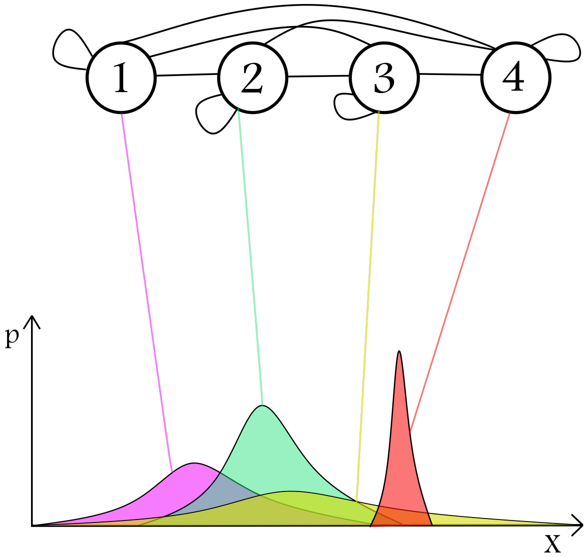


Figure 8: Example of a four-state HMM with continuous one-dimensional distributions. The observed value depends on the current state of the process. In this setting, the training process of the modelling rescales the distributions according to the observed values.

2.6 Training the models

In this setting, four different models will be computed, trained on the respective genre data sets. The four models will have distinct state transition matrices, initial state distributions and distribution parameters, and these are then used to evaluate new songs for genre membership.

So how is each model trained? The parameters that define the model are now the state transition matrix **P**, the initial state distribution $\boldsymbol{\pi}$ and the distribution parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ for all states i. For simplicity sake we call the parameters $\boldsymbol{\lambda} = \{\mathbf{P}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$. In order to find the correct parameters $\boldsymbol{\lambda}$, we need to be able to answer some questions:

1. How do we compute the likelihood $\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda})$ of observed features given parameters $\boldsymbol{\lambda}$?
2. Given the model, and a set of observations, how do we update the parameters $\boldsymbol{\lambda}$ to improve the likelihood?
3. Given the model, and a set of observations, how do we compute the most likely state sequence in the model that produced the observations?

The answer comes in three algorithms, the **forwards-backwards algorithm**, the **Baum-Welch algorithm**, and the **Viterbi algorithm** [9].

2.6.1 Forwards-backwards algorithm

> Question 1: How do we compute the likelihood $\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda})$ of observed features given parameters $\boldsymbol{\lambda}$?

First we need to use the law of total probability, by setting.

$$\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda}) = \sum_{\text{All } \mathcal{X}} \mathbb{P}(\mathcal{O}, \mathcal{X}|\boldsymbol{\lambda}),$$

where $\mathcal{X}$ is the complete state sequence. We can rewrite one observed sequence of states as $\mathbf{x}_1^T$, and the observed features as $\mathbf{o}_1^T$, giving the following expression:

$$\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda}) = \sum_{\text{All } \mathbf{x}_1^T} \mathbb{P}(\mathbf{o}_1^T, \mathbf{x}_1^T|\boldsymbol{\lambda})$$

We define the forward likelihood for state j as:

$$\alpha_t(j) = \mathbb{P}(\mathbf{x}_1^t, \mathbf{o}_1^t|\boldsymbol{\lambda}),$$

If we apply the Markov property to the equation above, we get that:

$$\alpha_t(j) = \sum_{i=1}^{n} \alpha_{t-1}(i)\mathbb{P}(x_t = j|x_{t-1} = i, \boldsymbol{\lambda})\mathbb{P}(\mathbf{o}_t|x_t = j, \boldsymbol{\lambda}),$$

where the probability of current state x_t depends only on the previous state x_{t-1} and the current observation o_t depends only on current state [6]. We can now recursively compute the forward likelihood by the following steps:

Initialize the forwards procedure by setting:

$$\alpha_1(i) = \pi_i b_i(\mathbf{o}_1), \quad \text{for } i \in \{1, 2, \ldots, n\},$$

then repeat for $t = 2, 3, \ldots, T$:

$$\alpha_t(j) = \sum_{i=1}^{n} \alpha_{t-1}(i) p_{ij} b_j(\mathbf{o}_t), \quad \text{for } i \in \{1, 2, \ldots, n\},$$

finalise:

$$\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda}) = \sum_{i=1}^{n} \alpha_T(i).$$

We then consider the backwards algorithm, where the backward likelihood is defined as:

$$\beta_t(i) = \mathbb{P}(\mathbf{o}_{t+1}^{T}|\mathcal{X}_t = i, \boldsymbol{\lambda})$$

The backwards procedure is initialized by:

$$\beta_T(i) = 1, \quad \text{for } i \in \{1, 2, \ldots, n\}$$

Then repeat for $t = T-1, T-2, \ldots, 1$:

$$\beta_t(i) = \sum_{j=1}^{n} p_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j) \quad \text{for } i \in \{1, 2, \ldots, n\}$$

Finally

$$\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda}) = \sum_{i=1}^{n} \pi_i b_i(\mathbf{o}_1) \beta_1(i)$$

2.6.2 Baum-Welch algorithm

Now that we can calculate the likelihood of a set of observations, given the model, we want to consider the second question:

> Question 2: Given the model, and a set of observations, how do we update the parameters $\boldsymbol{\lambda}$ to improve the likelihood?

In order to update the parameters and guarantee an increase in likelihood we define the following measures:

$$\begin{aligned}\gamma_t(i) &= \mathbb{P}(\mathbf{x}_t = i|\mathcal{O}, \boldsymbol{\lambda}) \\ &= \frac{\mathbb{P}(\mathbf{x}_t = i, \mathcal{O}|\boldsymbol{\lambda})}{\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda})} \\ &= \frac{\alpha_t(i)\beta_t(i)}{\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda})}\end{aligned}$$

$$\begin{aligned}\xi_t(i,j) &= \mathbb{P}(\mathbf{x}_{t-1} = i, \mathbf{x}_t = j|\mathcal{O}, \boldsymbol{\lambda}) \\ &= \frac{\mathbb{P}(\mathbf{x}_{t-1} = i, \mathbf{x}_t = j, \mathcal{O}|\boldsymbol{\lambda})}{\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda})} \\ &= \frac{\alpha_{t-1}(i) p_{ij} b_j(\mathbf{o}_t) \beta_t(j)}{\mathbb{P}(\mathcal{O}|\boldsymbol{\lambda})}\end{aligned}$$

$\gamma_t(i)$ denotes the likelihood of occupying state i at time t. $\xi_t(i,j)$ is a transition likelihood between states i and j at time t.

The optimal update $\hat{\boldsymbol{\lambda}} = \{\hat{\mathbf{P}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}\}$ to the model parameters $\boldsymbol{\lambda}$ is given by the following formulas:

$$\begin{aligned}
\hat{\mathbf{P}} &= \{\hat{p_{ij}}\} = \frac{\sum_{t=2}^{T} \xi_t(i,j)}{\sum_{t=2}^{T} \gamma_{t-1}(i)} && \text{for } i,j \in \{1,2,\ldots,n\} \\
\hat{\boldsymbol{\pi}} &= \{\hat{\pi_i}\} = \gamma_1(i) && \text{for } i \in \{1,2,\ldots,n\} \\
\hat{\boldsymbol{\mu}} &= \{\hat{\boldsymbol{\mu}}_i\} = \frac{\sum_{t=1}^{T} \gamma_t(i)\mathbf{o}_t}{\sum_{t=1}^{T} \gamma_t(i)} && \text{for } i \in \{1,2,\ldots,n\} \\
\hat{\boldsymbol{\Sigma}} &= \{\hat{\boldsymbol{\Sigma}}_i\} = \frac{\sum_{t=1}^{T} \gamma_t(i)(\mathbf{o}_t - \hat{\boldsymbol{\mu}}_i)(\mathbf{o}_t - \hat{\boldsymbol{\mu}}_i)'}{\sum_{t=1}^{T} \gamma_t(i)} && \text{for } i \in \{1,2,\ldots,n\}
\end{aligned}$$

It can be shown that re-estimation increases the likelihood of the training data for the new model parameters $\hat{\boldsymbol{\lambda}}$:

$$\mathcal{P}(\mathcal{O}_{\text{train}}|\hat{\boldsymbol{\lambda}}) \leq \mathcal{P}(\mathcal{O}_{\text{train}}|\boldsymbol{\lambda})$$

Albeit the algorithm guarantees to not drop in likelihood for each subsequent iteration, it does not guarantee to converge to a global maximum.

2.6.3 Viterbi algorithm

When the model is now estimated, and we have observed features saved for testing the models, we want to answer the final question:

> Question 3: Given the model, and a set of observations, how do we compute the most likely state sequence in the model that produced the observations?

This is done by using the **Viterbi algorithm**:

Given observations $\mathcal{O} = \{\mathbf{o}_1, \ldots, \mathbf{o}_T\}$, find the HMM state sequence $\mathcal{X} = \{x_1, \ldots, x_T\}$ that has the greatest likelihood. We denote the optimal state sequence as:

$$\mathcal{X}^* = \underset{x}{\text{argmax}}\ \mathcal{P}(\mathcal{O}, \mathcal{X}|\boldsymbol{\lambda}),$$

where

$$\begin{aligned}
\mathcal{P}(\mathcal{O}, \mathcal{X}|\boldsymbol{\lambda}) &= \mathcal{P}(\mathcal{O}|\mathcal{X}, \boldsymbol{\lambda})\mathcal{P}(\mathcal{X}|\boldsymbol{\lambda}) \\
&= \pi_{x_1} b_{x_1}(\mathbf{o}_1) \cdot \prod_{t=2}^{T} p_{x_{t-1},x_t} b_{x_t}(\mathbf{o}_t)
\end{aligned}$$

The Viterbi algorithm goes as follows (for $i \in \{1,2,\ldots,n\}$):

$$\begin{aligned}
\delta_1(i) &= \pi_i b_i(\mathbf{o}_1) \\
\psi_1(i) &= 0
\end{aligned}$$

Recur for $t = 2, \dots, T$:

$$\begin{aligned} \delta_t(j) &= \max_i \ \delta_{t-1}(i) p_{ij} b_j(\mathbf{o}_t) \\ \psi_t(j) &= \underset{i}{\operatorname{argmax}} \ \delta_{t-1}(i) p_{ij} \end{aligned}$$

Finalise:

$$\begin{aligned} \mathcal{P}(\mathcal{O}, \mathcal{X}^* | \boldsymbol{\lambda}) &= \max_i \ \delta_T(i) \\ \mathbf{x}_T^* &= \underset{i}{\operatorname{argmax}} \ \delta_T(i) \end{aligned}$$

Trace back, for $t = T-1, T-2, \dots, 1$:

$$\begin{aligned} \mathbf{x}_t^* &= \psi_{t+1}(\mathbf{x}_{t+1}^*) \\ \mathcal{X}^* &= \{\mathbf{x}_1^*, \mathbf{x}_2^*, \dots, \mathbf{x}_T^*\} \end{aligned}$$

2.6.4 Initial values

A common problem with using the Baum-Welch algorithm to improve the parameters is that it is heavily influenced by the accuracy of the initial guess [3]. Since the algorithm only aims to improve the log-likelihood, it does not check if the converged point is a local maxima or a global maxima. It is therefore important to find an initial guess that is reasonable, as opposed to picking initial parameters at random.

The initial parameters $\boldsymbol{\lambda}_0 = \{\mathbf{P}_0, \boldsymbol{\pi}_0, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\}$ are set the following way:

As $\mathbf{P}_0$ and $\boldsymbol{\pi}_0$ do not influence the convergence as much, we initialize these parameters as random stochastic matrices of sizes (n x n) and (1 x n), respectively.

Choosing $\boldsymbol{\mu}_0$ and $\boldsymbol{\Sigma}_0$ is a bit trickier, and they are therefore initialized using **K-means clustering**. This clustering algorithm works as follows:

1. Pick $\boldsymbol{\mu}_0 = \{\boldsymbol{\mu}_{0,i}\}$ at random for $i \in \{1, 2, \dots, n\}$.
2. Assign all observations membership to a class (one of the n state distributions) according to the closest vector to the mean.
3. Compute new $\hat{\boldsymbol{\mu}}_0 = \{\hat{\boldsymbol{\mu}}_{0,i}\}$ from the members of each class $i \in \{1, 2, \dots, n\}$.
4. Repeat from 2 until the mean does not change (convergence).

