TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY
Julkaisu 815 • Publication 815

Marko Helén

Similarity Measures for Content-Based Audio Retrieval

Tampere 2009

Tampereen teknillinen yliopisto. Julkaisu 815
Tampere University of Technology. Publication 815

Marko Helén

Similarity Measures for Content-Based Audio Retrieval

Thesis for the degree of Doctor of Technology to be presented with due permission for public examination and criticism in Tietotalo Building, Auditorium TB103, at Tampere University of Technology, on the 11th of June 2009, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2009

ISBN 978-952-15-2168-3 (printed)
ISBN 978-952-15-2356-4 (PDF)
ISSN 1459-2045

Abstract

Personal multimedia databases contain thousands of items and other databases on the Internet may contain even billions of items. Finding a particular item manually from such databases becomes overwhelming and thus automatic search engines are required to lighten the job. Query by example refers to automatically finding multimedia items from a database, which are similar to the example provided by the user. This is an important task in modern multimedia databases.

This thesis deals with automatic query by example of audio samples. The emphasis is on representation and distance measures between two audio signals, which are used to estimate the similarity between these two signals. The thesis also covers computational issues, which are highly important when it comes to practical implementation of the algorithms.

Two different audio signal representations are proposed. These representations are interconnected, since the first separates drums from a polyphonic music signal although the same approach could be used to separate other parts of the original signal as well, for example, harmonic instruments. The second representation models the harmonic sound using only a few parameters. The proposed method is based on Mel frequency cepstral coefficients, which are further modeled using attack-decay-sustain-release curves with temporal evolution of harmonic instruments.

Most distance measures, used in audio signal processing, are based on dividing a signal into frames, extracting perceptually motivated features from each frame, and calculating the distance between the features. Most of the proposed distance measures use Gaussian mixture models to estimate the probability density functions of the frame-wise features and calculate the distance between the Gaussian mixture models. However, the thesis also introduces a parameter free distance measurement. This is based on compression ratios of audio signals and hence it removes the user influence on the results, since no features or other parameters need to be set.

In a query by example application, the similarity between the example provided by the user and each database item need to be calculated in order to obtain a ranked list of database samples. However, in practical applications this operation is very time-consuming if the database contains millions of items. The proposed method applies key-sample transformation to reduce the series of feature vectors, used to represent each signal, into a single feature vector. The database is then clustered and the search is restricted to only a few clusters, thus saving retrieval time with some loss of accuracy.

Preface

This work has been carried out at the Department of Signal Processing, Tampere University of Technology, during 2002–2009. I wish to express my deepest gratitude to my supervisor Professor Moncef Gabbouj for his guidance, and collaboration during my thesis work. I also wish to thank my former supervisor Professor Jarkko Niittylahti and the reviewers of the thesis Professor Jay Kuo and Professor Mircea Giurgiu.

I would like to thank the members of Audio Research Group for providing me the most professional and relaxed working environment during the past seven years. Especially I wish to thank the group leader Professor Anssi Klapuri for the opportunity to work in the group, Tuomas Virtanen who has guided me in my research, and my colleagues Jouni Paulus, Matti Ryynänen, Pasi Pertilä, Hanna Silén, and Elina Helander. At the Nokia Research Center I wish to thank Tommi Lahti for the co-operation in research projects.

I wish to thank my parents Marja and Olavi for their constant support. Finally, the warmest thanks go to my wife Suvi for all her love and understanding.

Marko Helén
Tampere, 2009

Contents

List of Abbreviations

AAC	Advanced audio coding
AMR	Adaptive multi-rate
ADSR	Attack, decay, sustain, release
BIC	Bayesian information criterion
CD	Compact disc
DCT	Discrete cosine transform
DFT	Discrete Fourier transform
GMM	Gaussian mixture model
EM	Expectation-maximization
F_0	Fundamental frequency
FFT	Fast Fourier transform
HMM	Hidden Markov model
ICA	Independent component analysis
IDFT	Inverse discrete Fourier transform
ISA	Independent subspace analysis
KL	Kullback-Leibler
LDA	Linear discriminant analysis
MFCC	Mel-frequency cepstral coefficient
MIR	Music information retrieval
MP3	MPEG-1 audio layer 3
MPEG	Moving picture experts group
NCD	Normalized compression distance
NID	Normalized information distance
NMF	Non-negative matrix factorization
PCA	Principal component analysis
PDF	Probability density function
QBB	Query by beatboxing
QBE	Query by example
QBH	Query by humming
RMS	Root mean square
SNR	Signal-to-noise ratio

