Thematically-Driven Guitar Music Retrieval

Daniel Filipe Pombo da Silva Baptista

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Doctor David Manuel Martins de Matos
Doctor Ricardo Daniel Santos Faro Marques Ribeiro

Examination Committee

Chairperson:	Doctor José Manuel da Costa Alves Marques
Supervisor:	Doctor David Manuel Martins de Matos
Member of the Committee:	Doctor Bruno Emanuel da Graça Martins

May 2016

Acknowledgements

At the top of the list, I want to thank my advisor, Prof. David Matos, for his sage advice and insightful criticisms, constantly helping me along this year of work and for always finding a slot for me on his busy schedule. I am also thankful to my girlfriend and family for being kind and supporting.

Lisboa, June 11, 2016
Daniel Filipe Pombo da Silva Baptista

Dedicated to my family and
girlfriend for their endless love,
support and encouragement.

Resumo

Na última década métodos de Query-by-example têm recebido um enorme interesse por parte da comunidade MIREX . Técnicas para identificar a semelhança no audio são bastante utilizadas para um sistema de Query-by-example. No entanto as técnicas atuais de semelhança de áudio ignoram informação sequencial. Neste trabalho, descrevemos as principais abordagens de semelhança de áudio e propomos a adaptação de algumas dessas técnicas, o algoritmo *dynamic time warping* e o modelo *locally weighted bag-of-words* para incorporar informação sequencial. Com esta adaptação o nosso objetivo é proporcionar um sistema Query-by-example com uma melhor correspondência de áudio e a capacidade de lidar com as relações subjacentes, entre as features, que são reconhecidas no domínio musical.

Abstract

In the last decade Query-by-example methods have received a huge interest by the MIREX community. Audio matching is commonly used for a query-by-example model. However current audio matching techniques ignore sequential information. In this work, we describe the main approaches for audio matching and propose the adaptation of some of those techniques, dynamic time warping and locally weighted bag-of-words to incorporate sequential information. With this adaptation our goal is to provide a query-by-example with better audio matching and the ability of dealing with underlying relations between the features that are known to exist in the musical domain.

Palavras Chave
Keywords

Palavras Chave

Extracção de Informação Musical
Query-by-Example
Correspondência de Áudio
Locally Weighted Bag-of-words
Dynamic Time Warping
Harmony Progression Analyzer
MIREX

Keywords

Music Information Retrieval
Query-by-Example
Audio Matching
Locally Weighted Bag-of-words
Dynamic Time Warping
Harmony Progression Analyzer
MIREX

Index

List of Figures

List of Tables

1 Introduction

Large music collections are easily accessible due to rapid evolving technology, providing new opportunities for research using these collections to discover trends and patterns in music (Casey, Veltkamp, Goto, Leman, Rhodes, and Slaney 2008). With music information retrieval, it is possible to develop a strategy to compute similarity between segments of songs, in order to provide new services, such as, detecting copyright violations or query by example. Music teachers, students, and even hobbyists, sometimes need more music material to practice their technical skills, instead of repeating the same pattern/exercise over and over. Students often end up in a situation where they are practicing a technique in a song due to a specific section and have the need for more songs with section/sections similar to that one. The time spent searching is sometimes longer than the time available for practice since it is not easy finding songs with similar sections in a large database.

1.1 Music Information Retrieval

Music information retrieval (MIR) is the interdisciplinary science of retrieving information from music. MIR is a growing field of research with many real-world applications such as creating recommender systems or automatic music transcription. To help strive the MIR community MIREX was created. MIREX is a community dedicated to Music Information Retrieval where several tasks are proposed with the objective of comparing state-of-the-art algorithms in tasks, such as audio cover song detection, onset detection, symbolic music similarity and query-by-humming.

1.2 Query by Example

Query-by-example methods have been target of interest by the MIREX community in the last decade. These methods differ from audio classification as its key issue is how to model each audio segment rather than each audio category (Hu, Liu, Jiang, and Yang 2014). The issue of audio segment modeling is not to be taken lightly (Casey, Veltkamp, Goto, Leman, Rhodes, and Slaney 2008). In the mid-to-low audio similarity specificity range, the user seeks specific musical content of the query audio but not necessarily the same audio content. These are among the most challenging problems for audio similarity-based MIR systems with less specific retrieval tasks still mostly unsolved.

In thematically-driven retrieval, or fragment-level retrieval scenarios, the query consists of a short fragment of an audio recording. The goal is to find all fragments of a given music collection that are related to the query even though entire songs are returned as matches. Typically, such fragments may cover only a few seconds of audio content or may correspond to a theme,

or a musical part of a recording. We can further specify this retrieval being thematically-driven audio matching where the goal is to retrieve all audio fragments that musically correspond to a query fragment from all audio documents contained in a given database. In this case, one explicitly allows semantically motivated variations since they typically occur in different arrangements and performances of a piece of music. These variations include significant non-linear global and local differences in tempo, articulation, and phrasing as well as differences in executing note groups such as grace notes, trills, or arpeggios (Grosche, Müller, and Serrà 2012).

1.3 Problems

The problem we face is the creation of a query-by-example system that thematically retrieves musical pieces given a musical fragment. There are some difficulties inherent to this problem for instance how similar are two songs? There are objective and subjective similarity measures, however for experimental purposes objective measures are preferred. Another difficulty is in the feature extraction process, this can interfere with how we model a song and how we perform audio matching. Another problem is the music genre, such as classical, pop, rock, among others. Finally the key issue lies in the method we use to perform our audio modeling.

1.4 Goals

Given a musical fragment as input, produce as output a ranked list of thematically related musical pieces based on tempo, articulation and phrasing. Our system will be based on the locally weighted bag-of-words (Lebanon, Mao, and Dillon 2007; Lebanon 2012) to easily identify thematic trends across songs. Popular music will be used in this work for evaluation purposes given its wide corpus availability.

1.5 Structure

This thesis is divided into three main logical parts. The first part (chapter 2) is related to the state-of-the-art which describes how audio matching is done as of the writing of this thesis. We start by reviewing a continuous probability distribution model, Gaussian Mixture Models in audio similarity retrieval, we analyze bag-of-audio-words approaches similar to the bag-of-words model used in text processing, and we review applications of topic models to audio retrieval, such as Hierarchical Dirichlet Process and Latent Dirichlet Allocation.

The second part (chapters 3 and 4) describe the work and subsequent achieved results. It starts with a case study regarding feature extraction and audio matching and respective experiments/results, then the construction of the query-by-example system and obtained results with the datasets.

The third part (chapter 5) is made of conclusions as well as the definition of future work to be done.

2 State of the Art

The chapter is divided into two main sections: section 2.1 begins by describing the typical architecture for a query-by-example system. It also describes each module of the architecture in greater detail. Section 2.2 presents some query-by-example systems taking into account the architecture described in section 2.1. Finally, we discuss and build a comparison table of all of the systems.

2.1 Architecture for Query-by-Example System

Query-by-Example systems share a common three stage architecture as shown in Figure 2.1.

From the audio query, systems start by using feature extraction module followed by a query engine, followed by an audio matching stage. Table 2.1 describes the main responsibility of each stage. The following subsections will explain each of the mentioned steps in greater detail.

Stage	Responsibility
Feature Extraction	Audio signal characteristics
Query Engine	Robust feature modeling
Audio Matching	Distance comparison and ranked sort

Table 2.1: Typical Query by Example Architecture Description.

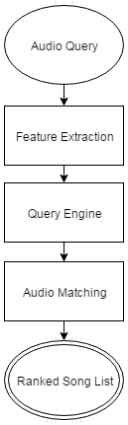

Figure 2.1: Typical Stages for Query-by-Example.

2.1.1 Feature Extraction

For audio matching to work we must first obtain features that represent the audio signal with a certain degree of robustness. Examples of those features are: timbre, melody, rhythm, pitch, among others. Depending on the task at hand, some features will be more useful than others. For instance, if the task is musical instrument recognition or genre detection, a timbre related feature, such as Mel-frequency Cepstrum Coefficients, will be more useful than a rhythm feature.

A good feature vector in this case is one that can represent audio structure properly while also having low dimensionality and robustness to some harmonic changes as described in Section 1.2.

2.1.1.1 Chroma Vector

The most popular feature currently used for the task of audio matching is the chroma vector, in particular for detecting harmony-based relations. Chroma features have turned out to be a powerful mid-level representation for comparing and relating music data in various realizations and formats.

Chroma-based audio features are obtained by pooling a signal's spectrum into twelve bins that correspond to the twelve pitch classes or chroma of the equal-tempered scale. Identifying pitches that differ by an octave, chroma features show a high degree of robustness to variations in timbre and are well-suited for the analysis of Western music which is characterized by a prominent harmonic progression. The temporal evolution of these chroma vectors is called chromagram, It has been widely used in literature for audio matching to better understand the degree of robustness of the chroma features.

2.1.1.2 Chroma Variations

There have been several improvements/variations over the standard chroma vector. Two considerably important variations for audio matching can be found in the Chroma Toolbox Chroma Energy distribution Normalized Statistics CENS and CRP (Müller and Ewert 2011a).

2.1.1.2.1 CENS In computing CENS features, each chroma vector is first normalized with respect to the $L1$-norm thus expressing relative energy distribution. Then, a quantization is applied based on suitably chosen thresholds. Here, choosing thresholds in a logarithmic fashion introduces some kind of logarithmic compression.

In a subsequent step, the features are further smoothed over a window of length w and downsampled by a factor of d. The resulting features are normalized with respect to the $L2$-norm and denoted by CENS$_w d$

The CENS feature sequences correlate closely with the smoothed harmonic progression of the underlying audio signal. Other parameters, however, such as dynamics, timbre, or articulation are masked out to a large extent.

The normalization makes the CENS features invariant to dynamic variations. Furthermore, using chroma instead of pitches not only takes into account the close octave relationship in both melody and harmony as typical for Western music, but also introduces a high degree of

robustness to variations in timbre. Then, applying energy thresholds makes the CENS features insensitive to noise components as may arise during note attacks.

Finally, taking statistics over relatively large windows not only smooths out local time deviations as may occur for articulatory reasons but also compensates for different realizations of note groups such as trills or arpeggios (Müller, Kurth, and Clausen 2005).

2.1.1.2.2 CRP Starting with the Pitch features, a logarithmic compression is applied and transforms the logarithmized pitch representation using a Discrete Cosine Transform (DCT). Only the upper coefficients of the resulting pitch frequency cepstral coefficients (PFCCs) are kept. Afterwards an inverse DCT is applied and finally the resulting pitch vectors are projected onto 12-dimensional chroma vectors, which are then normalized with respect to the $L2$-norm.

These vectors are referred to as CRP (Chroma DCT-Reduced log Pitch) features. The upper coefficients to be kept are specified by a parameter $n \in [1 : 120]$. As reported by Müller and Ewert (2010), the parameter $n = 55$ yields good results. The resulting features are denoted by CRP[n] (Müller, Ewert, and Kreuzer 2009).

2.1.1.3 Chords

Chords are mid-level musical features which concisely describe the harmonic content of a piece. This is evidenced by chord sequences often being sufficient for musicians to play together in an unrehearsed situation (McVicar, Santos-Rodríguez, Ni, and De Bie 2014).

Ni, Mcvicar, Santos-Rodriguez, and De Bie (2012) proposed a new machine based system called Harmony Progression Analyzer (HPA).

HPA is a machine learning system for the harmonic analysis of popular musical audio. It is focused on chord estimation, although the proposed system additionally estimates the key sequence and bass notes (Ni, Mcvicar, Santos-Rodriguez, and De Bie 2011).

2.1.2 Query Engine

After the features have been extracted we need to represent the extracted features in a modeling framework. The chosen model can be Gaussian Mixtures, Histograms or Topic models, and the model is implicitly related with the audio matching stage (Hu, Liu, Jiang, and Yang 2014; Helén and Virtanen 2007; Casey, Veltkamp, Goto, Leman, Rhodes, and Slaney 2008). These models will be further described in section 2.2. Afterwards, the received query also needs to have its features modeled so that we can move to the audio matching stage where we compare the query against the database.

2.1.3 Audio Matching

Audio Matching is the stage where we compare the query and database given a certain feature and representation of choice. Since we are interested in comparing sequences of different lengths, the queries are usually short fragments and songs are comprised of multiple fragments. This stage becomes one of solving a subsequence audio matching problem. The comparison is made through a distance measure where we calculate the similarity between a

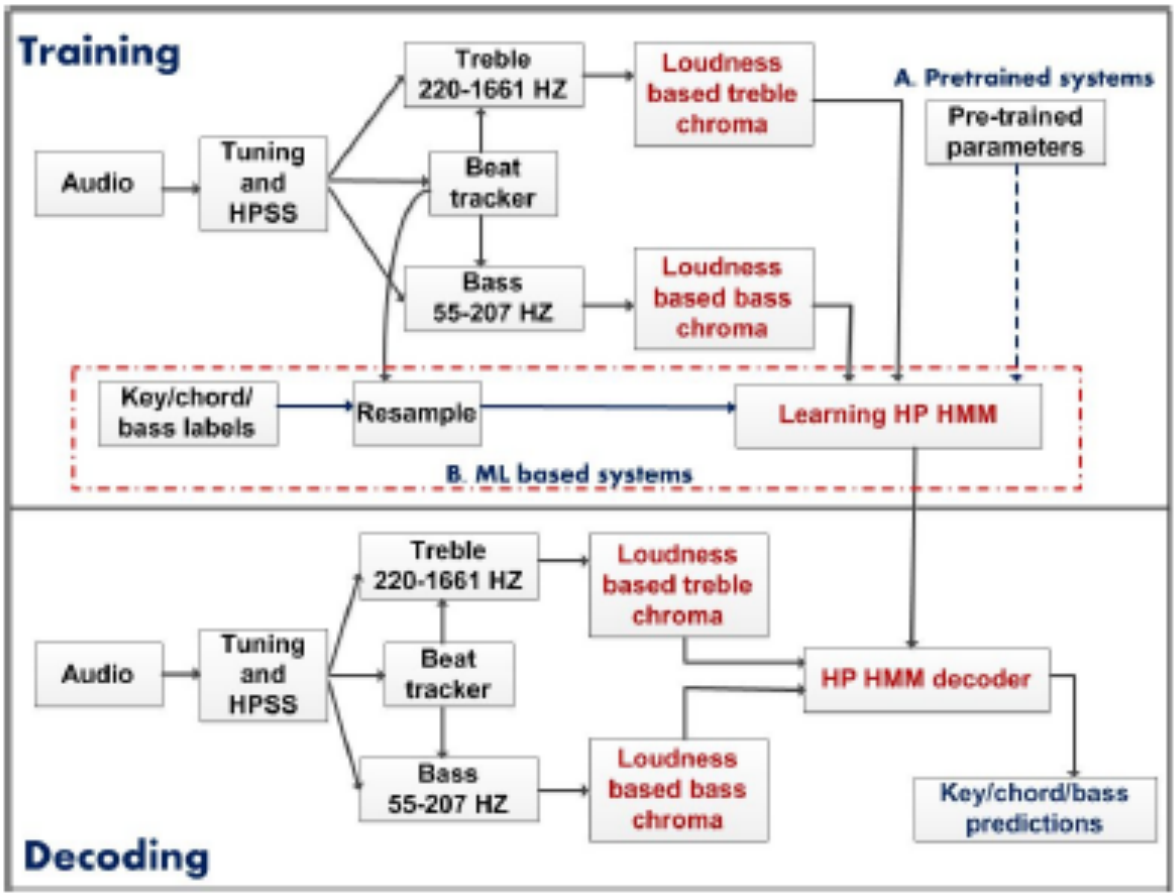

Figure 2.2: Components of the harmony progression analyzer (HPA). (Taken from (Ni, Mcvicar, Santos-Rodriguez, and De Bie 2011))

query and a song. In the literature there are various algorithms proposed to handle the subsequence matching, most of them use dynamic programming, such as Dynamic Time Warping (Shokoohi-Yekta, Wang, and Keogh 2015), SPRING (Sakurai, Faloutsos, and Yamamuro 2007), Time Warped Longest Common Subsequence (TWLCS) (Guo and Siegelmann 2004). The purpose of this stage is to automatically retrieve and rank all songs that musically correspond to a query from all the audio documents contained in the database.