$\boldsymbol{\Sigma}_0 = \{\boldsymbol{\Sigma}_{0,i}\}$ is then computed as a sample covariance matrix of the members of class i, for all $i \in \{1, 2, \dots, n\}$.

2.6.5 Evaluating the model

When the models have been trained using a fraction of the data set as training data, the models must then be evaluated in some way to assess the accuracy. This is done by running the testing data through the models, and comparing to the ground truth labels. The likelihoods of the observations given the models $\mathbb{P}(\mathbf{o}_1^T|\boldsymbol{\lambda})$ are given by the forwards-backwards algorithm, as explained in Section 2.6.1.

The likelihoods obtained for the different genres are then compared, and the testing data is given a label $(l|\mathbf{o}_1^T)$ according to the model with the maximum likelihood:

$$l|\mathbf{o}_1^T = \underset{\text{genre}}{\text{argmax}}\ \mathbb{P}(\mathbf{o}_1^T|\boldsymbol{\lambda}_{\text{genre}}),$$

where genre $\in$ {Pop, Jazz, Classical, R&B}

3 Methodology

In this section, the work flow that was used in the analysis will be presented.

3.1 Data Collection

The pre-analysis methods that were conducted was divided into data collection and data processing. This subsection will explain how the data was collected, and sorted.

3.1.1 Transition Data

First off, data of chord transitions was collected from the website Hooktheory, which has information about popular songs from different genres. The four genres were decided upon, and then the data collection process started. For every song, several different variables were collected, like chord progression, melody, complexity, genres and subgenres, tempo, mode and number of different chords. An explanation of variables can be found in the appendix. The variables were collected into a .CSV-file and then imported into MATLAB. A total of 100 samples per genre were taken, 400 in total.

3.1.2 Audio Data

For the audio part, 100 songs from each genre were collected from the website Jamendo, which is a music streaming and downloading website where users can upload their music for sharing and promoting. The labels for genres are set by the community, and by the artists themselves, making it somewhat consistent with how songs are normally put in genres. However, the songs were checked before entered in the model, since the classification accuracy is highly dependent on a correct label. Initially, the 100 songs chosen for each genre were the 100 most played tracks for that specific genre, trying to maximize the probability of the genre label being correct.

The tracks were all in .mp3-format, with a sample rate of 44.1 kHz.

3.2 Data Processing

This subsection will consider how the data was processed in order to complete the analysis.

3.2.1 Transition Data

The chord transitions was transposed to the same key, in order to make them comparable. All major songs were transposed in to the key of C, and all minor songs into Am. In this way they are not only comparable with other songs of the same mode, but also with each other. Then the data was divided into training and testing data, and a 60 x 60 sample transition matrix was created from the one-step chord transitions for every genre. In this way, one could find chord transitions that separates the different genres.

In order to find more information about the sequences of chords, such as recurring themes and chord progressions, higher order transition matrices were created for each genre. These transition matrices contain information about the chord sequence several steps back, which gives the model the ability to memorize what has happened before. Also, a reverse transition matrix was created, to find information about the probability of the chord before. This was useful to find separation between the genres.

In the same way as the chords, all melodies were transposed according to C, following the corresponding mode. Then a 12 x 12 sample transition matrix was made for the melodies as well, for every genre.

3.2.2 Audio Data

From every audio file, 100 seconds of music was extracted from the center of the clip. This led to the restriction of the audio files having to be longer than 1m 40s. The songs were all in stereo format, hence they were transformed to mono by simple averaging over the channels. Then these 100 seconds were partitioned into 10 clips of 10 seconds each. These clips would serve as the basis for feature extraction. In Figure 9 we can see an example clip of 10 seconds from a song.

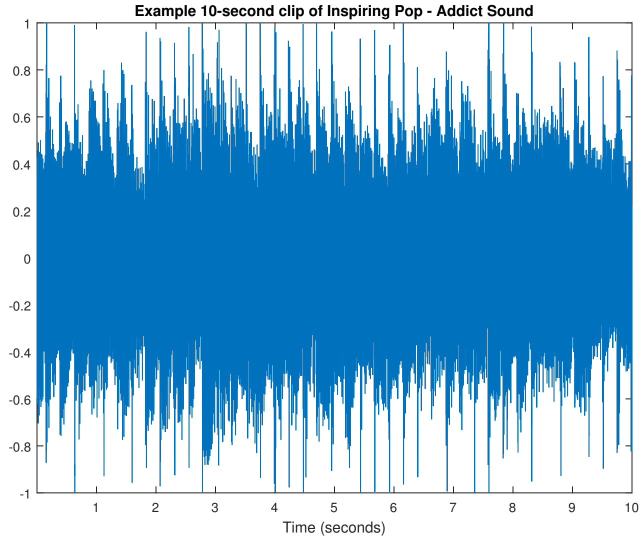


Figure 9: Example of 10-second clip used as feature basis. This is then chunked up into frames of 25 ms where spectrum is computed.

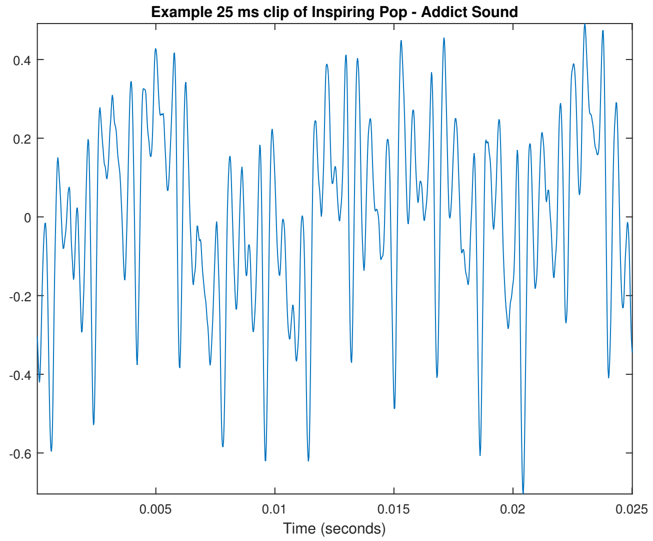


Figure 10: Example of a 25 ms long audio clip. It can be assumed that this signal is a realization of a stationary process, and therefore spectral methods can be used.

Firstly, the signal is filtered using pre-emphasis filtering with a $\alpha = 0.97$. The clip is then chunked into smaller frames of 25 ms, with a 10 ms time shift. This means that there will be a 15 ms overlap between all subsequent frames. An example of such a frame can be seen in Figure 10. This signal can be assumed to be a realization of a stationary process, and is therefore eligible for spectral analysis.

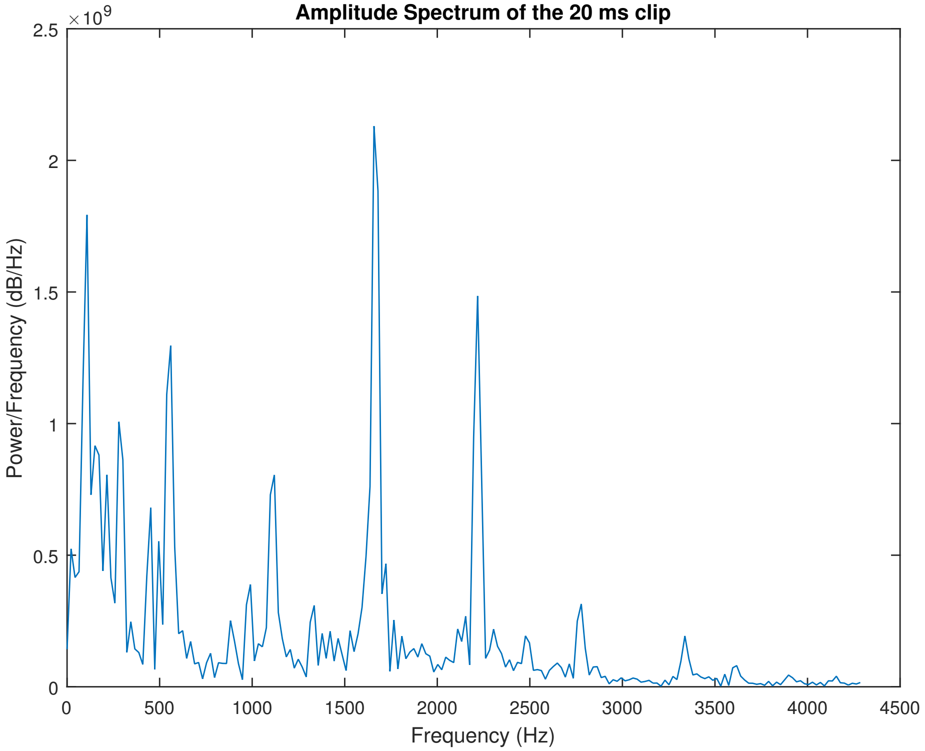


Figure 11: Spectrum of the 25 ms long signal in Figure 10. This particular signal seems to have a lot of power around 150 Hz and 1700 Hz. The smaller peaks are possibly audible as well, but could also be a consequence of timbre.

The spectrum of the frame is computed using the windowed periodogram estimator, with a *Hamming* window. An example of a spectrum can be seen in Figure 11.

The spectrum is then filtered using a triangular filterbank uniformly spaced on the mel-scale. The number of filterbank channels M was set to 15, 20 or 25 depending on parameter configuration, and the frequency range varied between trials. For the example in Figure 12 the frequency range was set to 60-3700 Hz.

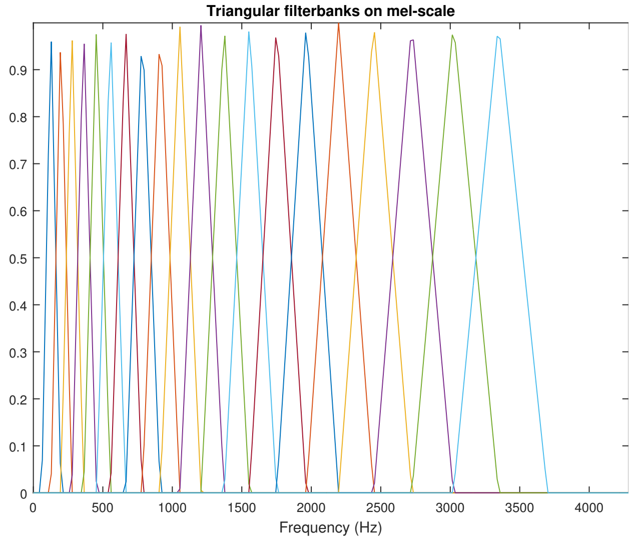


Figure 12: Example of 20 triangular filterbanks spaced uniformly on the mel-scale, here depicted on the frequency scale. The frequency range is 60-3700 Hz.

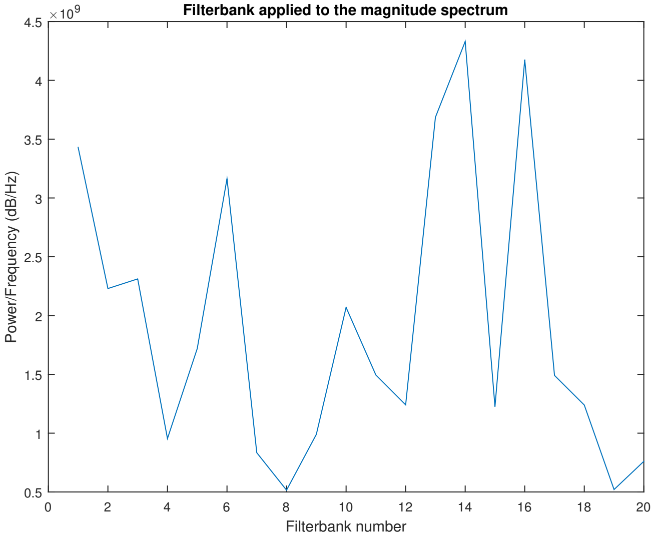


Figure 13: Filtered spectrum of the 25 ms long signal in Figure 10. This vector of values now correspond to the power of the spectrum multiplied by the triangular filterbanks.