A typical approach in global matching is to calculate the euclidean distance between the query and the songs but other more meaningful approaches dependent on the modeling can also be used, such as Fisher's distance (Lebanon, Mao, and Dillon 2007).

2.1.3.1 Dynamic Time Warping

The main idea of Dynamic Time Warping (DTW) is to align two sequences without the restriction of having same length and find point-to-point alignment that minimizes the error between the two sequences. In DTW, an individual element of one sequence can be matched with at least one and possibly more elements of the other sequence, thus allowing for each sequences to be stretched locally along the time axis. This method is usually computed by dynamic programming. The dynamic time warping cost $D(i, j)$ is defined as follows:

$$D(0, 0) = 0. \tag{2.1}$$

$$D(0, j) = \infty. \tag{2.2}$$

$$D(i, 0) = \infty. \tag{2.3}$$

$$D(i, j) = d(i, j) + min \begin{cases} D(i, j-1) \\ D(i-1, j) \\ D(i-1, j-1) \end{cases} \tag{2.4}$$

$$\forall (i = 1, ..., |Q|; j = 1, ..., |X|)\ D(Q, X) = D(|Q|, |X|). \tag{2.5}$$

Notice that $d(i, j)$ is the Lp norm difference of i and j.

By placing an additional constraint, which narrows down the set of positions in one sequence that can be matched with a specific position in the other sequence we obtain the Constrained DTW (cDTW). Given a warping width w, the constraint is defined as follows:

$$D(i, j) = \infty \text{ IF } |i - j| > w \tag{2.6}$$

cDTW has been shown to be significantly more efficient than DTW for full sequence matching and to also produce more meaningful matching scores.

Shokoohi et al. studied the Multidimensional DTW case. The DTW distance is applicable to only single-dimensional sequences leaving open the question of how to extend it to the multidimensional sequences. Considering two M-dimensional sequences Q and C, the authors show two possible approaches for doing this, DTW_I and DTW_D.

DTW_I is the cumulative distance of all dimensions independently measured under DTW. If DTW(Qm,Cm) is defined as the DTW distance of the mth dimension of Q and the mth dimension of C, we can write DTW_I as:

$$DTW_I(Q,C) = \sum_{m=1}^{M} DTW(Qm, Cm) \tag{2.7}$$

In Eq. 2.7, each dimension is considered to be independent and DTW is allowed the freedom to warp each dimension independently of the others.

The multi-dimensional DTW can also be computed in a manner that forces all dimensions to warp identically. In other words, the independence of dimensions is no longer allowed and the we assume mutual dependence between all dimensions.

DTW_D is calculated in a similar way to DTW for single-dimensional sequences, except that we redefine $D(q_i, c_j)$ as the cumulative squared Euclidean distances of M data points instead of the single data point used in the more familiar one-dimensional case. Formally, if $q_{i,m}$ is the i^{th} data point in the m^{th} dimension of Q and $c_{j,m}$ is the j^{th} data point in the m^{th} dimension of C, we replace $D(q_i, c_j)$ with:

$$D(q_i, c_j) = \sum_{m=1}^{M} (q_{i,m} - c_{j,m})^2 \tag{2.8}$$

2.1.3.2 SPRING

Both DTW and cDTW described above, need to use a sliding window in order to determine optimal subsequence match of a query in a large database. SPRING uses the same recursive definitions as those used by DTW, however it allows a warping path to start at any position of the target sequence and not always the first as in the case of DTW (Sakurai, Faloutsos, and Yamamuro 2007). This is possible due to an extra "sink" state, the differences in regard to DTW equations 2.2 and 2.5 are respectively:

$$D(0, j) = 0. \tag{2.9}$$

$$D(Q, X) = \min_{j \in 1,\ldots,|X|} D(|Q|, j). \tag{2.10}$$

$$T(i,j) = min \begin{cases} T(i-1, j-1) + S(x_i, y_i) \\ T(i-1, j) + gapPenalty \\ T(i, j-1) + gapPenalty \end{cases} \tag{2.11}$$

2.1.3.3 Time-Warped Longest Common Subsequence Algorithm

TWLCS was conceived in a query by humming scenario where they want to deal with singing errors involving rhythmic distortions. Specifically it merges DTW with the Longest

Common Subsequence algorithm (LCS) (Guo and Siegelmann 2004). The longest common subsequence algorithm belongs to the edit distance family of string matching algorithms, specifically, the LCS finds the longest subsequence that two sequences have in common, regardless of the length and number of intermittent non-matching symbols.

Formally, the LCS algorithm has the following recurrence equation, where the cost for the edit operations is stored in c.

$$c(i,j) = \left\{ \begin{array}{ccccccc} 0 & & \text{IF} & i=0 & \text{or} & j=0 \\ c(i-1,j-1)+1 & & \text{IF} & i,j>0 & \text{and} & Xi=Yj \\ max[c(i,j-1),c(i-1,j)] & & \text{IF} & i,j>0 & \text{and} & Xi \neq Yj \end{array} \right\} \tag{2.12}$$

Using LCS as a similarity measure between two sequences has the advantage that the two sequences we are comparing can be of different length and have intermittent non-matches.

Taking the desirable properties of LCS and DTW, TWLCS is defined by the following recurrence formula:

$$c(i,j) = \left\{ \begin{array}{ccccccc} 0 & & \text{IF} & i=0 & \text{or} & j=0 \\ max[c(i,j-1),c(i-1,j)],c(i-1,j-1)+1 & & \text{IF} & i,j>0 & \text{and} & Xi=Yj \\ max[c(i,j-1),c(i-1,j)] & & \text{IF} & i,j>0 & \text{and} & Xi \neq Yj \end{array} \right\} \tag{2.13}$$

Combining DTW ability to handle the expansion and contraction of the sequences with LCS ability that two sequences being compared can be of different length and have intermittent non-matches, in the music retrieval context this allows for the use of partial and noisy inputs and also variations in speed.

For example, if 44556677 or 42536172 were matched against 4567, the output should be a higher score in the first match, but LCS outputs 4 in both. With TWLCS the match between 44556677 with 4567 receives a score of 8.

2.2 Related Work

The general consensus in audio matching tasks is that the approaches can be grouped into three large categories: Gaussian Mixture Models (GMM), Histograms (Bag-of-Words) and Topic Models (Hu, Liu, Jiang, and Yang 2014; Helén and Virtanen 2007; Casey, Veltkamp, Goto, Leman, Rhodes, and Slaney 2008).

2.2.1 Music Retrieval using Gaussian Mixture Modeling

A Gaussian Mixture Model (GMM) is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. GMM estimates probability density as the weighted sum of these simpler Gaussian densities, called components of the mixture. They are often used for clustering. The clusters are assigned by the component that maximizes the posterior probability, and like k-Means, GMM uses an iterative algorithm.

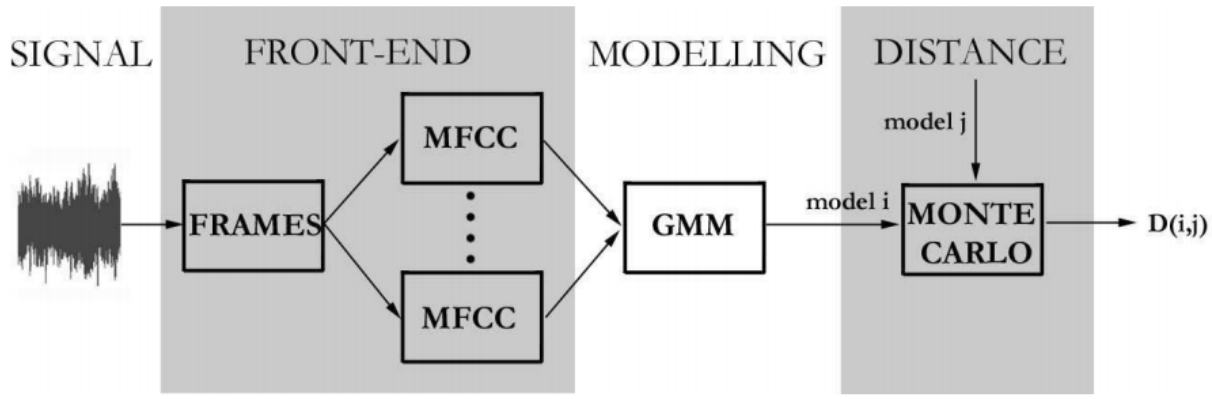

Figure 2.3: Block diagram for GMM Representation (Pachet and Aucouturier 2004).

The Gaussian Mixture Model is defined in Eq. 2.14

$$p(F_t) = \sum_{m=1}^{M} c_m N(Ft, \mu_m, \tau_m). \tag{2.14}$$

where F_t is the feature vector observed at time t, N is a Gaussian pdf with mean μ_m, covariance matrix τ_m, and c_m is a mixture coefficient.

GMMs of Mel-Frequency Cepstrum Coefficients have been widely used and researched in MIR (Aucouturier and Pachet 2002b; Aucouturier and Pachet 2002a; Helén and Virtanen 2007; Aucouturier and Pachet 2008; Casey, Veltkamp, Goto, Leman, Rhodes, and Slaney 2008). Since this approach has been widely researched, its strong and weak aspects are well known. In terms of high-level descriptions of music signals, such as genre, mood or computing timbre similarity between songs, the GMM approach is the most predominant paradigm having led to some success. However, latest research by Aucouturier and Pachet shows this approach has a performance glass-ceiling for polyphonic timbre similarity at around 70% precision even after exhaustive fine-tuning or applying delta-coefficients or Markov modeling. One possible cause for this glass-ceiling in precision is the existence of hubs, false positives which are mostly the same songs regardless of the query, so hubs are songs which are irrelevantly close to all other songs. Further research regarding these hubs has been done by Aucouturier, Defreville, and Pachet (2007) and Pachet (2008).

2.2.1.1 Aucouturier et Pachet (2004)

The authors researched the application of Gaussian Mixture Models for music similarity purposes and its similarity measures (Aucouturier and Pachet 2002a; Aucouturier and Pachet 2002b; Pachet and Aucouturier 2004). Their main music similarity focus is regarding global timbral quality, since music taste is often correlated with timbre. Additionally timbre similarity is a natural way to build relations between music titles.

For the feature extraction they cut the signal into 2048 points frames (50ms) and computed for each frame the short-time spectrum. Afterwards, they used Mel Frequency Cepstrum to estimate the spectral envelope of each frame, in order to obtain a timbre measure independent of pitch they only used the first 8 coefficients resulting in a feature vector of dimension 8 for

each frame. The choice of MFCC is that this feature explains a large part of the timbre of instruments and is a good representation of the "local timbre" of the frame.

In the Gaussian Mixture Modeling they initialized the GMMs parameters by k-Mean clustering and modeled the distribution of each song's MFCCs as a mixture of Gaussian distributions over the space of all MFCCs. The authors trained the model with the Expectation-Maximization algorithm (Bishop 1995). They thoroughly explored the influence of the following parameters with the original algorithm, the Signal Sample Rate (SR), number of MFCCs (N), number of components (M), distance sample rate (DSR), and window size by fixing all but one parameter at a time to evaluate its influence.

Their findings were that the SR had a positive influence on the precision, probably due to the increased bandwidth of the higher definition signals which enables the algorithm to use higher frequency components than with low SR. The DSR also has a positive influence on the precision when it increases from 1 to 1000 (further increase has little influence). They also found that the optimal DSR is not dependent of either N or M. In regards to N and M, they made a complete exploration of the associated 2-D space having N vary between 10 to 50 by steps of 10 and M from 10 to 100 by steps of 10. This showed that the previous values for the chosen algorithm of N=8 and M=3 were not optimal (Aucouturier and Pachet 2002a). Too many MFCCs ($N \geq 20$) hurt the precision. Increasing the number of components at fixed N, and increasing N at fixed M is detrimental to the precision as well. They also note that the number of MFCCs is a more critical factor than the number of Gaussian components therefore a decrease in M to values smaller than the optimal does not hurt the precision. The window size experiment showed that it has a small positive influence on the precision when it increases from 10 to 30ms, but further increase up to 1s has a negative effect.

After the experiment, the optimal values for their music similarity task were a value of 44kHz for the Sample rate, the number of chosen MFCCs was 20, the number of Gaussian components used to model MFCCs was 50 Gaussian distributions, and a distance sample rate of 2000. Then, they computed the distance between models with the classical Kullback-Leibler distance (Bishop 1995).

The corpus (Aucouturier and Pachet 2002a) consisted of 17,075 popular music titles together with metadata such as information about artists, genres, among others. For each title its "timbral distance" was computed against all the other titles and the evaluation showed very poor results regarding a query on genre based on timbral distance (14.1%).

A different evaluation metric was used by Pachet and Aucouturier (2004): the R-precision which is the standard within the Text Retrieval Conference, and is the precision measured after R documents have been retrieved, R being the number of relevant documents. For a given query on a song S_i belonging to a cluster C_{S_i} of size N_i, the precision is given by Eq. 2.15,

$$p(S_i) = \frac{|(S_k / C_{S_k} = C_{S_i} \text{ and } R(S_k) \leq N_i)|}{N_i}, \tag{2.15}$$

where $R(S_k)$ is the rank of song S_k in the query on song S_i. The corpus consisted of 350 songs from 37 artists chosen in order to have "timbrally" consistent clusters.

The evaluation obtained was 48 R-Precision (Aucouturier and Pachet 2002a). With further optimization taking into account the exploration on the influence of the parameters (Pachet and Aucouturier 2004), they were able to raise the value to 65 R-Precision and they suggested there was a glass-ceiling at around this precision.

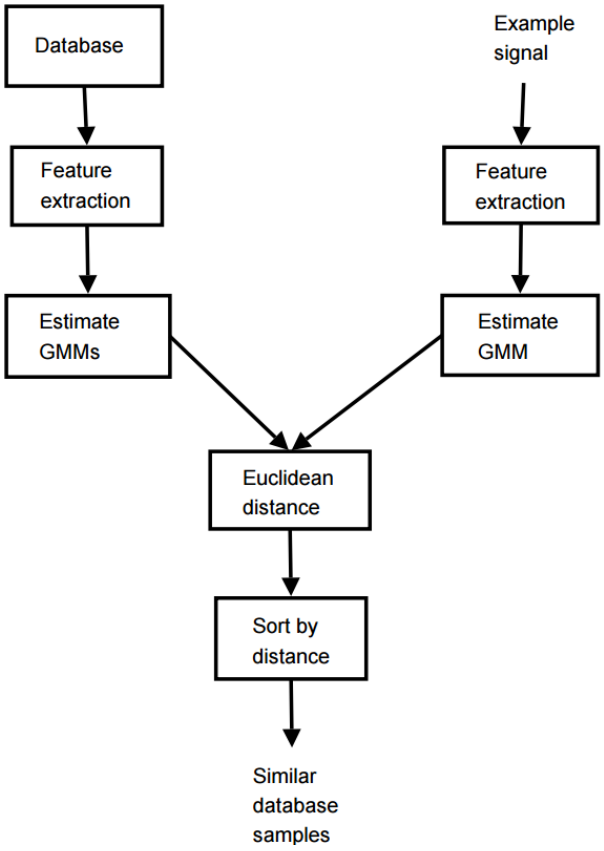

Figure 2.4: Query-by-example GMM Architecture (Helén and Virtanen 2007).

2.2.1.2 Helén et Virtanen (2007)

GMMs were used to develop a query-by-example system. Additionally, the authors also defined a method to calculate the Euclidean distance between two GMMs for audio retrieval (Helén and Virtanen 2007).

For their feature extraction step, the input signal was divided into 46ms frames and a set of features was extracted for each frame. The frequency content of the frame is characterized using three Mel-frequency cepstral coefficients, spectral centroid, noise likeness, spectral spread, spectral flux, harmonic ratio, and maximum autocorrelation lag. The Temporal characteristics of the signal are described using zero crossing rate, crest factor, total energy, and variance of instantaneous power. Each of the extracted features is normalized to have zero mean and unity variance over the whole database.

In the Gaussian Mixture Modeling the authors used two methods for estimating the parameters of the GMMs. The first, used Expectation-Maximization algorithm to estimate the means and variances for a fixed number of components. The second used the Parzen-window approach which assigns a GMM component with fixed variance for each observation. The number of Gaussians used in the EM algorithm was 8 and the feature variances were restricted above unity. In the Parzen-Euclidean method the authors tested different variances and $\sigma^2 = 2$ produced approximately the best results.

Regarding the similarity measure the Euclidean distance was used. It consists of determin-

ing the similarity of two samples by measuring the square of the Euclidean distance between their Gaussian distributions $p_1(\mathrm{x})$ and $p_2(\mathrm{x})$. This is obtained by integrating the squared differences over the whole feature space:

$$e = \int_{-\infty}^{\infty} [p_1(\mathrm{x}) - p_2(\mathrm{x})^2] d\mathrm{x}. \tag{2.16}$$

The corpus consisted of 240 samples with 16kHz sampling rate. The lengths varied between 5 and 30 seconds and the samples were manually annotated into 4 classes. The evaluation comprised of drawing one sample at a time to serve as an example for a query and the rest served as the database. A database sample was correctly retrieved if it was annotated into the same class as the example.

The query was repeated using each of the S samples as the example, resulting in altogether $S(S-1)$ pairwise comparisons. The number of correctly retrieved samples c_u was calculated for each class $u \in 1, 2, 3, 4$. The ratio of correctly retrieved samples to all the comparisons is given by the average of recall of each query

$$recall(u) = \frac{c_u}{S_u(S_{u-1})}, \tag{2.17}$$

where S_u is the total number of samples in the class u. The ratio of correctly classified samples to all the samples r_u retrieved for class u examples is given by the precision

$$precision(u) = \frac{c_u}{r_u}. \tag{2.18}$$

The overall precision and recall were estimated for the whole database as the average of the class-wise precision and recall.

Obtained results were higher in comparison to previous query by example methods based on histograms of features and likelihoods of GMMs (Helén and Virtanen 2007). The Parzen-Euclidean method produced the best results on average, with 5 p.p higher average precision and recall than the EM-likelihood method and 13 p.p higher than the histogram method. The comparison between likelihoods of GMMs showed that the average precision and recall rates increased from 65% to 70% with the Parzen-Euclidean.

2.2.1.3 Summary

GMM is typically an unsupervised technique and is also a basic but useful algorithm to be used in clustering.

The main advantage of using GMM is because it is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters therefore there is no loss of information.

The main disadvantages are that it is hard to fit the best mixture of Gaussians as could be seen in the work by Aucouturier et Pachet, the fitting is exponential in the assumed number of latent Gaussian distributions, and GMMs have achieved a glass-ceiling in precision (Aucouturier and Pachet 2002a).

2.2.2 Music Retrieval using Histograms

Bag-of-words models can be generically defined as an orderless document representation given certain frequencies of words from a dictionary, in other words, a sparse vector of occurrence counts of words. These models are characterized by having two main steps.

After performing a feature extraction step, where features of choice, such as Chroma or MFCC are extracted, the feature representation step is performed. Each music is abstracted by several vectors called feature descriptors. A good descriptor should have the ability to be robust to changes up to some extent.

Vector quantization or codebook generation consists in converting vector represented features of a song to audio-words, which also produces a "codebook". An audio-word can be considered as a representative of several similar feature vectors. Typically, the simplest method is performing k-Means clustering over all the vectors. Audio-words are then defined as the centers of the learned clusters. The number of the clusters is the codebook size. Finally, each song can be represented by the histogram of the audio-words, given the centers of the learned clusters and audio-word frequencies, this final step is named, histogram construction.

The representation of a song by its histogram of audio-words is rich, and is also capable of capturing the sequential information by extending it to weight the audio-words based on the histogram.

Bag-of-words have been widely applied and researched in text domain obtaining good results (Lebanon, Mao, and Dillon 2007; Lebanon 2012). It also has been applied to the music domain with good results (Riley, Heinen, and Ghosh 2008; Zhu 2013; Lu and Cabrera 2012).

2.2.2.1 Lebanon et al.(2007)

Lebanon et al. created a representation of the bag-of-words where it is possible to have a continuous and differentiable sequential representation (Lebanon, Mao, and Dillon 2007). This representation goes beyond the traditional bag-of-words representation and its n-gram extensions by being able to capture sequential information, and yet it is efficient and effective. It is called locally weighted bag-of-words (LOWBOW).

The smoothing method employed in the traditional bag-of-words model is categorical rather than temporal since no time information is preserved. And temporal smoothing has far greater potential than categorical smoothing since a word can be smoothed out to varying degrees depending on the temporal difference between the two document positions. With this in mind, the main idea behind the LOWBOW is to use a local smoothing kernel to smooth the original word sequence temporally, by borrowing the presence of a word at a certain location in the document to a neighboring location but discounting its contribution depending on the temporal distance between the two locations. To handle the problem that several words can occupy one location through temporal smoothing of words, the authors provided a broader definition of a document (Lebanon, Mao, and Dillon 2007) ultimately resulting in an association between a document location with a local histogram or a point in the simplex.

The multinomial simplex $\mathbb{P}_m$ for $m > 0$ is the m-dimensional subset of $\mathbb{R}^{m+1}$ of all

probability vectors or histograms over $m+1$ objects

$$\mathbb{P}_m = \theta \in \mathbb{R}^{m+1} : \forall i \theta_i \geq 0, \sum_{j=1}^{m+1} \theta_j = 1. \tag{2.19}$$

Its connection to the multinomial distribution is that every $\theta \in \mathbb{P}_m$ corresponds to a multinomial distribution over $m+1$ items.

The topological structure of $\mathbb{P}_m$, which determines the notions of convergence and continuity, is naturally inherited from the standard topological structure of the embedding space $\mathbb{R}^{m+1}$. The geometrical structure of $\mathbb{P}_m$ that determines the notions of distance, angle, and curvature is determined by a local inner product $g\theta(.,.), \theta \in \mathbb{P}_m$, called the Riemannian metric.

The authors proved theorems regarding the LOWBOW, specifically

Theorem 1 *The LOWBOW representation is a continuous and differentiable parameterized curve in the simplex, in both the Euclidean and the Fisher geometry.*

Theorem 2 *Let $K_{\mu,\sigma}$ be a smoothing kernel such that when $\sigma \to \infty$, $K_{\mu,\sigma}(x)$ is constant in μ, x. Then for $\sigma \to \infty$, the LOWBOW curve $\gamma(y)$ degenerates into a single point corresponding to the bow representation.*

Theorem 3 *The LOWBOW curve $\gamma(y)$ satisfies $||\gamma_\mu(y) - \gamma_\tau(y)||_2 \leq |\mu - \tau| O(K), \forall \mu, \tau \in [0,1]$.*

Where O(K) is a Lipschitz constant. As a result of the lowbow curve being Lipschitz continuous, the curve complexity is connected with the shape and scale of the kernel. Thus, we can represent LOWBOW in a finite dimensional space by sampling the path at representative points $\mu_1, ..., \mu_l \in [0,1]$.

Given a Riemannian metric g on the simplex, its product form

$$g'_\theta(u,v) = \int_0^l g_{\theta(t)}(u(t), v(t))dt \tag{2.20}$$

defines a corresponding metric on LOWBOW curves. This results in geometric structures that are compatible with the base metric g, such as distance or curvature. For example, the distance between LOWBOW representations of two word sequences $\gamma(y), \gamma(z) \in \mathbb{P}_m^{[0,1]}$ is the average distance between the corresponding time coordinates

$$d(\gamma(y), \gamma(z)) = \int_0^1 d(\gamma_\mu(y), \gamma_\mu(z))d\mu \tag{2.21}$$

The integrated distance formula in Eq. 2.21 allows the possibility to adapt distance-based algorithms to the LOWBOW representation. To use a distance-based algorithm with LOWBOW we just need to replace its standar distance such as the Euclidean distance with LOWBOW's integrated distance or its discretized version.

Figure 2.5: Block diagram for "Bag-of-Audio-Words" Representation (Riley, Heinen, and Ghosh 2008).

In summary, this representation generalizes bag-of-words by considering the collection of local word histograms throughout the document. In contrast to n-grams, which keep track of frequently occurring patterns independent of their positions, LOWBOW keeps track of changes in the word histogram as it sweeps through the document from beginning to end (Lebanon, Mao, and Dillon 2007). In contrast to n-gram, LOWBOW captures topical trends, and incorporates long range information. On the other hand, it is possible to combine the two, so the LOWBOW is orthogonal to n-gram (Lebanon 2012). The ability to capture topical trends and keep track of sequential information is by having the LOWBOW curves.

LOWBOW is represented employing smooth curves in the multinomial simplex. With this representation there are interesting geometrical features to be used in modeling, applied to tasks of retrieval, classification, filtering, segmentation, and visualization. The distance between LOWBOW curves can be used in various modeling tasks, such as K-nearest neighbors, SVM, or even constructing generative models. Other geometrical features can be used, such as the instantaneous direction of the curve which describes sequential topic trends and their change.The authors showed several applications of the LOWBOW framework, such as to text classification with nearest neighbors or support vector machines, text segmentation tasks and applied Dynamic Time Warping of LOWBOW curves.

The LOWBOW framework achieved good results in practice for the referred experiments. And it has also achieved good results in the scope of video however it has still not been applied in MIR.

2.2.2.2 Riley et al. (2008)

Riley, Heinen, and Ghosh (2008) took the Bag-of-words approach from text retrieval and applied it in audio similarity retrieval, renaming the representation to a Bag-of-Audio-Words (BOAW). They also showed that a technique based on methods for text retrieval performs well, having practical applicability and benefiting from established research in the area.

First they segmented the audio in non-overlapping 200 millisecond clips. Then for the feature extraction they chose the normalized 12-dimension Chroma feature. Their main reasoning behind this choice is the performance of the audio-word histogram representation, this way feature vectors for an audio segment and distorted version are very similar.

Then, they performed a vector quantization step, which consists mostly of performing clustering in the 12-dimensional Chroma space. The clustering identifies k dense regions within a set of Chroma features extracted from the data set, referred as audio-words. Thereafter, the nearest audio-word is calculated for any Chroma feature extracted from a song segment and that segment will be considered as an occurrence of that audio-word. For the clustering, they collected approximately 100,000 Chroma vectors from a variety of songs separate from the test set and used k-Means to compute the k audio-words.

In summary, the vector quantization takes as input a song's sequence of Chroma vectors, and for each outputs one or more numbers corresponding to the closest audio-word(s), measured by the Euclidean distance.

Finally, the last step is the histogram construction, where the song x is mapped to a k-dimensional vector, which encodes the frequency of each audio-word occurrence in the song. Since they represented individual Chroma vectors by the cluster center to which they belong, it made equivalent in the histogram representation any segments musically very similar but with slightly different Chroma vectors then the k terms of the audio-word histograms are weighted according to the term-frequency inverse document frequency (TF-IDF) scheme.