The spectrum in Figure 11 is then multiplied with the filterbanks in Figure 12 to get the smoothed spectrum in Figure 13.

The smoothed spectrum is now used to compute the MFCCs using discrete cosine transformation. The resulting coefficient vector of the entire 10-second long clip in Figure 9 can be seen in

Figure 14. The coefficients are then liftered (cepstral filtering) to get the results in Figure 15. Liftering is necessary to force the coefficients to be of similar magnitude, to make classification tasks more stable [22].

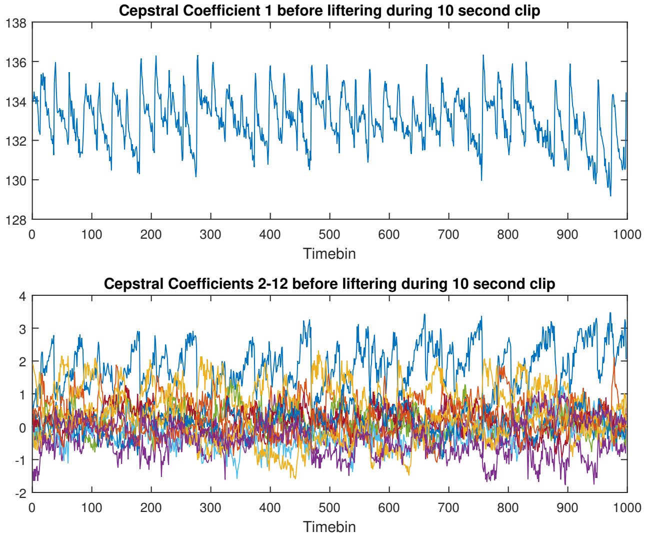


Figure 14: MFCCs of the entire 10-second long clip in figure 9. The number of coefficients were chosen to be 12.

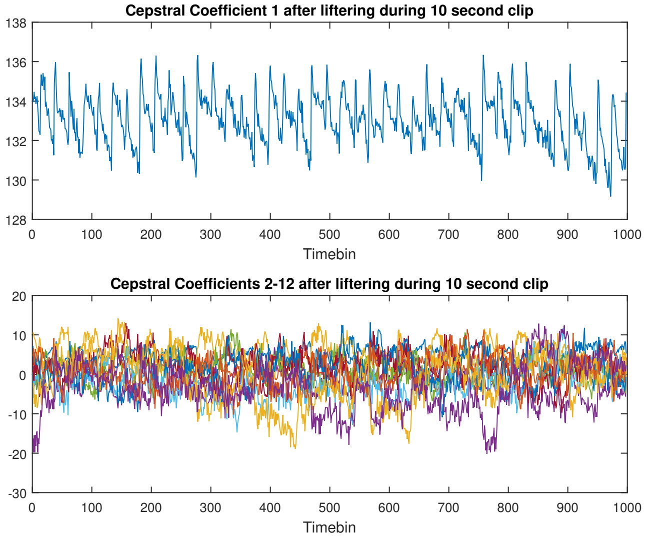


Figure 15: Liftered MFCCs of the entire 10-second clip in figure 9.

After liftering, the delta and acceleration vectors $\mathbf{\Delta}_i$ and $\mathbf{a}_i$ are created from the coefficient vectors $\tilde{\mathbf{c}}_i$ that can be seen in Figure 15. This gives a feature vector $\mathbf{z}(t)$ of 36 values for each time frame of 25 ms. This 36-dimensional feature vector is then used as one observation in the hidden Markov model. It is assumed to be normally distributed. In Figure 16, we can see the histogram of the first 12 coefficients. If they can be assumed to be normally distributed, the delta and acceleration coefficients also go under the same assumptions since they are linear combinations of the first 12.

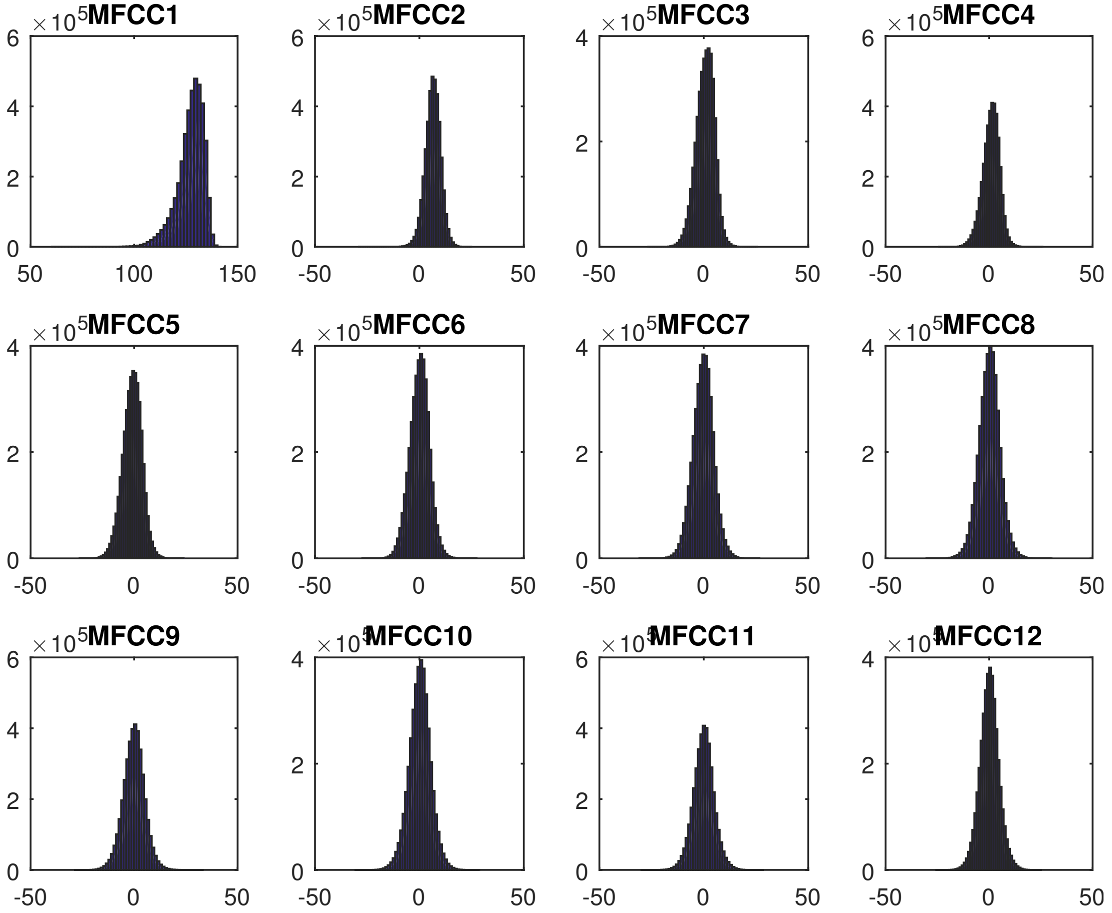


Figure 16: Histogram of the first 12 MFCCs. Other than perhaps the first coefficient, they can be deemed to be normally distributed.

3.3 Classifying transition data

The transition data was classified using several different methods, using a 10-fold cross-validation on the entire data set. Features were collected from the chord transition matrices up to a maximum order of 10. The four external variables *Tempo*, *Complexity*, *NumChords* and *Mode* were also considered. First, an ANOVA was made for each of the 277 possible explanatory variables separately, only keeping the ones with a p-value below the threshold of 0.05. This left a total of 124 considered variables as input to the different models. The transition variables were labeled as [first chord]-[order][second chord], so for example the first order transition C-F would be labeled C-1F.

For the classification procedure, the package `classificationLearner` in Matlab was used. In this package, several different classification methods such as decision trees, support vector machines and ensemble methods were tested. The different methods were tried both using chord transitions and without.

3.4 Classifying audio

For the classification procedure, 70% of the songs were selected for training and 30% were saved for testing. For each parameter setting, a repeated random subsampling cross-validation method was selected, to be able to keep the training/testing data ratio. A total of 10 cross-validation folds were made.

Four models were created using the procedures in Section 2.6. The rest of the data saved for testing were used for prediction of genres, which were then compared to the ground truth label. The first part of the classification was to compare different parameter settings, in order to try to optimise the algorithm. The variables that were considered for optimisation were the number of filter banks M, the frequency range R and the number of states used in the hidden Markov model n.

Once the parameters had been decided upon, a final test was made, using a constrained subsampling cross-validation, making sure that all 10-second clips from each song was put in either training or testing. This was to make sure the classifier measured similarity within genre, and not within songs [10].

When this had been done, an external data set was tested on a model using all data as training data, to see what labels would be put to more famous songs.

3.5 Evaluation

For evaluating the methods, confidence matrices were computed for each of the parameter settings. The parameter setting with the highest total accuracy was declared to be the best one, and was subsequently chosen for the final model.

Two different kinds of measures were used comparing predicted labels to the ground truth labels: **strict comparison** (SC) and **ambiguous comparison** (AC). The strict comparison assigns one predicted label to each song, and compares it to the one ground truth label. The predicted label in this case equals the mode of the labels given to the ten 10-second clips. The ambiguous comparison can accept up to three prediction labels, depending on the result of the prediction for the ten 10-second clips. The ambiguous comparison was included to better suit music pieces that are a mixture of two genres, such as Pop/R&B, and Jazz/Pop. Naturally, this will result in a larger total accuracy, but might also, when compared to the strict comparison, highlight the separation between the genres.

Table 2: Example classification using strict and ambiguous comparison for a song with six clips being classified as Pop, and four as R&B, and one song that is a mixture of Classical, Jazz and Pop. Using AC, we can pick up the mixtures within the same song.

	Audio clip #											
Song	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**SC Prediction**	**AC Prediction**
1	P	P	P	R	R	P	R	R	P	P	Pop	Pop/R&B
2	P	P	P	J	J	C	J	C	C	C	Classical	Classical/Jazz/Pop

4 Results

In this section, the results of the analysis will be presented. The section is divided into two parts: Transition data, which will cover the results of the classification of the theoretical data set from Hooktheory, and Audio data, which covers the results of the classification of the .mp3-files collected from Jamendo, as well as the external test set.

4.1 Transition Data

First, the distribution of chords and chord transitions are shown in Figures 17 through 22. These figures visualize the differences in the raw chord data that was obtained from the Hooktheory data set. In Figures 17 and 18 we can see the distribution of the first order chord transitions from C (major mode) and Am (minor mode) for the different genres. One noticeable difference is the variety of chord transitions from C and Am in Jazz and Classical, and the lack of variety in Pop. The most common transitions are C-G, C-F and C-Am for Pop, Jazz and Classical. For R&B, the transition C-Dm is the second most frequent.

When considering the distribution of chord transitions from Am (minor key), we can see in Figure 18 that the common chord transitions in the different genres are more different. In Pop and R&B, we see that the transitions Am-F and Am-G are quite frequent, while in Classical Am-E and Am-Dm are the most frequent. Jazz has a varied distribution with many different observed transitions.

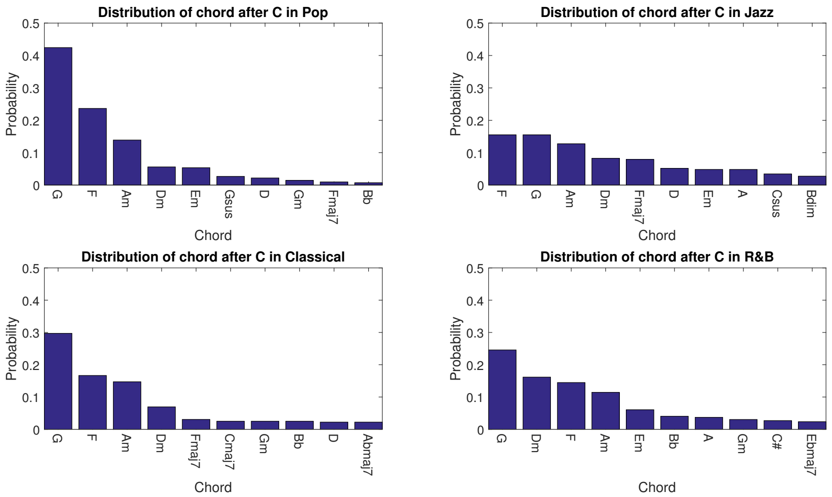


Figure 17: Distribution of the chord transitions from C, assuming the songs starts on the chord of C. A good thing to note here is the difference in spread between the genres. Pop has more probability mass in F and G, while for example Jazz has a lot more variety.