They experimented with several clustering algorithms in the Vector Quantization besides K-Means, such as GMMs with expectation maximization. However most of these other algorithms involved greater computation and did not improve the Vector quantization step. Interestingly they experimentally determined the matching performance could be improved by assigning the Chroma vector to three closest centroids instead of one.

The similarity measures used for computing the similarity between audio-word histograms were the Cosine Similarity, Chi-Squared Similarity, and normalized Euclidean distance with the Chi-Squared being the best (Riley, Heinen, and Ghosh 2008). Given a query song or audio clip they use the similarity measure to return a ranked list of similar songs from the database.

The evaluation was done with a data set of 4000 songs from a variety of musical genres. 60 additional tracks were selected as query songs and quantify the system's ability to correctly retrieve distorted versions of the tracks from within the 4060 total song set. For the first experiment they evaluated the robustness to signal distortion applying a signal distortion to each of the 60 query songs and calculated, for each similarity measure, the percentage at which the distorted query songs were most similar to the original query songs.

The results showed that the Bag-of-Audio-Words approach had excellent retrieval accuracy for a wide variety of distortions and was found useful in matching original studio recordings with live performances and cover songs to a lesser degree. However with this approach all the time-series information present in the initial song is ignored, the authors suggest this would lead to even better performance results.

2.2.2.3 Grosche et al. (2012)

The authors discuss audio-similarity based retrieval strategies that follow the query-by-example paradigm (Grosche, Müller, and Serrà 2012). They classify these strategies according to their specificity, the degree of similarity between the query and the database documents: so high-specificity refers to a strict notion of similarity whereas low specificity a vague one. And they also classify based on granularity, where one distinguishes between fragment-level and document-level retrieval. The classified strategies can be seen in figure 2.6. For the audio identification, audio matching, and version identification the authors give an overview of representative state-of-the-art approaches.

In Audio Matching, they used chroma-based audio features, with the reasoning being that the descriptors had to be invariant to properties of a particular recording: chroma features are a well-established tool for analyzing Western tonal music and suitable mid-level representation

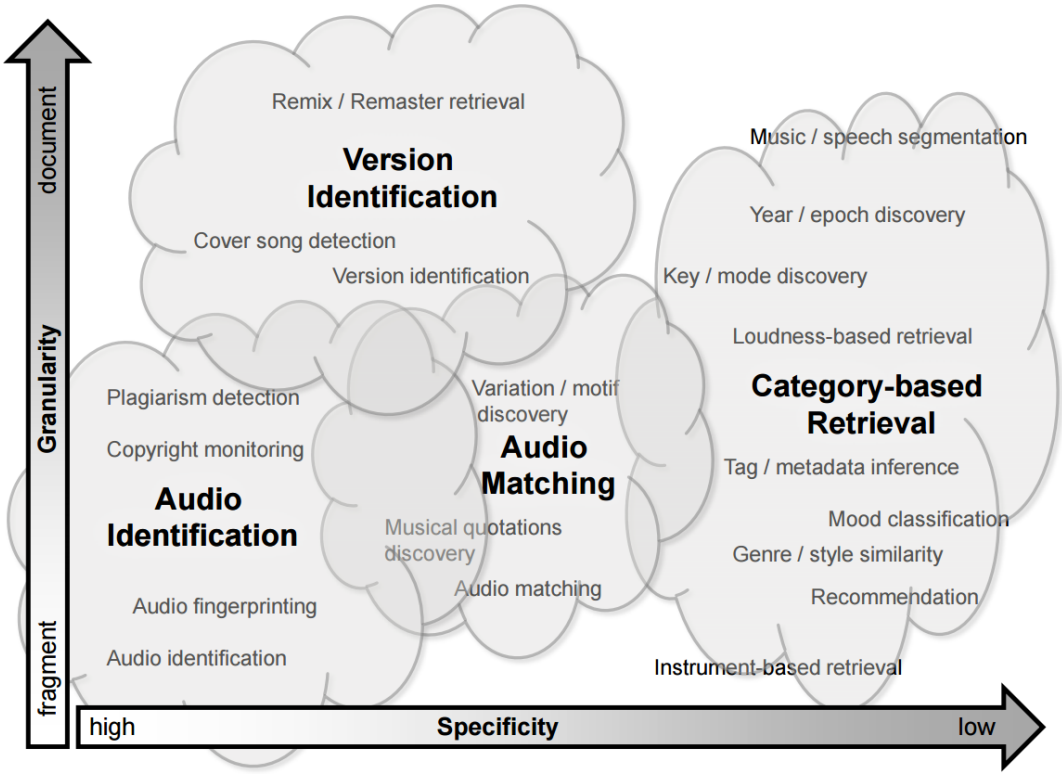

Figure 2.6: Specificity/granularity pane showing the various facets of content-based music retrieval (Grosche, Müller, and Serrà 2012).

in the retrieval context, besides closely correlate to the harmonic progression of the underlying piece of music.

Regarding the feature extraction, they discuss and present various methods for computing chroma features and also discuss the degree of robustness of the chroma features against musically motivated variations depending on suitable post-processing steps. The robustness can vary from robustness to timbre to being invariant to changes in loudness or dynamics, or even being more robust against local temporal variations. The resulting chroma features can behave quite differently in the subsequent analysis task. For more information on variants of chroma features and how to extract them see (Müller and Ewert 2011b).

The evaluation consists on a subsequence search, directly performed on chroma features. Therefore a query chromagram is compared with all subsequences of database chromagrams. Accordingly, the similarity measure consists of obtaining a matching curve where a smaller value indicates that the subsequence is similar to the query sequence, the best match being the minimum of the matching curve. For this purpose, it is typical to apply distance measures that can deal with tempo differences, such as dynamic time warping (DTW), or the Smith-Waterman algorithm (Grosche, Müller, and Serrà 2012).

To speed up these exhaustive matching procedures, the authors discuss the need for methods that allow efficient detection of near neighbors rather than exact matches. Through vector quantization it is possible to obtain a codebook of a finite set of characteristic chroma vectors which can be used with an inverted file indexing approach. This would allow the classification of chroma vectors and to index them according to the assigned codebook vector. However the performance depends greatly on the codebook and this approach is only applicable for medium sized databases. Another, more recent approach is locality sensitive hashing (LSH) (Grosche, Müller, and Serrà 2012). Instead of considering long feature sequences, the audio content is split up into small overlapping short chroma feature subsequences. They are indexed using locality sensitive hashing which allows scaling this approach to larger datasets.

In summary, the authors found mid-specific audio matching using a combination of highly robust chroma features and sequence-based similarity measures result in a good retrieval quality. However, the low specificity of this task makes indexing much harder than in the case of audio identification (Grosche, Müller, and Serrà 2012).

2.2.2.4 Summary

Histograms can be both discriminative in large data sets and robust to common signal distortions while having high retrieval accuracy, and also benefits from established research in the text domain.

The main disadvantage is that it involves a vector quantization implying discretization and it ignores the relationships among words, which can be very important in audio representation but with the use of LOWBOW or codebooks this disadvantage can be overcome.

2.2.3 Music Retrieval using Topic Models

Topic Models are a suite of algorithms capable of uncovering the hidden thematic structure of large and unstructured collections of documents. The structure uncovered by topic models

can be used to explore an otherwise unorganized collection. For example, rather than finding documents through keyword search alone, we might first find the theme that we are interested in, and then examine the documents related to that theme. Topic Modeling algorithms discover not only the themes but also how those themes are connected to each other, and how they change over time. Topic models have been adapted to many kinds of data from its origin in text, to image and audio domain (Blei 2012).

The simplest and most intuitive topic model is the Latent Dirichlet Allocation (LDA). LDA represents documents as mixtures of topics. Each topic has probabilities of generating various words intuitively related to the topic. LDA makes three assumptions. One is the "bag of words" assumption, that the order of the words in the document does not matter. Another is the order of documents does not matter. The third assumption is that the number of topics is assumed to be known and fixed. With this assumptions LDA is still a powerful tool for discovering and exploring the hidden thematic structure. However by relaxing the assumptions of LDA it can easily be used in more complicated models.

As previously mentioned, topic models have been applied in the audio domain, specifically for audio information retrieval purposes (Hoffman, Blei, and Cook 2008; Hu 2009; Hu, Liu, Jiang, and Yang 2014; Ren, Dunson, and Carin 2008; Kim, Narayanan, and Sundaram 2009). They have been shown to be useful in discovering hidden topics for audio similarity retrieval surpassing other known approaches, such as Histograms and GMMs and have already been applied in a problem of query-by-audio as can be seen in the related work described in this section.

2.2.3.1 Hoffman et al. (2008)

The Hierarchical Dirichlet Process (HDP) was used by the authors to discover latent structure in audio, specifically timbral similarity (Hoffman, Blei, and Cook 2008). The HDP is a nonparametric Bayesian approach to clustering group data, it uses a Dirichlet process for each group of data. The HDP mixture model is a generalization of Latent Dirichlet Allocation, where the number of topics can be unbounded and learnt from data.

The way the authors chose to model songs with an HDP is through Dirichlet Process Mixture Models (DPMM). HDP assumes the existence of a countably infinite set of mixture components, overcoming the issue of GMM that assumes the existence of K mixture components. As said before, HDP is a model of grouped data being more appropriate than GMM for modeling collections of MFCCs, where each song is represented as a distribution over latent components but the population of latent components is shared across songs. For a better understanding of DPMM and HDP, the authors explain the processes with two metaphors, the Chinese Restaurant Process (CRP) and Chinese Restaurant Franchise (CRF), respectively (Hoffman, Blei, and Cook 2008).

In the CRP, we imagine a Chinese restaurant with an infinite number of communal tables and a positive scalar hyperparameter α. The restaurant is initially empty. The first customer sits at the first table and orders a dish. The second customer enters and decides either to sit at the first table with probability $\frac{1}{1+\alpha}$ or a new table with probability $\frac{\alpha}{1+\alpha}$. When sitting at a new table the customer orders a new dish. This process continues for each new customer, with the tth customer choosing either to sit at a new table with probability $\frac{\alpha}{\alpha+t-1}$ or at the kth existing table with probability $\frac{n_k}{\alpha+t-1}$, where n_k is the number of other customers already sitting at table k.

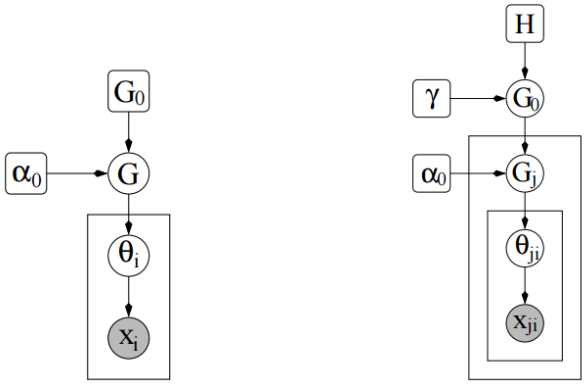

Figure 2.7: Hierarchical Dirichlet Process Graphical Representation.

The generative process underlying the HDP can be understood with the Chinese Restaurant Franchise (CRF). The CRF takes two hyperparameters α and γ. Each song j has its own CRP, and each feature vector $\gamma_{j,t}$ chooses a table from CRP(α). If it sits down at a new table, then it chooses a dish for that table from a global CRP (with hyperparameter γ) shared by all songs – that is, it either chooses a dish that is already being served at some number of other tables m with probability proportional to m, or it chooses a new dish with probability proportional to γ.

Although the CRP was defined as a sequential process, in fact the probability of a seating plan under the CRP is the same regardless of the order in which the customers sat down. We can think of the CRP as defining an implicit prior on infinite multinomial distributions over mixture components. Gibbs sampling is used to approximate the posterior distribution over the latent variables conditioned on observed data.

The songs were represented using the HDP, allowing the comparison of two songs in terms of the latent structure of their feature data unlike GMM-based algorithms that compare distributions over the low-level features. The feature extraction involved 13 MFCCs for each frame, approximately 23ms long, with a samping rate of 22050 Hz and no overlap. 1000 feature vectors were extracted from the middle of each song, and all models were trained on the same sets of feature vectors.

For the evaluation the authors compared the HDP against G1, GK (analogous to K-component GMM algorithm) and a Vector Quantization approach. The dataset consisted of 121 songs from seven genres. The authors used genre as a proxy for similarity. All songs labeled with the same genre are "similar", allowing the use of evaluation metrics from information retrieval. For each query song, each other song is given a rank based on its similarity to the query. R-precision, Average Precision, and the Area Under the ROC Curve were the metrics chosen.

The results show the amount of time required to compute distance matrices for GMMs was, enormous by comparison to the other models. The cost of computing the KL divergence for VQ and HDP-based models was quite lower than the cost of computing it between single Gaussians. HDP performed best out of all models, with VQ model being a very close second. The HDP approach generalizes well to new songs and does not suffer from hub similarity problem. As previously mentioned in the GMM section, hubs are an undesirable phenomenon of having bad matches selected as similar to a query.

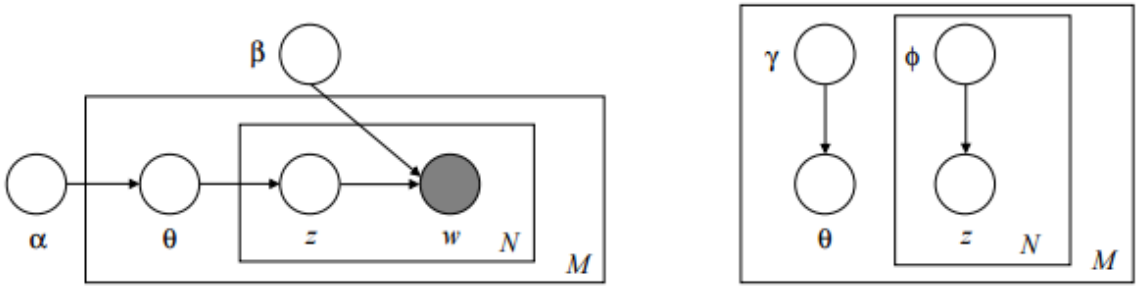

Figure 2.8: (Left) Graphical model representation of LDA. (Right) Graphical model representation of the variational distribution used to approximate the posterior in LDA.(Taken from (Blei, Ng, and Jordan 2003))

2.2.3.2 Hu (2009)

Diane Hu researched and applied LDA in the text domain and also in the image and music domain. In the music domain, the author discusses algorithms that extend LDA for automatic harmonic analysis and emphasizes approaches that go beyond LDA's standard bag-of-words representation (Hu 2009).

To better understand this model we first need to present the notation used

1. A word $w \in 1, ..., V$ is the most basic unit of discrete data. For cleaner notation, w is a V-dimensional unit-based vector. If w takes on the ith element in the vocabulary, then $w^i = 1$ and $w^j = 0$ for all $j \neq i$.

2. A document is a sequence of N words denoted by $w = (w_1, w_2, ..., w_N)$, where w_n is the nth word in the sequence.

3. A corpus is a collection of M documents denoted by $D = (w_1, w_2, ..., w_M)$.

4. A topic $z \in 1, ..., K$ is a probability distribution over the vocabulary of V words. Topics model particular groups of words that frequently occur together in documents, and thus can be interpreted as "subjects."

For each document indexed by $m \in 1, ..., M$ in a corpus the generative process is as follows:

1. Choose a K-dimensional topic weight vector θ_m from the distribution $p(\theta|\alpha) = Dirichlet(\alpha)$.

2. For each word indexed by $n \in 1, ..., N$ in a document:

   (a) Choose a topic $z_n \in 1, ..., K$ from the multinomial distribution $p(z_n = k|\theta_m) = \theta_m^k$.

   (b) Given the chosen topic z_n, draw a word w_n from the probability distribution $p(w_n = i|z_n = j, \beta) = \beta_{i\ j}$.

The parameter α is a K-dimensional parameter that stays constant over all of the documents within a corpus.

The Dirichlet distribution is given by:

$$p(\theta|\alpha) = \frac{\tau(\sum_i \alpha_i)}{\prod_i \tau(\alpha_i)} \prod_i \theta^{\alpha_i - 1} \tag{2.22}$$

The generative process given above defines a joint distribution for each document w_m. Assuming for now that parameters α and β are given to us, the joint distribution over the topic mixtures θ and the set of N topics z is:

$$p(\theta, z, w|\alpha, \beta) = p(\theta, \alpha) \prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n, \beta) \tag{2.23}$$

Now, the central task for using LDA is to determine the posterior distribution of the latent topic variables conditioned on the words that we observe in each document.

In the automatic harmonic analysis task, the author wanted to find the key of a musical piece. The main reason behind trying automatic key-finding is that a musical piece has a main key, however individual passages can exhibit complex variations. The author correlated as best as possible the notation according to the original LDA. Musical notes play the role of words, songs are documents and the topics are the musical keys. The chosen features were 12 pitch-class profiles. According to the LDA representation, the topics are expressed as distributions over the 12 pitch-classes.

In summary, the generative process begins by drawing a topic weight vector from a Dirichlet distribution, which will determine the keys present in the song. For each segment, a single key is picked from the topic weight vector. Finally, notes are repeatedly drawn from a probability distribution conditioned by the chosen key until all the notes in the segment have been generated. Results showed this algorithm achieved a 6 to 12% improvement over existing key-finding algorithms.

Regarding all the experiments, LDA proved to be a versatile, generative model. Because LDA ignores word order, the models in each domain, such as text and audio, will face the challenge of going beyond the bag-of-words representation and incorporate order information into their LDA framework. The most developed models that include ordering information are in the text domain and the author proposes that directly incorporating statistics of word transitions in the Bigram Topic Model can be very useful for harmonic analysis.

2.2.3.3 Hu et al. (2014)

Hu et al. combined the Latent Dirichlet Allocation with Gaussian modeling, introducing Gaussian-LDA which directly models each topic as a Gaussian distribution over audio features (Hu, Liu, Jiang, and Yang 2014). Gaussian-LDA is built on the same principles of a topic model. It shares the properties of standard LDA but in the last distribution instead of using a multinomial distribution over words it defines a Gaussian distribution for each topic over the audio feature. Therefore it does not need a vector quantization step like the standard LDA, avoiding discretization and also integrates the procedure of clustering.

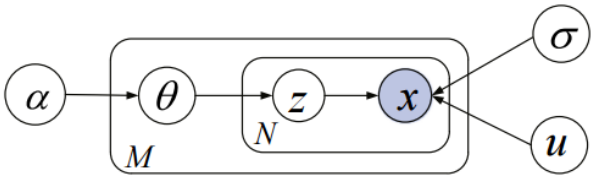

Figure 2.9: Gaussian LDA Graphical representation (Hu, Liu, Jiang, and Yang 2014).

For the feature extraction the authors normalized the sound files to 16kHz sampling and 16 bit per sample. Then they remove the silence frames with a log energy threshold and only afterwards the short-term features are extracted. The authors computed Mel-Frequency Cepstral Coefficients plus normalized energy and the first derivatives, considering spectral centroid, spectral flatness, spectral rolloff and zero crossing. The second feature set consisted of concatenating 26 dimensional MFCC with 8 dimensional perceptual features.

The authors compute distances between vectors since each audio clip is represented by a vector. The chosen similarity measure is the cosine distance which measures similarity between query audio and training audio clips.

In the evaluation all training audio clips are ranked according to distances to query audio, and the closest k candidates will be retrieved. The evaluation was done with 1214 audio documents, each associated with a category, randomly selecting 100 documents as query and the rest of the dataset as training set. The documents are considered similar if their category is the same as the query. For each query the precision and recall rates are calculated.

The results were compared with standard LDA, a histogram method and GMM as baselines. The Gaussian-LDA outperformed the standard LDA topic model regardless of the number of latent topics (Hu, Liu, Jiang, and Yang 2014). The authors found experimentally the MFCC feature set provided a higher performance with a 34 dimensional feature instead of a 26 dimensional feature set. Overall, the experimental results showed Gaussian-LDA produces higher precision and recall rate than the others. The histogram method achieved better results than GMM and the standard LDA enhanced the performance of histogram when less than 10 most similar samples are retrieved. The authors argue that the discretization of features in standard LDA affects the performance so Gaussian-LDA has better performance due to avoiding discretization. They also argue that both the histogram method and topic models way of exploring similarity are beneficial and there is a correlation between them for audio analysis. However, standard LDA just combines two measurements doing nothing about weighting them. They suggest this is the essential reason why Gaussian-LDA shows a better and more stable performance. They conclude some limitations from the standard LDA have been inherited, such as the sequence information among short-term features being neglected.

2.2.3.4 Summary

Topic models are Unsupervised techniques where each song is a mixture of topics. These models can be easily extended, in particular LDA, which is highly modular (Hu 2009). The number of topics can be unbounded and learnt from data as seen in HDP (Hoffman, Blei, and Cook 2008).

The main disadvantages are that topics are soft-clusters without any objective metric to tell

if the choice of hyperparameters is the best and they comprehend a vector quantization step for discretization. This has been overcome through Gaussian-LDA (Hu, Liu, Jiang, and Yang 2014) but the sequence information among short-term features is still neglected.

2.3 Discussion

In overall terms, we can distinguish between the presented models in two main properties, if they discretize information and if they are able to maintain the sequential information. The properties of each model are represented in the following table:

Model	Discretized	Sequential Information
GMM	No	Yes
Histogram	Yes	No
LOWBOW	Yes	Yes
HDP	Yes	No
LDA	Yes	No
Gaussian-LDA	No	Yes

Table 2.2: Model Discretization and Sequential Information Properties.

As can be seen from table 2.2, GMMs are capable of retaining the sequential information and do not need to discretize so there is no loss, however this model suffers from hubs (Aucouturier and Pachet 2002b), (Pachet and Aucouturier 2004), songs which are irrelevantly close to all other songs and there is a glass-ceiling around 70% precision. Therefore in a query-by-example scenario this model is not the best.

Histograms (Riley, Heinen, and Ghosh 2008; Grosche, Müller, and Serrà 2012) require a discretization step where there is a small loss of information and the sequential information is disregarded but this model presents good results being capable of competing with the GMMs. The histograms are suitable for a query-by-example scenario if we do not need to take into account sequential information.

The LOWBOW framework (Lebanon, Mao, and Dillon 2007) extends the histogram model by retaining the sequential information among the LOWBOW curves. This model still has not been applied to audio, but the results in the text domain show great prospects and this framework is suitable in a query-by-example scenario where we need to take into account sequential information.

The HDP (Hoffman, Blei, and Cook 2008) and LDA (Hu 2009) topic models are both suitable for query-by-example scenarios. The main difference is HDP is a generalization of LDA with the number of topics unbounded and learnt from data. They both inherit the discretization step from the histograms and present good results, higher than the histogram model.

The Gaussian-LDA topic model (Hu, Liu, Jiang, and Yang 2014) extends the LDA by directly modeling each topic as a Gaussian distribution bypassing the vector quantization step avoiding discretization. This model is also suitable for a query-by-example scenario.

The authors of the Gaussian-LDA (Hu, Liu, Jiang, and Yang 2014), argued that both the

histogram method and topic models way of exploring similarity are beneficial and that there is a correlation between them for audio analysis.

Taking into account these various models, we consider the most suitable models for our query-by-example to be LOWBOW and Gaussian-LDA due to their capability of maintaining sequential information. We consider that for our particular audio matching scenario, sequential information is of extreme importance. GMM was not considered due to its hubs problem, which would affect our query-by-example.

2.4 Summary

In this chapter we discussed the current state-of-the-art for the audio matching task in order to get a better understanding of our task and see what can still be done to improve it. We presented the typical 3-stage architecture for query-by-example systems where we also showed some of the most used features and algorithms in audio matching. We then presented three major audio modeling approaches: Music retrieval using Gaussian Mixture Model, Histograms, and Topic models. We finish this section with a comparison table of all the studied systems.

3 Feature Case Study

Various features with different characteristics suitable to harmonic progression were previously presented in section 2.1.1. In this chapter we study the feature extraction process and, from there on, define possible changes and experiment for our task.

3.1 Feature Study

We will now begin studying the features of choice. We decided for each feature type to analyze a small set of songs,in order to try to get a better understanding of the features and their flaws for audio matching.

We chose three songs for this case study. The original ”River Flows in You” and two cover songs henceforth ”Coversong 1” and ”Coversong 6”. ”Coversong 6”, is the one most similar in terms of harmony and structure to ”River Flows in You”, ”Coversong 1” is far less similar.

3.1.1 Chroma Variations

The Chroma variations described and available in the Chroma Toolbox possess the desired characteristics for our audio matching task (Müller and Ewert 2011a; Müller, Kurth, and Clausen 2005; Müller, Ewert, and Kreuzer 2009). Therefore we studied the Chroma Toolbox and used the default setup $w25$ $d5$ for CENS and $n = 55$ for CRP, which, according to the authors, was demonstrated to have good results (Müller and Ewert 2011a; Müller, Kurth, and Clausen 2005; Müller, Ewert, and Kreuzer 2009). Experiments with the described setup are shown in Figure 3.1 to Figure 3.6.

Taking into account that the Chromagrams 3.5 and 3.1 are quite similar and 3.3 has big differences which means CENS is a valid choice to detect audio similarity. CRP Chromagrams showed similar results however we are more interested in CENS because we want to have audio-word histograms representations that are robust to distortion allowing for an audio segment and a distorted version to be very similar (Riley, Heinen, and Ghosh 2008; Müller, Kurth, and Clausen 2005).

3.1.1.1 Vector Quantization

To feed these extracted features to the Query Engine we must first process them with a vector quantization stage. In this stage we used the k-Means clustering algorithm through the weka framework with cluster number $k = 500$. Riley et al. (2008), experimented with several clustering algorithms besides k-Means and found that most algorithms involved greater computational complexity without improving the VQ and resulting song matching. The authors