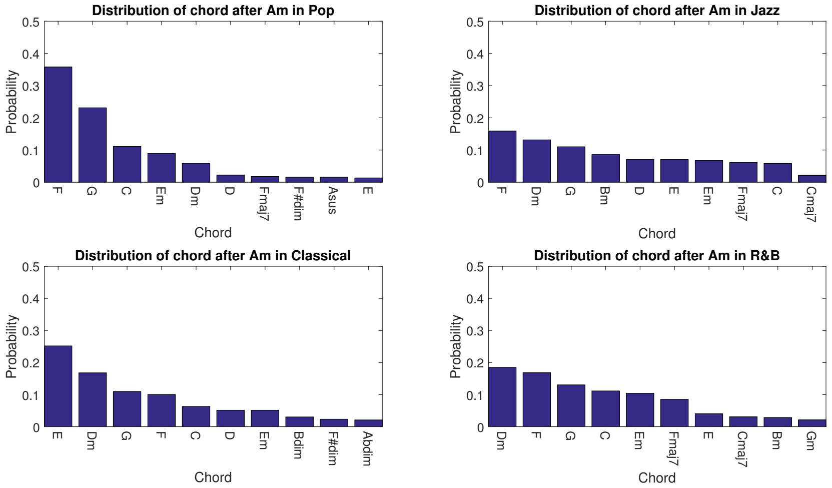


Figure 18: Distribution of the chord transitions from Am, assuming the songs starts on the chord of Am. As for transitions from C, Jazz has more variety than Pop.

The reverse transition matrices of C and Am (Figures 19 and 20 respectively) shows the frequency of chords right before the tonic chord (C for major and Am for minor). In Figure 19 we can see that the typical characteristics of the different genres are visualized. For Pop, the transition F-C is very prominent, while for Classical the transition G-C is almost of equal importance. In this regard, R&B is closer related to Pop, and Jazz is closer to Classical music. Even here, we can note that the variety of transitions onto the tonic chord is larger for Classical and Jazz than for the other two genres. Regarding the minor key, transitions onto the tonic chord are also quite explanatory. For Pop we can see that the transition G-Am is much more frequent than for the other genres. Classical music features the transition E-Am, which Pop and R&B does not to a large extent. Jazz has a very spread transition distribution onto Am, with both E-Am and G-Am being quite common. R&B also has the transition Dm-Am standing out from the rest.

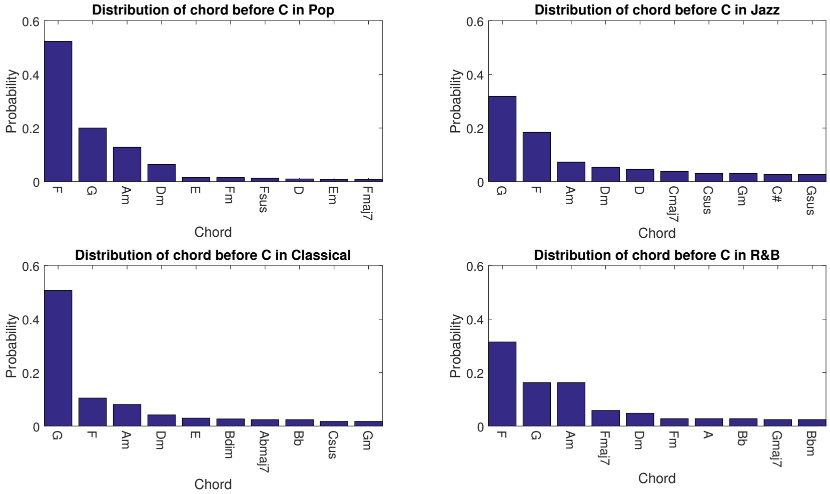


Figure 19: Distribution of the chord transitions onto C, showing the distribution of the chord before C. Here one can see for example that Pop songs tend to come back to C through F, while for Classical it is mainly G-C.

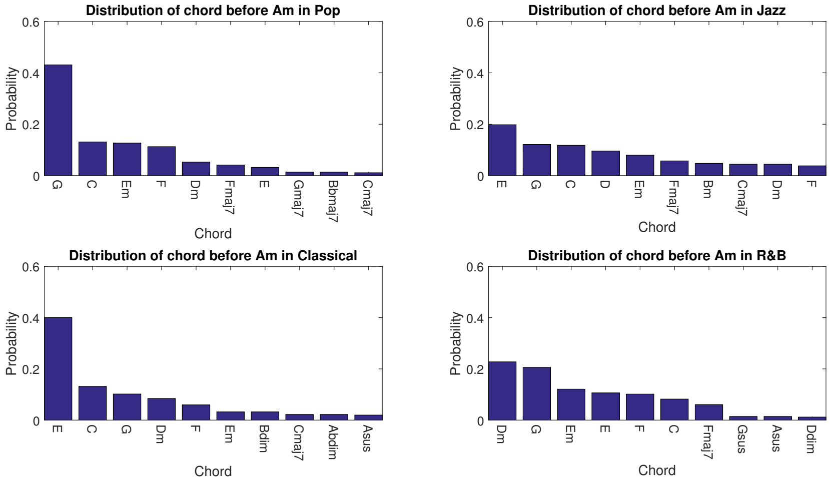


Figure 20: Distribution of the chord transitions onto Am, showing the distribution of the chord before Am. Even here the difference between Pop and Classical becomes clear, when Pop features the transition G-Am, while for classical the most featured one is E-Am. For R&B the transition Dm-Am shows to be quite common.

In Figure 21 we can see the histogram of the chord occurrences for the different genres. It is clear that Pop is not as varied as Jazz, Classical and R&B in terms of chord choices. One can also note the jump in frequency between the first four chords in Pop and the rest. Besides G, Am, F and C, other common chords are Dm, Em and E (mostly in Classical). The usage of major 7-chords in

Jazz can also be seen compared to the other genres.

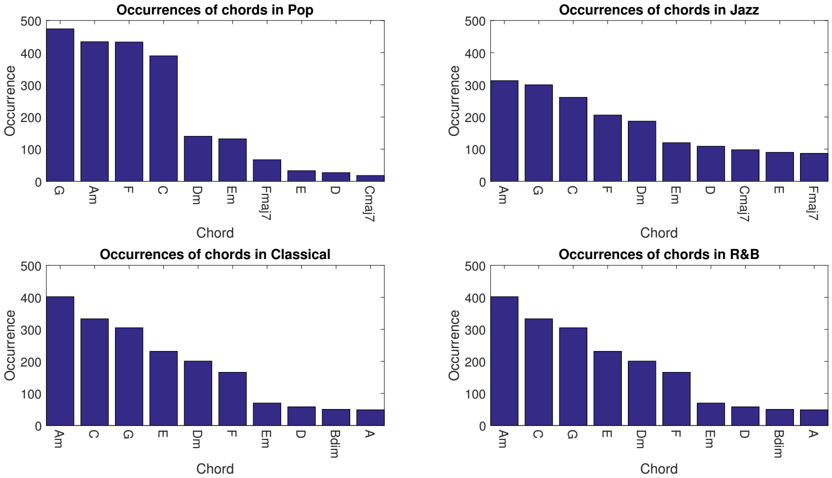
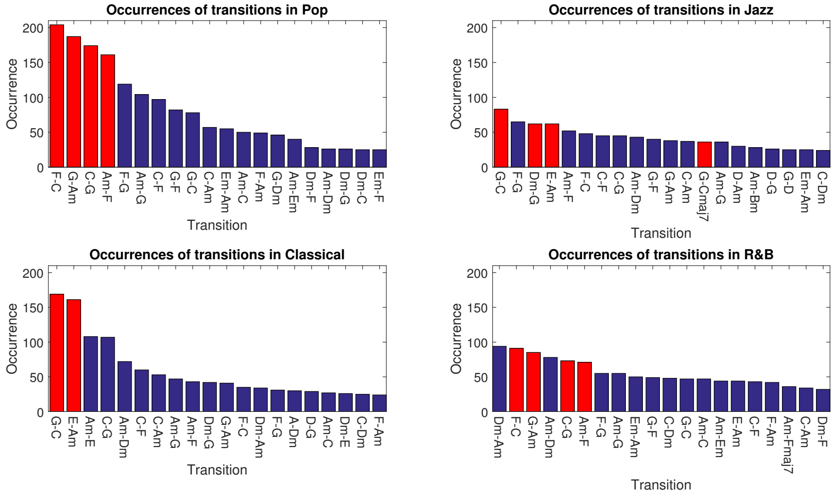


Figure 21: Histogram of the chord occurrences for the four different genres. Pop features the four chords G, Am, F and C in a lot of the songs, and has a quite narrow span of different chords featured. Jazz and Classical has a more varying histogram, the only clear difference being the use of E and F, where E is more frequent in Classical music and F more featured in Jazz. One can also note the use of △-chords for Jazz, which tend to stand out from the rest.

Figure 22 shows the frequency of transitions used for the different genres. The red bars indicate what is expected to be high according to prior knowledge about the genres. For Pop the four-chord transition C-G-Am-F seems to be very frequent in Pop music. It is not until the 11th most frequent transition we see a different chord than one of these four. It seems like these chords dominate the Pop genre to a great extent. For Jazz music we expect to see high frequency in the transitions Dm-G-C/$C^{\triangle}$, as well as E-Am. These can all be found in the histogram of the most frequent transitions. Noteworthy is that the total number of different transitions is so high for Jazz, which can be seen by the size of the bars in Jazz. For Classical we expect to see the circle-of-fifth progression G-C and E-Am, which also is common in the data. For R&B it was expected to look approximately like Pop, but the frequency of the four-chord progression was not as high. Instead, the transition Dm-Am was the most frequent one.



Occurrences of transitions in Pop

Occurrences of transitions in Jazz

Occurrences of transitions in Classical

Occurrences of transitions in R&B

Figure 22: Occurrences of different transitions. The figures shows the 20 most frequent transitions in the respective genre. The red bars indicate transitions that are expected to be high, from theory. We can note that the four-chord progression C-G-Am-F is very frequent in Pop music, and the Dm-G-C/$C^{\triangle}$-progression is quite common in Jazz.

The results of the ANOVAs of the parameters *Complexity*, *NumChords*, and *Tempo* can be seen in Figures 23 through 25 in the appendix. All three parameters are significant in terms of that they can explain some variation between the genres. Figure 26 shows an example of a classification tree obtained by training a classification tree model. The top nodes are complexity, tempo and Am-1E, meaning that these variables explain the most variation in this particular training set. Figure 27 shows the importance of the different variables when making a classification model for the theoretical transition data. It was shown that the three external variables Complexity, NumChords and Tempo explained more of the variability than any of the transition variables. In Table 3 we can see the results of these three observed variables. It shows that Pop is quite separable in the variable *Complexity*, while Classical can be separated by considering *NumChords*.

Table 3: 95% confidence intervals for the observed theoretical parameters in the transition data. Pop is significantly less complex than the other three genres. Classical music is on the contrary very complex and a rich chord structure with many different chords.

Genre	**Complexity**	**NumChords**	**Tempo**
Pop	[2.33, 2.93]	[4.09, 4.65]	[113.35, 124.19]
Jazz	[4.37, 4.91]	[5.49, 6.55]	[115.65, 136.15]
Classical	[4.81, 5.29]	[6.87, 8.07]	[103.64, 124.06]
R&B	[3.43, 4.07]	[4.39, 5.15]	[95.77, 107.53]

The results of the transition classifications with and without chord transitions as features can be seen in Tables 4 and 5 respectively. The best accuracy was reported 54.0% from a Linear SVM using chord transitions. The confusion matrix of the Linear SVM-model can be seen in Table 12.