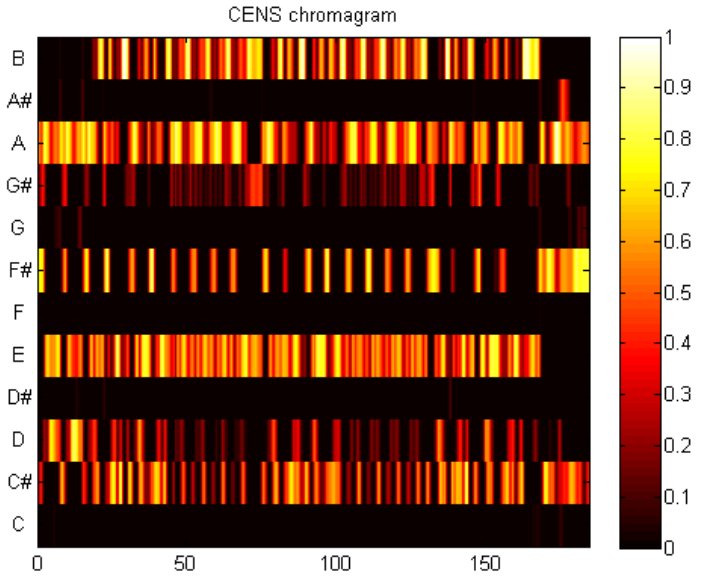

Figure 3.1: River Flows in You - CENS Chromagram

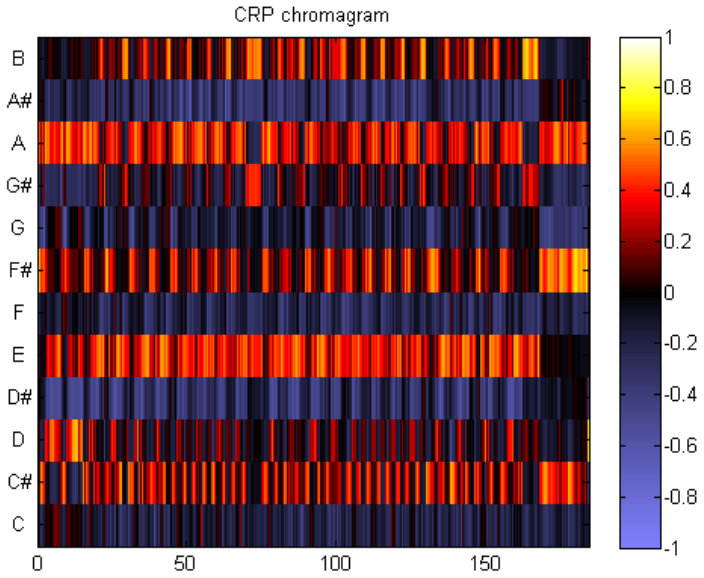

Figure 3.2: River Flows in You - CRP Chromagram

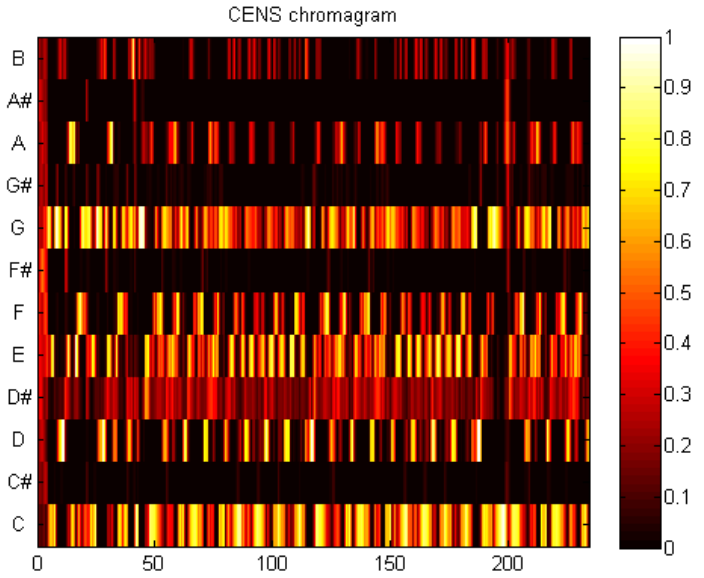

Figure 3.3: Cover song 1 - CENS Chromagram

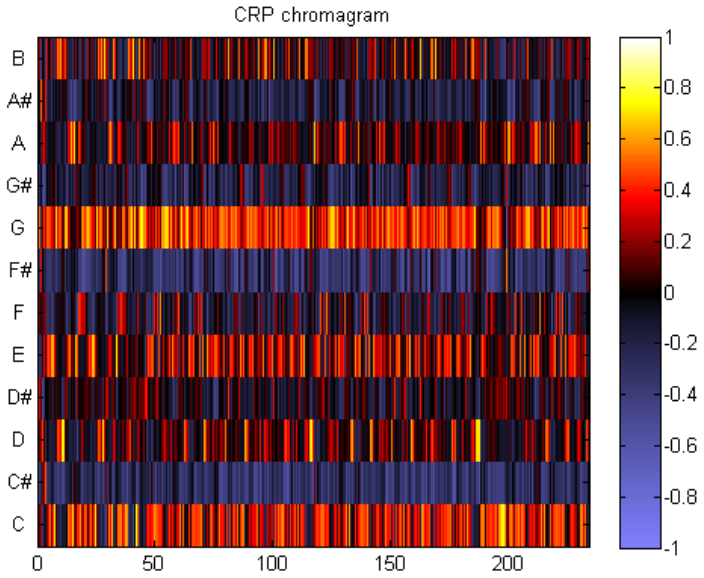

Figure 3.4: Cover song 1 - CRP Chromagram

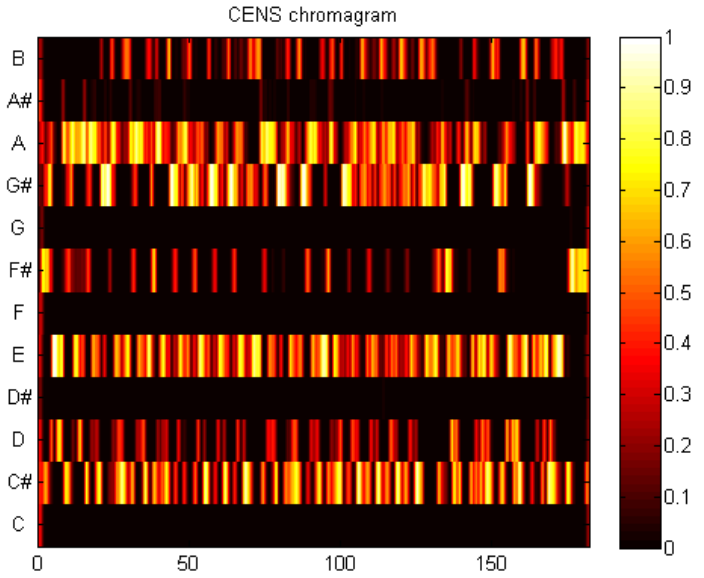

Figure 3.5: Cover song 6 - CENS Chromagram

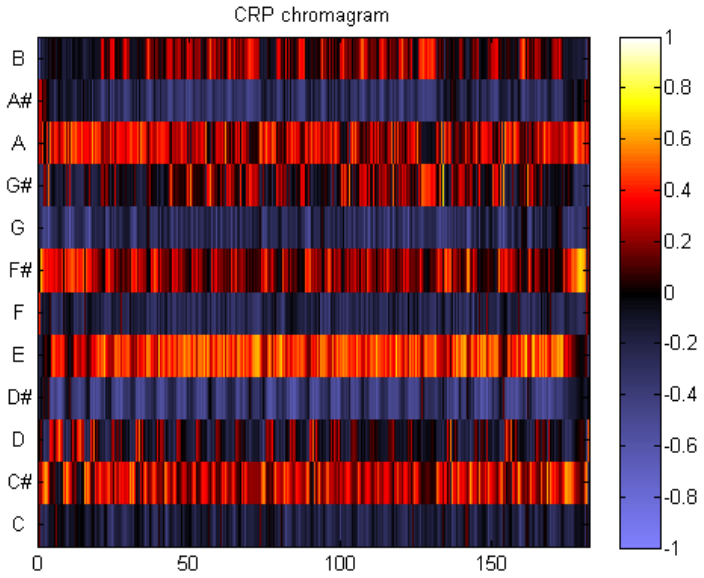

Figure 3.6: Cover song 6 - CRP Chromagram

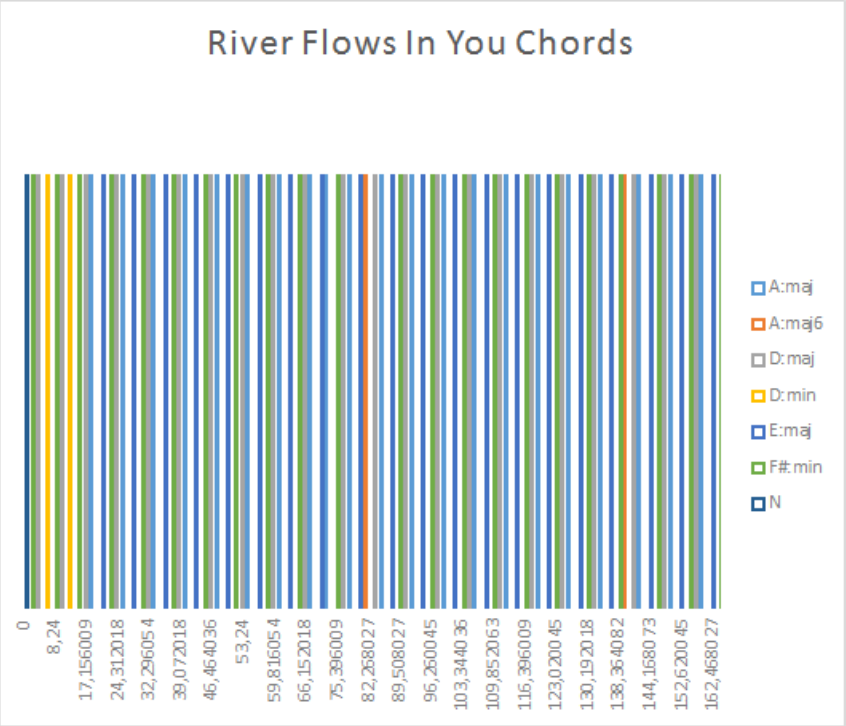

Figure 3.7: River Flows in You - Chords Plot

also determined experimentally that $k = 500$ performs successfully (Riley, Heinen, and Ghosh 2008).

The results are k feature descriptors corresponding to the centroids of the clustering, these feature descriptors act as audio words forming a vocabulary with length k and each feature vector in a song becomes associated with its closest descriptor. This means for each song there will be an output audiowords file where the feature vector of instant t is replaced with the closest audioword.

By representing individual Chroma vectors by the cluster center to which they belong, we make equivalent in the histogram representation any segments that were musically very similar but may have had slightly different Chroma vectors.

The audio Query also requires a quantization step where for each feature vector i in the Query we quantize by finding the best cosine similarity between the feature vector i and all the audiowords, which results in an output as above of an audiowords file.

3.1.2 Chords

The Harmony Progression Analyzer (HPA) (Ni, McVicar, Santos-Rodriguez, and De Bie 2012) also possesses the desired characteristics for our audio matching task. Therefore we studied HPA full chord extraction and used the default setup of 11025 Hz sample rate, for the bass chromagram 55 minimum frequency and maximum frequency of 207, and finally for the treble chromagram 220 minimum frequency and maximum frequency of 1661. Experiments with the described setup are shown in Figure 3.7 to Figure 3.9.

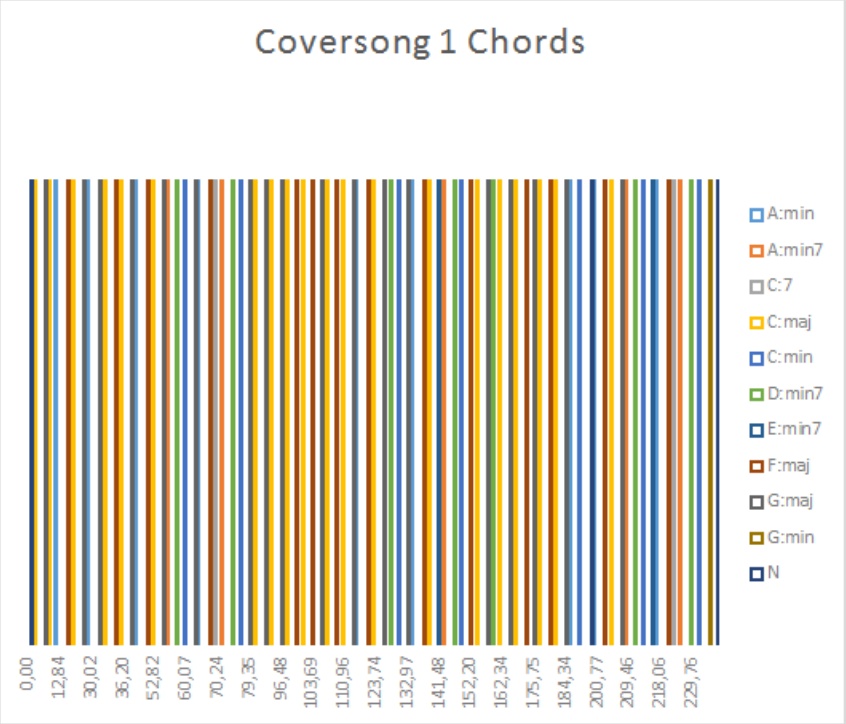

Figure 3.8: Cover song 1 - Chords Plot

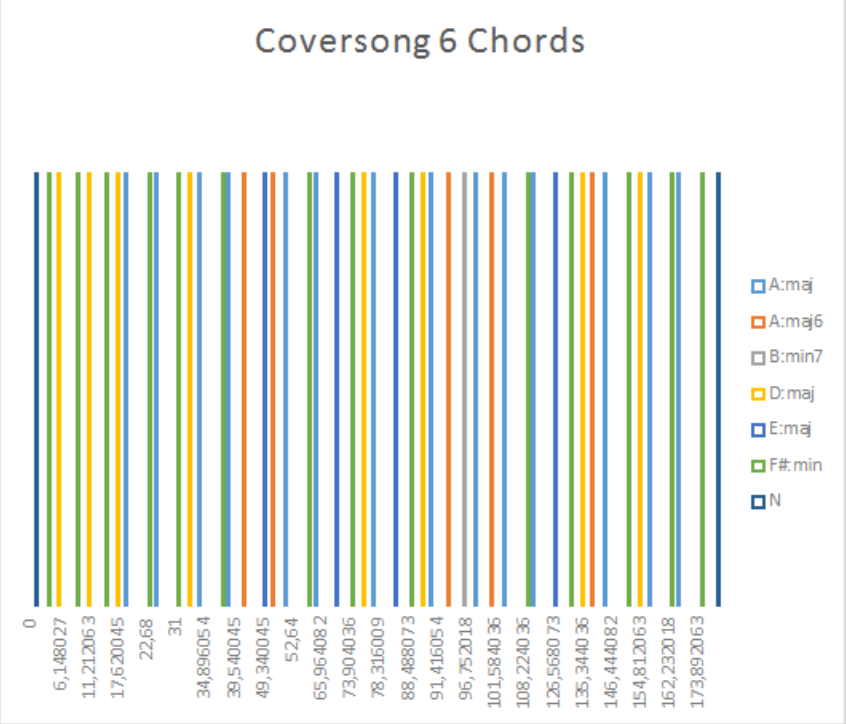

Figure 3.9: Cover song 6 - Chords Plot

Taking into account that the Chords Plots 3.9 and 3.7 are similar, although 3.9 has the chord sequence with a slower tempo, and 3.3 has larger differences, it is possible to conclude that HPA is a valid choice to detect harmonic similarity for our task.

3.2 Discussion

In table 3.1 we compare the main characteristics that distinguish CENS and Chords.

Features	Representation	Sparsity	Phrasing Similarity
CENS	Low-level	High	Medium
Chords	High-level	Low	High

Table 3.1: Feature Characteristics Comparison.

The Vector Quantization required for the CENS features leads to many similarity mismatches in phrasing since musically different vectors are grouped in the same cluster. Thinking of it in terms of chords it means, for example, a C7 chord was considered the same as CMaj. Therefore the usage of CENS or CRP is not a valid option for our system, as we are interested in possessing the capability to distinguish between chords.

The chord extraction through the HPA is quite robust, although some chords might be wrongly classified. This represents just a small influence in our phrasing similarity as opposed to CENS or CRP. Additionally, using chords provides a high-level representation closely correlated to the harmonic progression.

3.3 Summary

In this chapter we studied the features presented in section 2.1.1 in order to get a better understanding of their suitability towards our task and grasp their advantages and disadvantages. We present our rationale and finish this section with a comparison table of the studied features.

4 Query-by-Example Construction

The purpose of this chapter is to study some of the previously obtained results in order to construct an audio matching query-by-example system. We took into account the feature study performed and from there we define the stages of our query-by-example system. The full architecture of our query-by-example system is detailed in 4.1. In order to validate and verify the improvement of our system we define a suitable baseline for our task. Then we evaluate our query-by-example system and compare it against the bag-of-words baseline while showing the results and posterior conclusions we took. Afterwards we show the application of our query-by-example system to a cover song detection task.

4.1 Baseline

A baseline, generically speaking, is a measurement of some sort used as a basis for comparison. If we want to validate, study and improve previous work, one possible way of doing this is to have a basis for comparison, a baseline. Regarding this thesis, we decided to use a baseline for numerous reasons. The first reason is we are adapting an already existing technique in text to the audio domain. Creating a complex system from scratch would prove very troublesome due to its complexity. The second reason relates to the study and experimentation of the system: this means we wanted a fully functional system in order to test, study and possibly improve it. Another important aspect regarding baseline study is the ability to reproduce the baseline results.

4.1.1 Bag-of-Words

The audio matching task poses a concern in finding a good baseline because in the literature we could not find any available dataset for audio matching purposes to be used. The authors tend to create their own datasets.

We decided to replicate an audio Bag-of-words approach similar to the one proposed by (Riley, Heinen, and Ghosh 2008). The motive to replicate the audio Bag-of-words approach is that we use the lowbow framework which extends bag-of-words by locally weighting the terms with temporal smoothing. Given their similarities and the possibility to also use lowbow as a bag-of-words, by altering a parameter, we can efficiently use bag-of-words in lowbow as a baseline and assert our findings of using a locally weighted bag-of-words instead of a typical bag-of-words.

Due to the unavailability of audio matching datasets we decided to use our own datasets and experiment audio matching in two contexts, with a short-query dataset and covers dataset.

4.2 Feature Extraction

Taking into account the results from our feature study in Chapter 3, we used Harmony Progression Analyzer for our feature extraction since we want robustness and high level correlation with the harmonic progression.

4.3 Query Engine

Following our discussion in section 2.3 of the models used in the related work and the different properties they possessed, we chose the lowbow framework as our query engine because of its novelty in the audio domain and also being capable of retaining sequential information which makes it suitable for our query by example system since it will allow the capture of trends in tempo, articulation and phrasing.

We adopted the lowbow framework by implementing it in C++. This framework extends the bag-of-words by performing temporal smoothing where for each time instant the words are locally weighted, effectively becoming a locally weighted bag-of-words and capturing sequential information, such as local trends whereas the bag-of-words is not capable of doing since it is orderless.

Our audio lowbow framework has the following definitions:

Definition 1 *A song* x *of length* N *is a function* $x : 1, ..., N \times V \rightarrow [0, 1]$ *such that*

$\sum_{j \in V} x(i, j) = 1 \; \forall i \in 1, ..., N.$

The set of songs (of all lengths) is denoted by X.

For a song $x \in X$ the value $x(i, j)$ represents the weight of the audio word $j \in V$ at location i. Since the weights sum to one at any location we can consider Definition 1 as providing a local audio word histogram or distribution associated with each song position. The standard way to represent an audio word sequence as a song in X is to have each location host the appropriate single audio word with constant weight, which corresponds to the δ_c representation defined below with $c = 0$.

Definition 2 *the standard representation* $\delta_c(y) \in X$, *where* $c \geq 0$, *of an audio word sequence* $y = \langle y1, ..., y_N \rangle$

$$\delta_c(y)(i, j) = \begin{cases} \frac{c}{1+c|V|} & y_i \neq j \\ \frac{1+c}{1+c|V|} & y_i = j \end{cases}. \tag{4.1}$$

Equation 4.1 is consistent with Definition 1 since $\sum_{j \in V} \delta_c(y)(i, j) = \frac{1+c|V|}{1+c|V|} = 1$. The parameter c in the above definition injects categorical smoothing to avoid zero counts in the δ_c, representation.

Definition 1 lets several audio words occupy the same location by smoothing the influence of audio words y_j across different song positions. Doing so is central in converting the discrete-time standard representation to a continuous representation that is much more convenient for modeling and analysis.

Definition 1 is problematic since according to it, two songs of different lengths are considered as fundamentally different objects. To allow a unified treatment and comparison of songs of arbitrary lengths we map the set $1, ..., N$ to a continuous canonical interval, which we chose to be $[0, 1]$.

Definition 3 *A length-normalized song x is a function $x : [0, 1] \times V \to [0, 1]$ such that*

$\sum_{j \in V} x(t, j) = 1, \forall t \in [0, 1]$.

The set of length-normalized songs is denoted X'

A simple way of converting a song $x \in X$ to a length-normalized song $x' \in X'$ is expressed by the length-normalization function defined below.

Definition 4 *The length-normalization of a song $x \in X$ to a length-normalized song $x' \in X'$ is the mapping*

$\varphi : X \to X' \; \varphi(x)(t, j) = x(\lceil tN \rceil, j)$

where $\lceil r \rceil$ is the smallest integer greater than or equal to r.

The length-normalization process abstracts away from the actual song length and focuses on the sequential variations within the song relative to its length. In other words, we treat two songs with similar sequential contents but different lengths in a similar fashion. For example the two songs $\langle y1, y2, ..., y_N \rangle$ and $\langle y1, y1, y2, y2..., y_N, y_N \rangle$ would be mapped to the same length-normalized representation.

We formally define bag-of-audio-words as the integral of length-normalized songs with respect to time. This definition is equivalent to the popular definition of the traditional bag-of-words.

Definition 5 *The bag-of-audio-words or boaw representation of a song y is $\rho(\varphi(\delta_c(y)))$ defined by*

$$\rho : X' \to \mathbb{P}_{V-1} \text{where} [\rho(x)]_j = \int_0^1 x(t, j) dt, \tag{4.2}$$

and $[\cdot]_j$ denotes the j-th component of a vector.

Above, $\mathbb{P}_{V-1}$ stands for the multinomial simplex.

Definition 6 *The locally weighted bag-of-audio-words or lowbow representation of the audio word sequence y is $\gamma(y) = \gamma_\mu \in [0, 1]$ where $\gamma_\mu(y) \in \mathbb{P}_{V-1}$ is the local audio word histogram at μ defined by*

$$[\gamma_\mu(y)]_j = \int_0^1 \varphi(\delta_c(y))(t, j) K_{\mu,\sigma}(t) dt. \tag{4.3}$$

Equation 4.3 indeed associates a song location with a local histogram or a point in the simplex $\mathbb{P}_{V-1}$ since

$\sum_{j\in V}[\gamma_\mu(y)]_j = \sum_{j\in V}\int_0^1 \varphi(\delta_c(y))(t,j)K_{\mu,\sigma}(t)dt = \int_0^1 K_{\mu,\sigma}(t)\sum_{j\in V}\varphi(\delta_c(y))(t,j)dt = \int_0^1 K_{\mu,\sigma}(t)\cdot 1dt = 1.$

The Simplex of the lowbow framework is initialized with the obtained vocabulary from the HPA chords. Query and database songs received are initialized as smooth curves through the corresponding audiowords file receiving categorical and temporal smoothing.

The temporal smoothing in our framework is done through the Gaussian probability density function (pdf) restricted to [0,1] and renormalized:

$$K_{\mu,\sigma}(x) = \begin{cases} \dfrac{N(x;\mu,\sigma)}{\phi((1-\mu)/\sigma)-\phi(-\mu/\sigma)} & x\in[0,1] \\ 0 & x\notin[0,1] \end{cases} \tag{4.4}$$

where $N(x;\mu,\sigma)$ is the Gaussian pdf with mean μ and variance σ^2 and ϕ is the cumulative distribution function (cdf) of $N(x;0,1)$.

4.4 Audio Matching

The primary reason behind choosing our audio matching algorithm is in regards to what similarity we want to tolerate. As stated previously we are interested in tempo, phrasing and articulation so chord length deviations and insertions/removals are considered. Secondly, it has to be easily adaptable to our subsequence matching algorithm. Thirdly, the time constraint, although it is not the focus of this thesis, we want to obtain an efficient query-by-example system.

Comparing all of the state-of-the-art algorithms presented in section 2.1.3 the constrained DTW (cDTW) is the best for our system. It captures the similarities in tempo, phrasing and articulation. It also is easily adaptable to subsequence matching, such is the case of SPRING. Additionally it has a better performance than the other algorithms.

We adapted cDTW for our task by using sink states as explained in the SPRING algorithm. We allow the match of a query to start at any point between the start and end of the music being compared while also allowing the match of a query to end at any point between the start and end of the music being compared. We also set a threshold of distance equal to 1 in order to exclude resulting sequences of higher distance which deviate from our query in order to be in accordance with our task. With these changes we effectively solve the query to subsequence problem of our task. We are able to compare between queries and full songs using cDTW to capture the similarities in tempo, phrasing and articulation.

4.5 Experimental Setup

As mentioned before in section 3.1.2, we applied the same settings specified for our feature extraction with Harmony Progression Analyzer, respectively 11025Hz sample rate, 55 minimum frequency and maximum of 207 for the bass chromagram and 220 minimum frequency

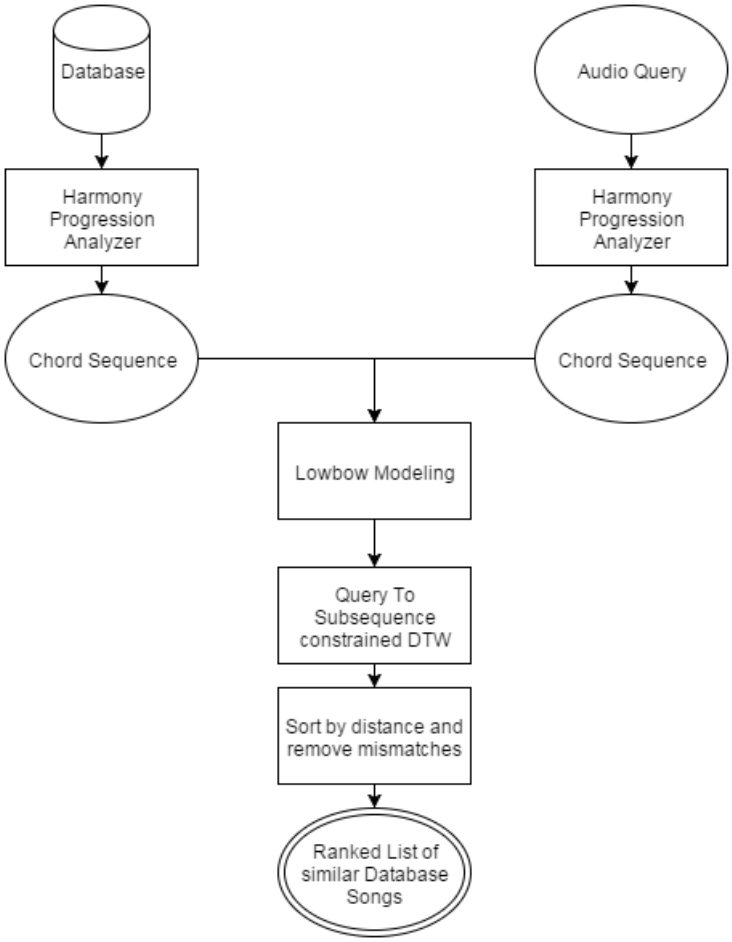

Figure 4.1: Architecture of our Query-by-Example system.

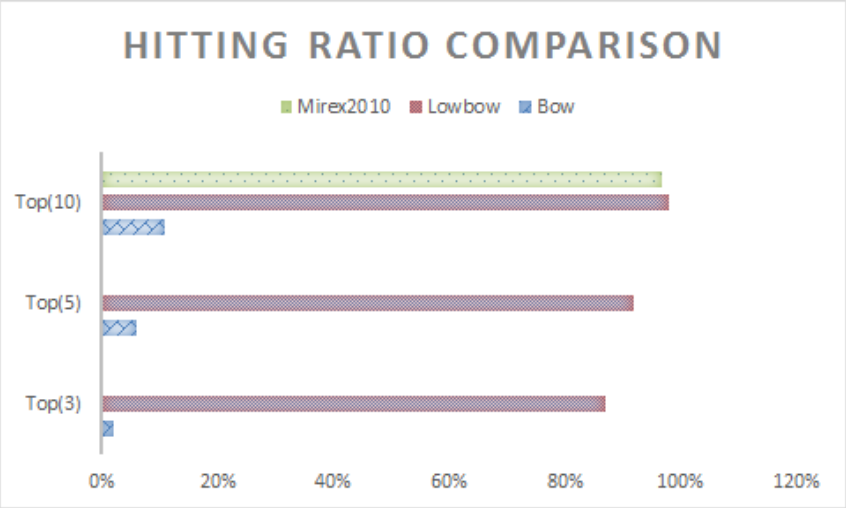

Figure 4.2: Hitting ratio comparison between Lowbow and Bow.

and maximum of 1661 for the treble chromagram. For our modeling using locally weighted bag-of-words we chose for our sigma $\sigma = e^{-3}$, a smoothing coefficient of e^{-2} and the sampling was done with a 1.0 ratio (Lebanon, Mao, and Dillon 2007).

4.6 Short Query Dataset

The dataset is comprised of 700 noise songs and 25 popular guitar songs from various genres, such as blues, rock, metal, etc. For each song we extracted four short length queries between 10 to 15 seconds resulting in 100 queries.

The experiment consisted of running each query against our query-by-example system and retrieving the respective song from which we extracted the short query. The experimental setup was previously described in section 4.5.

The main purpose of this experiment was to validate our query-by-example system while also demonstrating that lowbow retains sequential information being leveraged in our system.

The metrics used to assess our performance were:

- Hitting ratio $Top(N) = \frac{Hit(N)}{Total}$.

- Mean Reciprocal Rank $MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{Rank(i)}$.

These metrics are also used by the MIREX community for query-by-humming systems considered as standard metrics for query system performance evaluation.

The results are shown in detail in Figure 4.2.

4.6.1 Query Results

As we can see in Figure 4.2, our query-by-example system using lowbow is capable of achieving hitting ratios similar to MIREX's query-by-humming state of the art, a MIREX 2010

submission.

Assessing the results from our task point of view, of retrieving songs where we can play the submitted query. We found that the songs, between the first rank and our target song from which we extracted the query, had higher similarity due to lowbow's smoothing. By removing the smoothing process we observed all these songs had the same DTW distance, respectively 0.