Table 4: Accuracy of the transition classification methods using chord transitions as part of the feature space. Linear SVM (Support Vector Machines) was the high performer with a 54.0% accuracy over a 10-fold cross-validation. 54% is borderline ok as a four-way classification, but definitely not more.

Method	**Accuracy**
Linear SVM	**54.0**%
Cosine KNN	53.5%
Medium Gaussian SVM	53.0%
Ensemble Bagged Trees	53.0%
Cubic KNN	51.2%
Medium KNN	47.3%
Simple Tree	45.5%

Table 5: Accuracy of the classification methods without chord transitions. Medium KNN seemed to be the best performer judging from the accuracies of the model. It should be noted that 45.8% is a very poor accuracy for a four-way classification.

Method	**Accuracy**
Medium KNN	**45.8**%
Linear SVM	45.3%
Medium Gaussian SVM	45.0%
Cosine KNN	44.8%
Cubic KNN	44.5%
Simple Tree	43.0%
Ensemble Bagged Trees	42.5%

4.2 Audio Data

First, the parameter **R** and the inclusion of the delta- and acceleration coefficients were tested, by looking at what values gave the best results. The results of this parameter testing can be seen in Table 6. It was shown that the frequency range **R** influences the classification accuracy in the lower bound. A decreasing lower bound increases the accuracy of the classification. It was decided to continue with the range of $60 - 3700$ Hz. For the delta and acceleration coefficients, it was shown that including them increases the accuracy severely, which was to be expected [10].

Table 6: Result of the parameter optimisation of **R** and the delta- and acceleration coefficients. It is clear that including the delta and acceleration variables increases the classification accuracy. The frequency range **R** also influences the accuracy. It seems like the decreasing the lower bound from the initial 300 to 60 will increase the overall classification.

n	**R**	**M**	$\Delta(t)$ and $a(t)$	Accuracy (%)
5	60-3700	20	Yes	$[76.6, 77.5]$
6	60-3700	20	Yes	$[76.1, 78.0]$
4	60-3700	20	Yes	$[74.5, 76.9]$
5	130-2000	20	Yes	$[72.6, 74.4]$
4	130-2000	20	Yes	$[72.5, 74.3]$
4	300-3700	20	Yes	$[72.0, 73.6]$
5	300-3700	20	No	$[71.6, 73.0]$
3	130-2000	20	Yes	$[71.1, 72.8]$
3	300-3700	20	Yes	$[69.4, 73.4]$
4	300-3700	20	No	$[69.9, 72.3]$
3	300-3700	20	No	$[69.0, 70.8]$

For the second parameter optimisation, the range was fixed to $60 - 3700$, and the delta and acceleration coefficients were added in all trials. The varying parameters were the number of states n and the number of filterbanks used M. The result of the optimisation is shown in Table 7. The results are presented as a 95% confidence interval for the accuracy. It was shown that the most stable classification with high accuracy was using n=7 and M=20.

Table 7: Result of the second parameter optimisation procedure of the audio classification. This was done using the constrained subsampling cross-validation. Optimising on n and M with R fixed, and using delta and acceleration coefficients, it was shown that the accuracy was maximised using $n = 7$ and $M = 20$.

n	**M**	Accuracy (%)
7	20	[77.0, 80.6]
6	20	[75.1, 81.7]
6	25	[75.9, 80.3]
5	20	[74.8, 80.2]
7	15	[75.3, 79.0]
7	25	[74.3, 79.9]
6	15	[74.6, 78.0]
4	20	[73.0, 77.8]

Using these parameter configurations, the models were tested on an external test set, with modern music. Both evaluation methods were used to collectively judge the performance of the model. In Table 8, we can see the confusion matrix of the **strict comparison** evaluation method. It is clear that the classifier is having problems separating Pop from R&B. The genres Jazz and Classical are performing decently. The overall classification accuracy of the external test set was 0.742.

Table 8: Confusion matrix using a **strict comparison** method of evaluation of the external test set. Most of the error lies in the R&B genre, with a large overlap into the Pop genre. For Jazz and Classical, the classification accuracy is over 0.8. The total accuracy is 0.742.

$$\begin{array}{ll}
 & \begin{array}{cccc} \multicolumn{4}{c}{\text{Predicted Class}} \\ \text{Pop} & \text{Jazz} & \text{Classical} & \text{R\&B} \end{array} \\
\text{True Class} \begin{array}{l}\text{Pop}\\ \text{Jazz}\\ \text{Classical}\\ \text{R\&B}\end{array} & \left[\begin{array}{cccc} \mathbf{31} & 1 & 0 & 8 \\ 3 & \mathbf{50} & 4 & 3 \\ 0 & 1 & \mathbf{25} & 4 \\ 26 & 0 & 0 & \mathbf{38}\end{array}\right] \begin{array}{l} 77.5\% \\ 83.3\% \\ 83.3\% \\ 59.4\%\end{array}
\end{array}$$

Table 9 shows the same classification using the **ambiguous comparison** evaluation method. It is clear that many of the misclassified songs in both Pop and R&B have elements of both genres, and should be labeled R&B/Pop instead of just one of the genres. Another observation is that Jazz does not benefit very much from this evaluation method. The overall classification accuracy using this method is 0.866.

Table 9: Confusion matrix using an **ambiguous comparison** method of evaluation of the external test set. The error still lies in the R&B genre, but it is noticeable that a lot of the misclassified songs from before had elements of both genres. The accuracy here explains how many of the songs had at least three 10-second clips classified as the given genre. The overall classification accuracy is 0.866.

$$
\begin{array}{rcl}
 & \begin{array}{cccc} \multicolumn{4}{c}{\text{Predicted Class}} \\ \text{Pop} & \text{Jazz} & \text{Classical} & \text{R\&B} \end{array} & \\
\text{True Class}\ \begin{array}{l} \text{Pop} \\ \text{Jazz} \\ \text{Classical} \\ \text{R\&B} \end{array} & \left[\begin{array}{cccc} \mathbf{37} & 1 & 0 & 2 \\ 2 & \mathbf{53} & 4 & 1 \\ 0 & 0 & \mathbf{28} & 2 \\ 14 & 0 & 0 & \mathbf{50} \end{array}\right] & \begin{array}{l} 92.5\% \\ 88.3\% \\ 93.3\% \\ 78.1\% \end{array}
\end{array}
$$

Finally, the same test was made but including the spectral features spectral centroid, spectral rolloff and spectral flux. The result in Table 10 shows that the accuracy did not improve, having a total accuracy of 0.856. However, it did increase the accuracy of the Classical genre, which indicates that these three features might be useful for some genres.

Table 10: Confusion matrix using an **ambiguous comparison** method of evaluation of the external test set with addition of the **spectral features** SC, SR and SF. There are some minor differences from Table 9, like improved classification of Classical music, and decreased accuracy of the R&B genre. The total accuracy is a bit lower, at 0.856.

$$
\begin{array}{rcl}
 & \begin{array}{cccc} \multicolumn{4}{c}{\text{Predicted Class}} \\ \text{Pop} & \text{Jazz} & \text{Classical} & \text{R\&B} \end{array} & \\
\text{True Class}\ \begin{array}{l} \text{Pop} \\ \text{Jazz} \\ \text{Classical} \\ \text{R\&B} \end{array} & \left[\begin{array}{cccc} \mathbf{37} & 1 & 0 & 2 \\ 2 & \mathbf{53} & 4 & 1 \\ 0 & 0 & \mathbf{30} & 0 \\ 18 & 0 & 0 & \mathbf{46} \end{array}\right] & \begin{array}{l} 92.5\% \\ 88.3\% \\ 100\% \\ 71.9\% \end{array}
\end{array}
$$

A full list of the classification of the external test set can be seen in the Appendix, Tables 13-17.

5 Discussion

In this section, we will discuss the results, their implications, and suggest possible changes for a future project. There will also be a section for possible errors that have occurred during the process.

5.1 Analysis of results

The results of the classification of the transition data suggests that an inclusion of such variables into a classification of audio files will not improve the accuracy. However, it did show some interesting results in terms of raw chord distribution data. Looking at Figures 17-20 we can see that they agree somewhat with the expected behavior of the different genres as discussed before. Pop is known for being very repetitive with the four-chords transition C-G-Am-F. In the raw data, C-G was the most common transition from C, and all the four-chord-transitions C-G, G-Am, Am-F and F-C were among the four most frequent transitions out of all in the Pop genre. On these results, we can also confirm that Pop music uses the four-chord transition more frequently than the other considered genres. However, the frequency of any transition pair was not large enough to build a classification on. This difference can also be seen in Table 3, where Pop is significantly less complex than the three other genres.

The Jazz genre is very diverse, and that can be seen from the figures as well. The typical Jazz transition Dm-G-C$^{\triangle}$ can be seen as one of the more frequent ones, but the main difference from Pop lies in the number of different possible transitions. It is not easy to say something that defines the Jazz genre, besides the use of major 7-chords ($\triangle$-chords). These are more common in Jazz than the other genres. Jazz is more complex than Pop, and has more different chords according to Table 3.

Classical music is the most separable out of the four genres, with a distinct form of cadence (Figure 19) of G-C, and a frequent use of E-Am. These results are reasonable, considering that classical music sounds less like the other genres. What is interesting about the misclassified classical songs is that the major portion of them is classified as R&B, which could be because of larger instrumental melodic sections within the R&B genre, or outliers in the training data. The two genres are typically not close to each other in terms of similarity of songs, which indicates that there are possible improvements that can be done to the model.

Lastly, the R&B genre was the hardest one to classify, possibly due to the overlap with Pop. Using strict comparison only 59.4% of the songs were correctly classified, which is a poor result. However, one has to bring up the argument about who puts the labels on the songs, and if Pop and R&B really are two distinct genres. The overlap in classification suggests that the two genres are very similar, and that there are a lot of songs that can be classified as Pop/R&B. The raw chord transition frequency data suggests that the four-chord transition that usually is used to identify Pop is also frequent in R&B. Additionally, R&B features more minor-chord transitions such as Dm-Am and the inverse Am-Dm. This might be used in the future as an indicator for R&B.

The optimal parameter setting seemed to depend greatly on the frequency range of the filterbank. When the lower bound was lowered from 300 to 60, the accuracy increased dramatically. This could potentially be due to importance of bass and drums in some genres.

Looking at the results from the transition data classification, we can see that the overall accuracy is poor, maximized at 0.540 for a Linear SVM using chord transitions. This is not enough to be deemed as a successful classifier for musical genres, and could have many reasons, some of which are explained in the next section.

The final classification accuracy of the audio files was 0.742 for strict comparison and 0.866 for ambiguous comparison. These results indicate that the algorithm is successfully extracting useful spectral information and separating the genres. Even in a setting like this, with genres as close to each other as Pop and R&B it is successful. The idea of combining chord transitions with this classification to improve the accuracy does not seem reasonable with the current state of transition classification.

It was also tested to try to improve the audio classification by adding the spectral features spectral centroid, spectral rolloff and spectral flux. The results in Table 10 shows that such an improvement was not found for the overall accuracy, getting 0.856 compared to the 0.866 of Table 9. However, the misclassified songs from Classical to R&B were removed, which indicates that it might be useful for classification problems with genres that are not very similar, as a coarse first-step classification. It should also be added that this step was made after the parameter optimisation, and might have changed if it was included in the optimisation.

5.2 Possible Errors

In this section, we will discuss what choices of methods might have influenced the outcome of the project.

When looking at the results of the classification of the transition data, there are plenty of reasons why the accuracy is so poor. The accuracy measure is just one way to assess how good a model is, but it shows how the fit is from the predicted labels to the ground truth. As stated before, one problem about dealing with labels of songs is that they often do not perfectly belong to one single genre. This makes the accuracy measure very unstable, and hence not very reliable. It should also be said, that the labels of the transition data is in many cases set by users of the website Hooktheory. This makes the ground truth labels very subjective, depending on what types of songs conforms to a certain genre, for a single individual. While looking at the problem this way, one can also argue that this is exactly the types of differences a classification algorithm should be able to pick up. We would like an algorithm like this to match the *public opinion* rather than the opinion of an expert. One better way of collecting the data would be by choosing the genre by popular vote, instead of a single person's vote.

The same issue arose when considering the classification of the audio data. The genres were mostly selected by the users of the website Jamendo, which in turn lead to possible unreliable data. In some rare cases, the genre labels were changed due to inconsistencies within the same genre. An example of this was when two songs from the same artist and album were put in different genres (Pop & R&B, respectively). This shows how difficult the problem of ground truth labels are. Ideally, we would like to capture the similarities of the genre, by collecting songs that are similar in the same classification class. If one in this case separates the two songs into different genres, similarities between the songs will make the classification worse, since it is the same artist, on the same album. If one instead puts the songs in the same genre, the data might be skewed towards particular artists instead of explaining the whole genre. If one instead removes any duplicate artists from different genres, one ends up in a situation where 100 different artists are needed for every genre, giving a very hard time finding suitable data. In this project, it was chosen to accept the first two flaws in favor of not having to spend too much time finding data. For future research in the area, one can make restrictions on having multiple songs from the same artist, and probably improve the data set.

The audio classification accuracy was a lot better than the transition-based classification, and so the error could not possibly lie completely in the form of ground truth labels. For the transition-based data, it is clear that the choice of variables are not enough to classify the data into genres. Based on Figure 27, we can see that the variables *Tempo*, *Complexity* and *NumChords* are the most important ones. The face that tempo is the variable explaining the most of the variance shows that the current choice of feature space is not optimal. The idea of this analysis was to see if one could use the music theory aspects of the genres, like special chord progressions to enhance the classification of the audio data in the future. Out of the transition features, the variable *G-1C* and *Am-1E* were the most useful, and it makes sense, judging from the histograms in Figure 22. Both of the transitions are very frequent for classical music, but not very much for the other genres. The idea was that occurrences of such transitions within a song would point more strongly towards a certain genre. A reason to why this did not work out was that just pure transitions without the rhythmic component is too simplifying. The transition G-1C only means that C is the next chord after G, but it does not tell us how long the G-chord is. It is clear that rhythmic components are very important for understanding the genre, and that ignoring them is a oversimplification of chord transitions.

Another possible error of dealing with chord transitions the way it was done in this project is that not all chord progressions are equally long. This was fixed by looping the chord progression until it reached 24 total chords. While this makes every song equally important in the sense that each song contributes to the training model with equal weight, it also skews the data towards smaller chord progressions. A chord progression which is only four chords long will count six times, while a chord progression which is twelve chords long will count only two. At the time, this was the clear way of solving the issue of unequal lengths, but there might be a more elegant way, that deals with the skewness of data as well.

The audio classification data was reasonable in terms of feature space, and the choices of variables was easily evaluated. However, there were some questions that arose when initially reading the data. All songs came in the form of .mp3-files, which is a common way of reducing the size of music. Compared to a CD, which is a lossless format of songs, a lot of the data is lost when converting to .mp3. For this project, it was enough, but it should be noted that using a lossless format, such as FLAC, would contain more information, a might have yielded a better result. Another simplification that was made was converting the songs to mono, by simple averaging, losing the stereo panning effect. There might have been information there that could have given a better result in classification accuracy. However, due to algorithm run time, this choice was preferred over keeping the stereo track.

It was chosen to divide each songs into ten 10-second long clips of music, centered on the middle of the song. This was to exclude parts like intros and outros, which often are slower and more mellow. These parts are less likely to be defining a certain genre. A possible problem of having 10-second long clips is that there might be clips where the mood of the song switches in the middle. This could for example be in transitions between choruses and verses. In these clips, it is expected that the training of the HMM is less accurate, since the distribution might change entirely. For windowing the spectrum, the Hamming window was chosen since it had proven to be useful in previous research.

In Figure 16 we can see that the first MFCC might not be completely normally distributed. This could have influenced the result. Different countermeasures were considered, ranging from adding a copula to deal with the first coefficient, or changing the distribution completely. Due to time constraints, this was not possible, but might have improved the results slightly.

5.3 Future Research

For future research, it is suggested to focus the classification tasks on close genres, to make the analysis more relevant. Personally, I don't think the research on more theoretical data is finished, but a different set of variables might prove to improve the accuracy. If this is the case, it could be combined with chord estimating techniques to enhance the audio classification from timbre features to pitch features as well.

In order to improve the feature space, there is plenty of alterations that can be made. One could question the Gaussianity of the MFCCs, especially the first coefficient, and maybe add some sort of copula to deal with the possible non-Gaussianity. One could also extend the number of genres to see how the algorithm would fare in such a test.
The ambiguous comparison evaluation method has been devised to deal with more complex classification methods, such as when the genres are close in similarity.

The theoretical data set could additionally be used to make an application that simulates chord transitions from a given genre, to help out in composing or songwriting. This was experimented with during the project, but with limited success.

6 Conclusion

In this project, several different types of analyses of music genre data has been carried out. A classification of theoretical transition data using different classification methods has shown to be less successful, with a final accuracy of 0.540. Another classification was made using HMMs of audio data, into the four genres Pop, Jazz, Classical and R&B. This gave a final accuracy of 0.742 using a strict comparison to ground truth. Another evaluation measure was devised, better suited to deal with music genre data, which gave a accuracy of 0.866. This could be deemed a successful classifier, which can be implemented into music streaming softwares. The optimal choice of parameters were tested, as well as a suitable feature space for the problem. Different ways of improving the classifier are discussed, and suggested.

7 References

[1] Alexandridis, A., Chondrodima, E., Paivana, G., Stogiannos, M., Zois, E., & Sarimveis, H. (2014, September). Music genre classification using radial basis function networks and particle swarm optimization. In Computer Science and Electronic Engineering Conference (CEEC), 2014 6th (pp. 35-40). IEEE. doi:10.1109/CEEC.2014.6958551

[2] Chai, W. & Vercoe, B. (2001). Folk music classification using hidden Markov models. Proc. of International Conference on Artificial Intelligence, June 2001.

[3] Cheshomi, S., Saeed, R. Q., & Akbarzadeh-T, M. R. (2010, August). HMM training by a hybrid of chaos optimization and Baum-Welch algorithms for discrete speech recognition. In Digital Content, Multimedia Technology and its Applications (IDC), 2010 6th International Conference on (pp. 337-341). IEEE. Available at:
http://ieeexplore.ieee.org/abstract/document/5568627/.

[4] Durey, A., & Clements, M. A. (2001, October). Melody Spotting Using Hidden Markov Models. In ISMIR. Available at:
http://ai2-s2-pdfs.s3.amazonaws.com/cc42/744e0bf5cf4452e31b34f471a5895890ed96.pdf.

[5] Fu, Z., Lu, G., Ting, K. M., & Zhang, D. (2011). A survey of audio-based music classification and annotation. IEEE transactions on multimedia, 13(2), 303-319.
doi:10.1109/TMM.2010.2098858.

[6] Gold, B., Morgan, N. & Ellis, D. (2000) Statistical Model Training, in Speech and Audio Signal Processing: Processing and Perception of Speech and Music, Second Edition, John Wiley & Sons, Inc., Hoboken, NJ, USA. doi: 10.1002/9781118142882.ch25.

[7] Haggblade, M., Hong, Y., & Kao, K. (2011). Music genre classification. Department of Computer Science, Stanford University. Available at:
http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf.

[8] Huang, Y. F., Lin, S. M., Wu, H. Y., & Li, Y. S. (2014). Music genre classification based on local feature selection using a self-adaptive harmony search algorithm. Data & Knowledge Engineering, 92, 60-76. Available at:
https://doi.org/10.1016/j.datak.2014.07.005

[9] Jackson, P. (2011). Hidden Markov Models [pdf slides]. Retrieved from:
http://personal.ee.surrey.ac.uk/Personal/P.Jackson/ISSPR/hmm_isspr11_hw.pdf.

[10] Karpov, I., & Subramanian, D. (2002). Hidden Markov classification for musical genres. Course Project. Available at:
http://www.cs.utexas.edu/users/ikarpov/Rice/comp540/finalreport.pdf.

[11] Lee, J. H., & Downie, J. S. (2004, October). Survey Of Music Information Needs, Uses, And Seeking Behaviours: Preliminary Findings. In ISMIR (Vol. 2004, p. 5th). Available at:
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.536.7593&rep=rep1&type=pdf

[12] Logan, B. (2000, October). Mel Frequency Cepstral Coefficients for Music Modeling. In ISMIR. Available at:
https://pdfs.semanticscholar.org/afe2/38f9ac0678e840ff1521f49c6fe749856109.pdf.

[13] Nanni, L., Costa, Y. M., Lumini, A., Kim, M. Y., & Baek, S. R. (2016). Combining visual and acoustic features for music genre classification. Expert Systems with Applications, 45, 108-117. Available at:
https://doi.org/10.1016/j.eswa.2015.09.018

[14] Panagakis, Y., Kotropoulos, C. L., & Arce, G. R. (2014). Music genre classification via joint sparse low-rank representation of audio features. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), 22(12), 1905-1917. doi:10.1109/TASLP.2014.2355774

[15] Poria, S., Gelbukh, A., Hussain, A., Bandyopadhyay, S., & Howard, N. (2013, June). Music genre classification: A semi-supervised approach. In Mexican Conference on Pattern Recognition (pp. 254-263). Springer Berlin Heidelberg. doi:10.1007/978-3-642-38989-4_26

[16] Qi, Y., Paisley, J. W., & Carin, L. (2007). Music analysis using hidden Markov mixture models. IEEE Transactions on Signal Processing, 55(11), 5209-5224. doi:10.1109/TSP.2007.898782.

[17] Stevens, S.S. (1937). A Scale for the Measurement of the Psychological Magnitude Pitch. The Journal of the Acoustical Society of America, 8, 185-190. doi:10.1121/1.1915893.

[18] Stewart, J. (2008). Fourier Series. Calculus Early Transcendentals, (6th ed.). Belmont: Thomson Higher Education. Available at:
http://www.stewartcalculus.com/data/CALCULUS%20Early%20Transcendentals/upfiles/FourierSeries5ET.pdf.

[19] Talupur, M., Nath, S. & Yan, H. (2001), Classification of music genre,
Available at http://www.cs.cmu.edu/ yh/files/GCfA.pdf.

[20] Tzanetakis, G., Jones, R., & McNally, K. (2007, September). Stereo Panning Features for Classifying Recording Production Style. In ISMIR (pp. 441-444). Available at:
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.205.8283&rep=rep1&type=pdf.

[21] Vergin, R., O'shaughnessy, D., & Farhat, A. (1999). Generalized mel frequency cepstral coefficients for large-vocabulary speaker-independent continuous-speech recognition. IEEE Transactions on Speech and Audio Processing, 7(5), 525-532. doi:10.1109/89.784104.

[22] Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P., 2006. The HTK Book (for HTK Version 3.4), 64-66. Engineering Department, Cambridge University. Available at:
http://speech.ee.ntu.edu.tw/homework/DSP_HW2-1/htkbook.pdf.

[23] Zucchini, W. & MacDonald, I.L. (2009) Hidden Markov Models for Time Series: An Introduction Using R. Boca Raton, Florida: Chapman & Hall/CRC.