The smoothed curves affect the local points in a way that the musical trends near that point are also retained. For example, a song which repeats the query sequence often, means the trend at the obtained subsequence is highly correlated with the original query whereas a song that has the original query but then shifts to a completely different theme does not.

Model	Mean Reciprocal Rank
Bow	0,03
Lowbow	0,756
Mirex 2010	0,947

Table 4.1: Mean Reciprocal Rank comparison.

In Table 4.1 the MIREX 2010 submission of the query-by-humming state of the art outperforms our system, although in the top 10 hitting ratio we have higher accuracy. This result directly translates to the reasoning that for short queries with popular chord sequences it is hard to find at rank 1 the song you are expecting since there are plenty of songs with that popular chord subsequence. In section 5.1 we talk about a possible solution to this problem.

4.6.2 Baseline Comparison

In terms of our bow baseline comparison, for our matching problem, bow is not capable of finding target songs for our task, as it only has a Mean Reciprocal Rank of 3%.

The lowbow greatly outperforms the bag-of-words baseline. This result can be explained by the sequential information being captured in lowbow which provides excellent results in our query to subsequence matching problem.

4.7 Cover Song Dataset

The cover song dataset is comprised of 700 noise songs, 30 original songs, each one with 10 respective cover songs.

The experiment consisted of running each original song against our query-by-example system and retrieving the respective cover songs. The experimental setup was previously described in section 4.5, however in this experiment we did not use the constraints, therefore allowing cDTW to start and end at any point since we are using full songs.

The results are shown in detail in Figures 4.3, 4.4, 4.5 and a Table 4.2 of the classification tendencies of bow and lowbow.

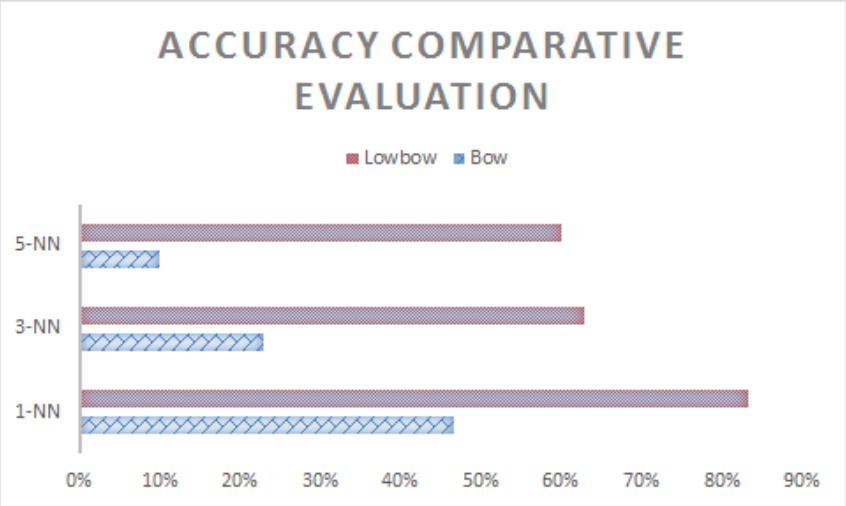

Figure 4.3: Cover Songs Accuracy Comparative Evaluation.

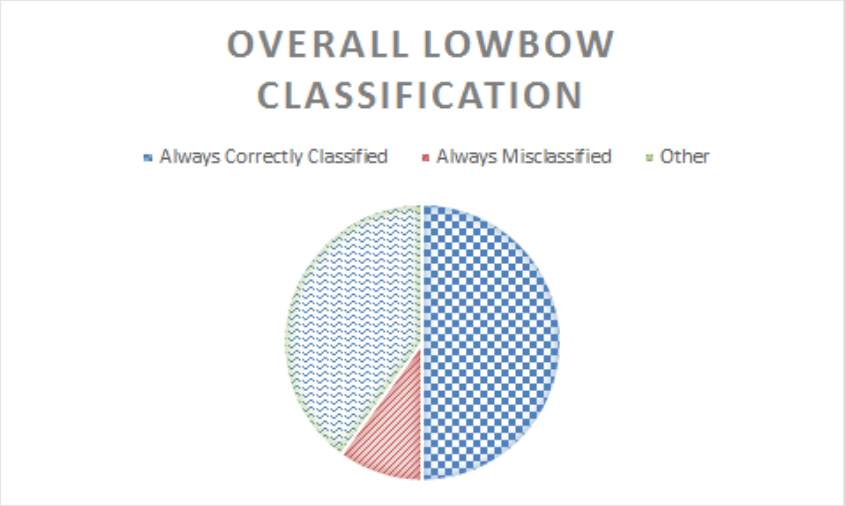

Figure 4.4: Overall Classification Tendencies of Lowbow.

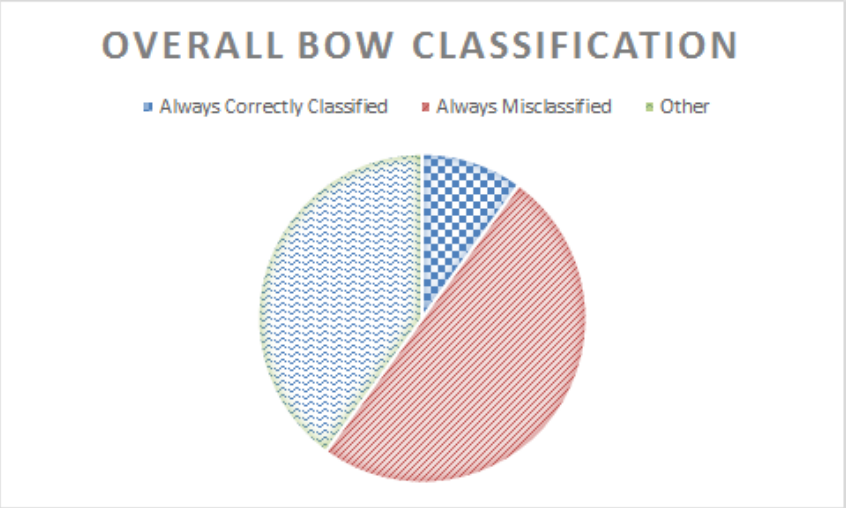

Figure 4.5: Overall Classification Tendencies of Bow

Model	Always Correctly Classified	Always Misclassified	Others
Bow	10%	40%	50%
Lowbow	50%	10%	40%

Table 4.2: Classification Tendencies Lowbow vs Bow

4.7.1 Cover Song Results

As we can see in Figure 4.3, our query-by-example system using lowbow is capable of detecting cover songs to a certain degree. In accordance with our system characteristics, Figure 4.4 shows that our system is capable of correctly identifying cover songs which are based on the original song harmonic sequence, whereas cover songs based on melody or rhythm have more harmonic differences.

4.7.2 Baseline Comparison

Assessing the results against our bow baseline, we see that for cover song detection bow is a valid choice for cover songs such as studio to live versions without large harmonic differences as it had already been applied to in related work in chapter 2.

Lowbow greatly outperforms the baseline as it is capable of extending the range of identified cover songs. This result is clearly noticed by looking at Table 4.2. These results can be explained again by the sequential information being captured which allows us to tolerate deviances in the harmonic sequence whereas the bag-of-words only takes into account the global harmony without any order.

4.8 Summary

In this chapter we presented and defined our baseline as being the traditional bag-of-words approach. We also defined our 3-stage architecture query-by-example system where we also showed adjustments made towards our task, such as in audio matching the query to subsequence adjustment. For the feature extraction the HPA is used. The lowbow framework will serve as our query engine. Finally the audio matching will be done with an adapted version of cDTW.

We showed our query-by-example system achieves good results against our baseline and is suitable for our task.

The sequential information captured in lowbow proves to be a major factor for the success of our proposed task since it allows both the capability of identifying similar harmonic sequences and provides a higher differentiation between songs that have the same harmonic sequence with different trends before and/or after the sequence.

We also achieved good results in the cover song task. We can conclude that lowbow is suitable for identifying cover songs relying on harmony.

5 Conclusions and Future Work

We started this work with a goal to create a robust and adaptable query by example system for audio matching. We decided to focus on the lowbow framework and on audio bag-of-words in order to obtain a baseline to enhance our knowledge of the lowbow framework.

5.1 Conclusions

After an extensive study we conclude that our query-by-example system successfully addresses our task of finding thematically similar songs allowing the user to play the query to be submitted and that our system can also be used for certain cover songs detection.

We believe that the future of audio matching will pass by a mixture of sequential information with some symbolic data or supervised learning. The reasoning behind our belief is that audio similarity requires a deeper connection between the sequential information and the musical context in which the sequence occurs. Knowing if we are in the presence of a chorus or a verse will help in improving audio matching and enhance the knowledge obtained from sequential information. With the added information we believe audio matching can grow to something bigger and become capable of relating an original with its cover songs with high precision.

5.1.1 Goals Discussion

For our particular problem of audio matching in a query-by-example scenario two major objectives must be fulfilled: firstly, a way to evaluate the problem; and secondly, an algorithm to solve it.

Regarding this thesis, we had some difficulties in solving the first problem since audio matching has some inherent subjectivity and there were no available audio matching datasets. Our solution was to validate short queries by creating our own audio matching dataset and also by applying our query-by-example system to the closest audio similarity tasks, cover song detection. The second problem was also one of this thesis goals: the creation of an audio matching query by example system. We solved this problem by using the lowbow framework and studying related work to define an audio bag-of-words baseline from it.

5.2 Future Work

We believe that changing our lowbow framework to capture local contexts, such as song chorus, verses, among others, to be of high value towards improving our system. With these

labels users can explicitly tag queries to be of a particular context, e.g. chorus. Afterwards our system would compare that particular query only to chorus sections.

Bibliography

Aucouturier, J.-J. and F. Pachet (2002a). Finding songs that sound the same. In *Proc. of IEEE Benelux Workshop on Model based Processing and Coding of Audio*, pp. 1–8.

Aucouturier, J.-J. and F. Pachet (2002b). Music similarity measures: What's the use? In *ISMIR*.

Aucouturier, J.-J. and F. Pachet (2008). A scale-free distribution of false positives for a large class of audio similarity measures. *Pattern Recognition 41*(1), 272–284.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM 55*(4), 77–84.

Blei, D. M., A. Y. Ng, and M. I. Jordan (2003). Latent dirichlet allocation. *Journal of Machine Learning Research 3*, 993–1022.

Casey, M. A., R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney (2008). Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE 96*(4), 668–696.

Grosche, P., M. Müller, and J. Serrà (2012). Audio content-based music retrieval. *Dagstuhl Follow-Ups 3*.

Guo, A. and H. Siegelmann (2004). Time-warped longest common subsequence algorithm for music retrieval.

Helén, M. and T. Virtanen (2007). Query by example of audio signals using euclidean distance between gaussian mixture models. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, Volume 1, pp. I–225. IEEE.

Hoffman, M. D., D. M. Blei, and P. R. Cook (2008). Content-based musical similarity computation using the hierarchical dirichlet process. In *ISMIR*, pp. 349–354.

Hu, D. J. (2009). Latent dirichlet allocation for text, images, and music. *University of California, San Diego. Retrieved April 26*, 2013.

Hu, P., W. Liu, W. Jiang, and Z. Yang (2014). Latent topic model for audio retrieval. *Pattern Recognition 47*(3), 1138–1143.

Kim, S., S. Narayanan, and S. Sundaram (2009). Acoustic topic model for audio information retrieval. In *Applications of Signal Processing to Audio and Acoustics, 2009. WASPAA'09. IEEE Workshop on*, pp. 37–40. IEEE.

Lebanon, G. (2012). Sequential document representations and simplicial curves. *arXiv preprint arXiv:1206.6858*.

Lebanon, G., Y. Mao, and J. V. Dillon (2007). The locally weighted bag of words framework for document representation. *Journal of Machine Learning Research 8*(10), 2405–2441.

Lu, Y. and J. E. Cabrera (2012). Large scale similar song retrieval using beat-aligned chroma patch codebook with location verification. In *SIGMAP*, pp. 208–214.

McVicar, M., R. Santos-Rodríguez, Y. Ni, and T. De Bie (2014). Automatic chord estimation from audio: A review of the state of the art. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on 22*(2), 556–575.

Müller, M. and S. Ewert (2011a). Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, Miami, USA.

Müller, M. and S. Ewert (2011b). Chroma toolbox: Matlab implementations for extracting variants of chroma-based audio features. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR), 2011. hal-00727791, version 2-22 Oct 2012*. Citeseer.

Müller, M., S. Ewert, and S. Kreuzer (2009, April). Making chroma features more robust to timbre changes. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Taipei, Taiwan, pp. 1869–1872.

Müller, M., F. Kurth, and M. Clausen (2005). Audio matching via chroma-based statistical features. In *ISMIR*, Volume 2005, pp. 6th.

Ni, Y., M. Mcvicar, R. Santos-Rodriguez, and T. De Bie (2011). Harmony progression analyzer for mirex 2011. *Proceedings of the 6th Music Information Retrieval Evaluation eXchange (MIREX)*, 1–4.

Ni, Y., M. McVicar, R. Santos-Rodriguez, and T. De Bie (2012). An end-to-end machine learning system for harmonic analysis of music. *Audio, Speech, and Language Processing, IEEE Transactions on 20*(6), 1771–1783.

Pachet, F. and J.-J. Aucouturier (2004). Improving timbre similarity: How high is the sky. *Journal of negative results in speech and audio sciences 1*(1), 1–13.

Ren, L., D. B. Dunson, and L. Carin (2008). The dynamic hierarchical dirichlet process. In *Proceedings of the 25th international conference on Machine learning*, pp. 824–831. ACM.

Riley, M., E. Heinen, and J. Ghosh (2008). A text retrieval approach to content-based audio retrieval. In *Int. Symp. on Music Information Retrieval (ISMIR)*, pp. 295–300.

Sakurai, Y., C. Faloutsos, and M. Yamamuro (2007). Stream monitoring under the time warping distance. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pp. 1046–1055. IEEE.

Shokoohi-Yekta, M., J. Wang, and E. Keogh (2015). On the non-trivial generalization of dynamic time warping to the multi-dimensional case. In *Data Mining. Proceeding of the 2015 International Conference on*, pp. 39–48. SIAM.

Zhu, X. (2013). Persistent homology: An introduction and a new text representation for natural language processing. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pp. 1953–1959. AAAI Press.