A Appendix

A.1 Explanation of variables and parameters

- $c_i(t)$: Mel-Frequency Cepstral Coefficient i in time frame (t).
- $\Delta_i(t)$: First-order change in $c_i(t)$.
- $a_i(t)$: Second-order change in $c_i(t)$.
- N: Number of coefficients used as features. Set to 12.
- M: Number of filterbanks used. Set to 20.
- R: Frequency range considered in the filterbank. Set to $60 - 3700$ Hz.
- L: Cepstral lifter sine parameter. Set to 22.
- α: Pre-emphasis filter coefficient. Set to 0.97.
- n: Number of states used in the hidden Markov model. Varies between 4-7.
- **Song Part**: What part of the song is considered, useful for backtracking purposes. Could be Intro, Verse, Pre-Chorus, Chorus, Instrumental or variations of the former.
- **Chord progression**: Sequence of 24 consecutive chords, ignoring parts where the same chord is played twice in a row. Total of 60 different chords were considered, 5 per key. The chords were major (C), minor (Cm), diminished (C°), major 7 ($C^{\triangle}$) and suspended (C^{sus}). All chords were transcribed to one of these chords. More complex chords were simplified. All chords were transposed into the key of C for songs in major, and Am for songs in minor. If there were not 24 different chords in the part that was considered, the entire chord progression was looped until 24 chords was reached.
- **Melody**: The vocals were translated into a number corresponding to the position of the tone in the chromatic scale (ignoring octaves), giving a C-tone number 1 and $C^{\#}$ number 2 and so on. This was done for the song part that was considered. For the songs without vocals (such as most classical music), the melody was considered to be the most prominent melody played during the passage.
- **Complexity**: A measure (1-6) of the complexity of the chords used in the part. The measure was defined by Hooktheory, and used accordingly. The explanations of the values can be found in table 11.
- **Genres**: Pop, Jazz, Classical and R&B.
- **Tempo**: Number of beats per minute.
- **Mode**: What scale was used in that part. The different modes considered were: Major, Minor, Dorian, Phrygian, Lydian and Mixolydian.
- **NumChords**: Total number of different chords used in the 24-chord sequence. This was to add the information about looping sequences, and repeating passages.

Table 11: Explanation of the complexity variable used in the analysis. The higher the number, the more complex the chords in the song are.

Value	Description
1	Songs that have only C, Am, F and G
2	Songs that have only the ones above, Dm and Em
3	Songs that have either a 7-chord (i.e., Am^7), a B°-chord, or an inversion (i.e., G^6 or Em^6_4).
4	Songs that have an applied chord (i.e., G/C) or an inverted 7th chord (i.e., Am^4_2).
5	Songs that have a borrowed chord (i.e., Fm) or that have an applied chord and an inverted 7th chord.
6	Songs with borrowed chords that also have an applied chord or an inverted 7th chord.

A.2 Figures and Tables

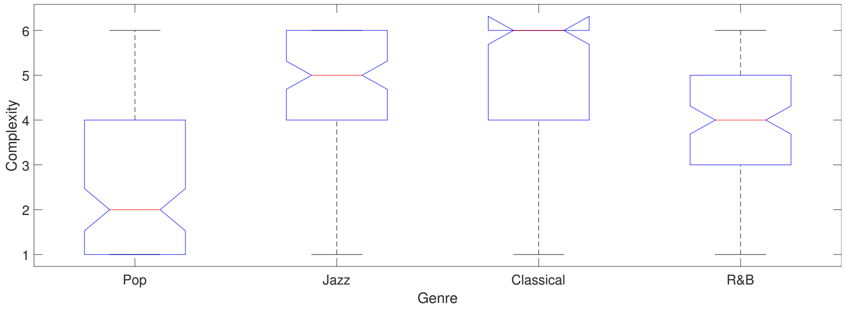


Figure 23: Anova of Genre vs. Complexity. $p < 0.01$, and hence the parameter complexity can be used in modelling.

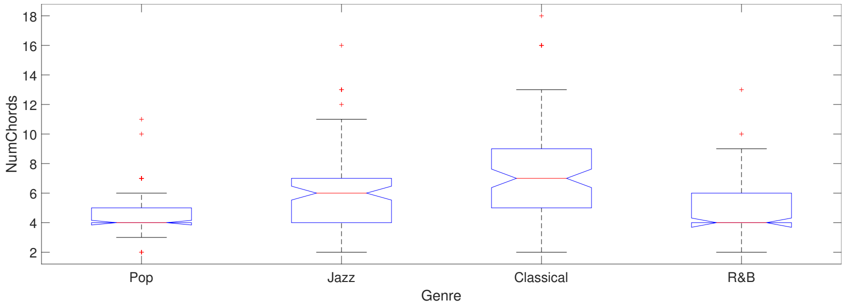


Figure 24: Anova of Genre vs. NumChords. $p < 0.01$, and NumChords can be used in modelling.

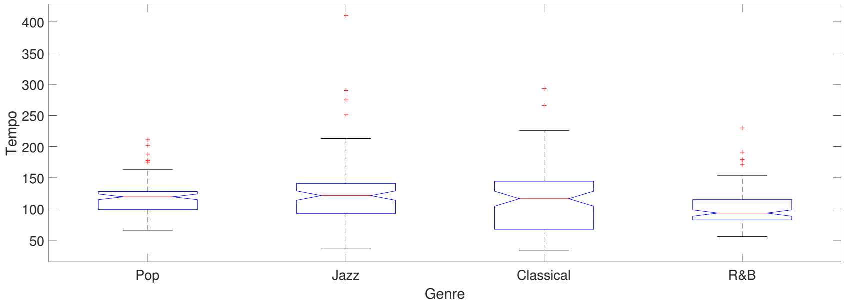


Figure 25: Anova of Genre vs. Tempo. $p < 0.01$, and Tempo is included in the model.

Table 12: Confusion matrix of the classification using a linear SVM with chord transitions. It is clear that Pop and R&B are quite hard to distinguish between using only theoretical data. The result is based on a 10-fold cross-validation of 100 data points per genre.

		Predicted Class				
		Pop	Jazz	Classical	R&B	
True Class	Pop	**55**	14	2	29	55%
	Jazz	12	**50**	14	24	50%
	Classical	9	21	**62**	8	62%
	R&B	25	18	8	**49**	49%

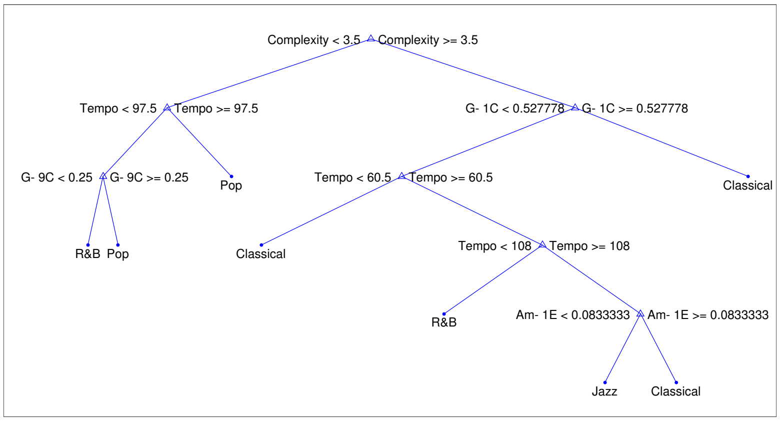


Figure 26: Example of a pruned classification tree suggested by the training of a classification tree model. The variables complexity, tempo and G-1C are among the top nodes, meaning that they explain most of the variability in the data set.

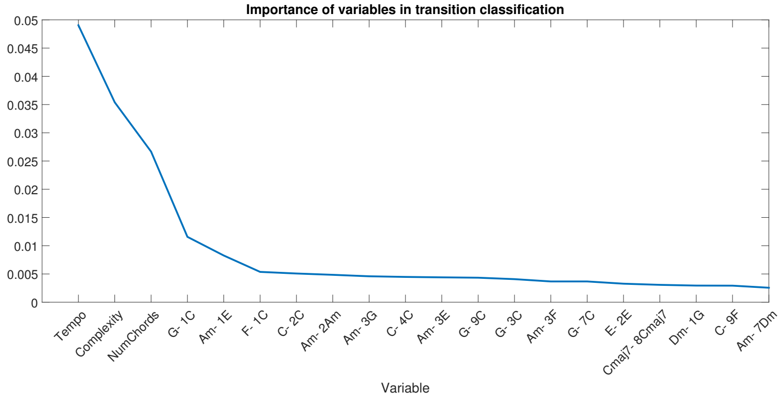


Figure 27: Importance of the variables used in transition data modelling. The three external variables Tempo, Complexity and NumChords are the most important one for modelling purposes, with the transition variables G-1C and Am-1E the most important such variables.

A.3 Available code

The code used for testing the classifier on an external test set is available here.

A.4 Full classification list of external test set

The following pages shows the full external test set, with corresponding classification result. The label is the ground truth, and the four rightmost columns shows the fraction of the song that were classified in the genres Pop, Jazz, Classical, and R&B respectively. Numbers in bold text are the chosen label for the song using ambiguous evaluation method.

Table 13: Results of the classification of the external test set (page 1).

			Classified as:			
Artist	**Song**	**Label**	**P**	**J**	**C**	**R**
Beethoven	Symphony No. 5 - 1st movement	C	0.1	0.1	**0.7**	0.1
Beethoven	Piano Sonata No. 14 - 1st Movement	C	0	0	**1**	0
Beethoven	Symphony No. 6 - 5th movement	C	0.2	0	**0.8**	0
Beethoven	Piano Sonata No. 9 - 2nd Movement	C	0	0	**1**	0
Beethoven	Coriolan Overture	C	0	0.1	**0.9**	0
Beethoven	German Dance - Rondo	C	0	0	**1**	0
Beethoven	Symphony No. 7 - 2nd Movement	C	0	0	**1**	0
Beethoven	Horn Sonata	C	0	0	**1**	0
Beethoven	Für Elise	C	0	0	**1**	0
Mozart	The Marriage of Figaro	C	0.1	0.1	**0.7**	0.1
Mozart	Piano Sonatta No. 15	C	0	0	**1**	0
Mozart	Serenata Notturna	C	0	0	**0.8**	0.2
Mozart	Piano Sonata No. 5	C	0	0	**1**	0
Mozart	Piano Concerto No. 26	C	0.1	0	**0.9**	0
Mozart	Clarinet Quintet	C	0	0	**1**	0
Mozart	Horn Concerto No. 4	C	0	0	**1**	0
Mozart	Piano Concerto No. 24	C	0	0	**1**	0
Mozart	Senerade No. 13 'A Little Night Music'	C	0	0	**0.8**	0.2
Strauss	Emperor Waltz	C	0	0.1	**0.9**	0
Strauss	Vienna Bonbons Waltz	C	0	0.1	**0.4**	**0.5**
Strauss	Treasure Waltz	C	0	0.1	**0.9**	0
Strauss	Wine, Women And Song	C	0	0.1	**0.7**	0.2
Strauss	Tales From The Vienna Woods	C	0	**0.7**	**0.3**	0
Strauss	On The Beatiful Blue Danube	C	0.1	**0.4**	**0.5**	0
Vivaldi	Spring From The Four Seasons	C	0	0	0	**1**
Vivaldi	Concerto In G-Dur	C	0	0	**0.4**	**0.6**
Vivaldi	Concerto In G-Moll	C	0	0	**1**	0
Vivaldi	Autumn From The Four Seasons	C	0	0.2	**0.8**	0
Vivaldi	Concerto In C Dur	C	0	0	**0.5**	**0.5**
Vivaldi	Winter From The Four Seasons	C	0	0.1	0.1	**0.8**

Table 14: Results of the classification of the external test set (page 2).

			Classified as:			
Artist	**Song**	**Label**	**P**	**J**	**C**	**R**
Anita O'day	Opus 1	J	0.1	**0.8**	0	0.1
Artie Shaw	Begin the beguine	J	0	**1**	0	0
Astrud Gilberto	Far away	J	0	**0.9**	0.1	0
Benny Goodman	Jersey bounce	J	0.3	**0.7**	0	0
Billie Holiday	I'll be seeing you	J	0	0	**1**	0
Billie Holiday	These foolish things	J	0	**0.5**	**0.5**	0
Billie Holiday	Blue moon	J	0	**1**	0	0
Billie Holiday	On the sunny side of the street	J	0	**0.7**	**0.3**	0
Billie Holiday	You go to my head	J	0	**0.9**	0.1	0
Carol Sloane	As time goes by	J	0	**1**	0	0
Carol Sloane	Misty	J	0	0	**1**	0
Carol Sloane	My foolish heart	J	0	**1**	0	0
Charlie Parker	A night in Tunisia	J	0	**1**	0	0
Charles Mingus	Stormy weather	J	0	**0.5**	**0.3**	0.2
Chick Corea	Moments notice	J	0	**1**	0	0
Count Basie	One o'clock jump	J	0	**1**	0	0
Dave Brubeck	Take five	J	0	**0.9**	0.1	0
Dexter Gordon	I should care	J	0	**1**	0	0
Dexter Gordon's All Stars	Blow Mr. Dexter	J	0	**1**	0	0
Dinah Washington	Blow top blues	J	0	**0.9**	0	0.1
Dizzy Gillespie	Blue 'n' boogie	J	0	**0.9**	0	0.1
Dizzy Gillespie	Love me or leave me	J	0	0	**1**	0
Duke Ellington	Take the 'A'-train	J	0	**0.4**	0	**0.6**
Duke Ellington	Perdido	J	0.2	**0.8**	0	0
Eddie Harris	Laura	J	0	**0.8**	0	0.2
Ella Fitzgerald	Angel eyes	J	0	**0.5**	**0.5**	0
Ella Fitzgerald	Oh lady be good	J	0	**1**	0	0
Ella Fitzgerald	How high the moon	J	0	**1**	0	0
Ella Fitzgerald	April in Paris	J	0	**0.8**	0.2	0
Ella Fitzgerald	Tenderly	J	0	**0.9**	0	0.1
Fats Waller	It's a sin to tell a lie	J	0	**0.7**	0	**0.3**
Frank Sinatra	Sweet Lorraine	J	0	**0.6**	**0.4**	0
Frank Sinatra	S'posin	J	0	**0.7**	**0.3**	0
Frankie Laine	If you were mine	J	0	**0.8**	0	0.2
George Benson	The masquerade is over	J	0	**1**	0	0
Glenn Miller	String of pearls	J	**0.7**	0	0.1	0.2
Glenn Miller	Tuxedo Junction	J	0	**0.7**	0.2	0.1
Jack Jones	You've changed round midnight	J	0	**0.9**	0.1	0
Jimmy Rushing	Exactly like you	J	0	**0.6**	0	**0.4**
Lena Horne	When I fall in love	J	0	**0.9**	0	0.1
Louis Armstrong	Ain't misbehavin'	J	**0.3**	**0.6**	0	0.1
Louis Armstrong	Do you know what it means to miss New Orleans	J	0	**0.9**	0.1	0
Martha Tilton	And the angels sing	J	**0.4**	**0.6**	0	0
Mel Torne	Night and day	J	**0.3**	**0.3**	0.2	0.2
Miles Davis	Out of nowhere	J	0	**0.7**	**0.3**	0

Table 15: Results of the classification of the external test set (page 3).

			Classified as:			
Artist	**Song**	**Label**	**P**	**J**	**C**	**R**
Miles Davis	My old flame	J	0	**0.8**	0.2	0
Nat King Cole	It's only a paper moon	J	0	**0.8**	0.2	0
Nat King Cole	I'm thru with love	J	0	**1**	0	0
Nat King Cole	Embraceable you	J	0	**0.9**	0	0.1
Nat King Cole	(Get your kicks on) Route 66	J	0	**0.5**	**0.5**	0
Ray Charles	I wonder who's kissing her now	J	0	0.1	**0.9**	0
Sarah Vaughan	Summertime	J	0.2	**0.5**	0	**0.3**
Sarah Vaughan	Lover man	J	0	**0.3**	0.1	**0.6**
Sarah Vaughan	What a difference a day makes	J	0	**0.7**	0.1	0.2
Sonny Rollins	Every time we say goodbye	J	0	**1**	0	0
Stan Getz	Heartplace	J	0	**1**	0	0
Stan Kenton	The peanut vendor	J	**0.8**	0.2	0	0
Thelonious Monk	Round midnight	J	0	**1**	0	0
Tony Bennett	I've grown accustomed to her face	J	**0.3**	0.1	0	**0.6**
Wynton Marsalis	A wheel within a wheel	J	0	**1**	0	0
36	Redlight	P	**0.6**	0	0	**0.4**
Aloe Blacc	The man	P	**0.7**	0	0	**0.3**
American Authors	Best day of my life	P	**0.9**	0	0	0.1
Andreas Bourani	Auf uns	P	**1**	0	0	0
Aneta Sablik	The one	P	**0.9**	0.1	0	0
Aram Mp3	Not alone	P	**0.7**	**0.3**	0	0
Avicii	Lay me down	P	**0.7**	0	0	**0.3**
Bakermat	One day	P	0.2	**0.4**	0	**0.4**
Bellini	Samba do Brasil	P	**1**	0	0	0
Bruno Mars	Young girls	P	**1**	0	0	0
The Chainsmokers	#selfie	P	**0.5**	0	0	**0.5**
Cris Cab	Loves me not	P	**0.6**	0	0	**0.4**
Dizzee Rascal	We don't play around	P	**0.8**	0	0	0.2
Duke Dumont	I Got U (Radio Edit)	P	**0.8**	0	0	**0.2**
Indila	Dernière Danse (Radio edit)	P	**0.7**	0	0	**0.3**
Jamie Starr	Every minute mi amore	P	**1**	0	0	0
Jamie Starr	Poverty & Beaches	P	**0.3**	0.2	0	**0.5**
Jan Delay	St. Pauli	P	**0.8**	0.2	0	0
Klangkarussell	Netzwerk (Falls like rain)	P	**0.8**	0.1	0	0.1
Kollegah	Du bist boss	P	**0.3**	0	0	**0.7**
Lady Gaga	G.U.Y	P	**1**	0	0	0
Mando Diao & Jan Hammer	Black saturday	P	**1**	0	0	0
Mateo	Isso	P	**0.6**	0	0	**0.4**
Mia.	Queen	P	**0.7**	0	0	**0.3**
Mia Martina	Danse	P	**0.8**	0	0	0.2
Michael Jackson	Love never felt so good	P	**0.8**	0	0	0.2
Mr. Probz	Waves	P	0.1	0.1	0	**0.8**
Nico & Vinz	Am I wrong	P	**1**	0	0	0
Parov Stelar	All night	P	**0.4**	0.1	0	**0.5**

Table 16: Results of the classification of the external test set (page 4).

			Classified as:			
Artist	**Song**	**Label**	**P**	**J**	**C**	**R**
Pete Kennedy	Alive	P	**0.5**	0	0	**0.5**
Pharell Williams	Happy	P	**0.3**	0	0	**0.7**
Pitt Leffer	No lies	P	**1**	0	0	0
Rea Garvey	Can't say no	P	**0.9**	0.1	0	0
Route 94	My love	P	**0.3**	0	0	**0.7**
Tiësto feat. Matthew Koma	Wasted	P	0.2	0	0	**0.8**
Vance Joy	Riptide	P	**0.7**	0	0	**0.3**
Wankelmut	Wasted so much time	P	**0.4**	0	0.1	**0.5**
Zedd	Stay the night	P	**1**	0	0	0
50 Cent	In da club	R	0	0	0	**1**
99 Souls feat. Destiny's Child	The girl is mine	R	**0.7**	0	0	**0.3**
Alicia Keys	Empire state of mind Pt. II	R	**0.6**	0.2	0	0.2
Ariana Grande feat. Nicki Minaj	Side to side	R	**0.4**	0	0	**0.6**
Black Eyed Peas	Just can't get enough	R	**0.5**	0	0.1	**0.4**
Blackstreet feat. Dr. Dre & Bill Whiters	No diggity	R	0.2	0	0	**0.8**
Blu Cantrell feat. Sean Paul & Dr. Dre	Breathe	R	0.1	0	0	**0.9**
Bobby Brown	Two can play that game	R	**1**	0	0	0
The Chainsmokers feat. Daya	Don't let me down	R	**0.9**	0	0	0.1
Chris Brown	Forever	R	**1**	0	0	0
Coolio feat. L.V.	Gangsta's paradise	R	0	0	0	**1**
Craig David feat. Big Narstie	When the bassline drops	R	0	0	0	**1**
Deorro feat. Chris Brown	Five more hours	R	**0.7**	0	0	**0.3**
DESIIGNER	Panda	R	0	0.1	0	**0.9**
Destiny's Child	Survivor	R	**0.9**	0	0	0.1
Emeli Sandé	Read all about it, Pt. III	R	**0.7**	**0.3**	0	0
Fifth Harmony feat. Ty Dolla $ign	Work from Home	R	0	0	0	**1**
Flo Rida	GDFR	R	**0.3**	0	0	**0.7**
Fugees	Killing me softly with his song	R	0	0	0	**1**
Fuse ODG feat. Sean Paul	Dangerous love	R	**0.5**	0	0	**0.5**
G-Eazy feat. Bebe Rexha	Me, myself & I	R	**0.5**	0	0	**0.5**
Grace feat. G-Eazy	You don't own me	R	**0.3**	0	0	**0.7**
Jason Derulo feat. 2 Chainz	Talk dirty	R	**0.3**	0	0	**0.7**
Jennifer Lopez	On the floor	R	**0.5**	0.2	0	**0.3**
Jeremih feat. YG	Don't tell em'	R	0.1	0	0	**0.9**
John Legend	All of me	R	**0.3**	0.1	0.1	**0.5**
Justin Bieber	What do you mean	R	0.1	0.1	0	**0.8**
Justin Timberlake feat. Timbaland	SexyBack	R	**0.3**	0	0	**0.7**
Kelis	Milkshake	R	0	0	0	**1**
Kent Jones	Don't mind	R	0	0	0	**1**
Little Mix feat. Jason Derulo	Secret love song	R	**0.9**	0	0	0.1
Luther Vandross	Never too much	R	**0.8**	0	0	0.2
Major Lazer feat. Nyla & Fuse ODG	Light it up	R	**0.6**	0	0	**0.4**
Mark Morrison	Return of the Mack	R	0	0	0	**1**
Mark Ronson feat. Bruno Mars	Uptown funk	R	**0.3**	0	0	**0.7**

Table 17: Results of the classification of the external test set (page 5).

			Classified as:			
Artist	**Song**	**Label**	**P**	**J**	**C**	**R**
Mary J. Blige	Family affair	R	**0.3**	0	0	**0.7**
Missy Elliot	Get ur freak on	R	0	0	0	**1**
Montell Jordan feat. Slick Rick	This is how we do it	R	0	0	0	**1**
Nicki Minaj	Starships	R	**0.8**	0	0	0.2
OMI	Cheerleader	R	0.2	0.1	0.1	**0.6**
OutKast	Hey ya	R	**0.8**	0	0	0.2
Pete Kennedy	Who cares	R	**0.4**	0	0	**0.6**
Pia Mia feat. Chris Brown, Tyga & J Boog	Do it again	R	0	0	0	**1**
Pitbull feat. Ke$ha	Timber	R	0	0.2	0	**0.8**
R. Kelly & Public Announcement	She's got that vibe	R	0.1	0	0	**0.9**
RITA ORA	I will never let you down	R	**0.8**	0	0	0.2
Rizzle Kicks	Mama do the hump	R	**0.6**	0	0	**0.4**
Robin Thicke feat. Pharell	Blurred lines	R	0	0	0	**1**
Rufus & Chaka Khan	Ain't nobody	R	0.1	0.2	0	**0.7**
Salt-N-Pepa	Shoop	R	0	0.1	0	**0.9**
Snakehips feat. Tinashe & Chance the rapper	All my friends	R	**0.5**	0	0	**0.5**
Soul II Soul feat. Caron Wheeler	Back to life	R	0	0.2	0	**0.8**
SWV feat. Michael Jackson	Right Here	R	**0.3**	0.1	0	**0.6**
Taio Cruz	Dynamite	R	**1**	0	0	0
Timbaland feat. One Republic	Apologize	R	**0.7**	0.2	0.1	0
Tinie Tempah feat. Zara Larsson	Girls like	R	**0.5**	0	0	**0.5**
TLC	No scrubs	R	**0.7**	0	0	**0.3**
Usher feat. Juicy J	I don't mind	R	0	0	0	**1**
The Weeknd	Can't feel my face	R	**0.6**	0.1	0	**0.3**
Whitney Houston	It's not right but it's okay	R	**0.3**	0	0	**0.7**
will.i.am feat. Cody Wise	It's my birthday	R	**0.4**	0	0	**0.6**
WSTRN	In2	R	0.2	0	0	**0.8**
ZAYN	Pillowtalk	R	**0.8**	0	0	**0.2**

Master’s Theses in Mathematical Sciences 2017:E20
ISSN 1404-6342

LUNFMS-3068-2017

Mathematical Statistics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/