SVM	Support vector machine
UBM	Universal background model

List of Symbols

a_n	Amplitude of the n^{th} FFT bin
$C(s)$	Size of compressed s
$\mathrm{d_B}$	Bhattacharyya distance
$\mathrm{d_M}$	Mahalanobis distance
f	Frequency
f_n	Frequency of the n^{th} FFT bin
$H_j(f)$	Frequency response of the j^{th} triangular filter at frequency f
$G(\mathbf{x})$	Gaussian impulse
I	Number of components in GMM
$K(s)$	Kolmogorov complexity of string s
l	Lag index
L	Maximum lag
m_j	Magnitude of j^{th} mel-band
N_c	Number of clusters
N_{comp}	Number of components in NMF
N_f	Length of the feature vector
N_{fft}	Number of FFT bins
N_{fr}	Number of frames
N_h	Number of harmonics in a harmonic sound
N_{hop}	Hop between consecutive frames
N_{mel}	Number of mel-bands
N_{obs}	Number of observations
NC	Number of required calculations
N_{sin}	Number of sinusoids
λ	Penalty factor
φ	Phase
$s(n)$	Time domain signal
$\mathrm{Tr}(X)$	Sum of diagonal elements in X (trace of X)
w	Weight of the Gaussian component
$W(f)$	Frequency dependent weight
x_j	j^{th} element of x

$\bar{x}$	Mean of $\mathbf{x}$
Γ	Autocorrelation function
σ	Standard deviation
ΔBIC	BIC difference
$\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$	Multivariate normal distribution
$\boldsymbol{\Sigma}$	Covariance matrix
$\boldsymbol{\mu}$	Mean vector
$\lvert A\rvert$	Determinant of matrix A

List of Figures

List of Tables

List of Publications

This thesis consists of the following publications, preceded by an introduction to the research field and a summary of the publications. Parts of this thesis have been previously published, and the original publications are reprinted, by permission, from the respective copyright holders. The publications are referred to in the text as [P1], [P2], and so forth.

[P1] M. Helén and T. Virtanen, "Perceptually Motivated Parametric Representation for Harmonic Sounds for Data Compression Purposes," in *Proceedings of the 6th International Conference on Digital Audio Effects (DAFx-03)*, pages 249-253, London, United Kingdom, September 8-11, 2003.

[P2] M. Helén and T. Virtanen, "Separation of Drums from Polyphonic Music Using Non-Negative Matrix Factorization and Support Vector Machine," in *Proceedings of the 13th European Signal Processing Conference (EUSIPCO 2005)*, Antalya, Turkey, September 4-8, 2005.

[P3] M. Helén and T. Lahti, "Query by Example Methods for Audio Signals," in *Proceedings of the 7th Nordic Signal Processing Symposium (NORSIG 2006)*, pages 302-305, Reykjavik, Iceland, June 7-9, 2006.

[P4] M. Helén and T. Virtanen, "Query by Example of Audio Signals Using Euclidean Distance Between Gaussian Mixture Models," in *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2007)*, vol. 1, pages 225-228, Honolulu, Hawaii, USA, April 15-20, 2007.

[P5] M. Helén and T. Virtanen, "A Similarity Measure for Audio Query by Example Based on Perceptual Coding and Compression," in *Proceedings of the 10th International Conference on Digital Audio*

Effects (DAFx-07), pages 173-176, Bordeaux, France, September 10-15, 2007.

[P6] M. Helén and T. Lahti, “Query by Example in Large Databases Using Key-sample Distance Transformation and Clustering,” in *Proceedings of the Third IEEE International Workshop on Multimedia Information Processing and Retrieval (IEEE-MIPR 2007)*, pages 303-308, Taichung, Taiwan, December 10-12, 2007.

[P7] T. Lahti, M. Helén, O. Vuorinen, E. Väyrynen, J. Partala, J. Peltola, and S.-M. Mäkelä, “On Enabling Techniques for Personal Audio Content Management,” in *Proceedings of the ACM International Conference on Multimedia Information Retrieval (MIR 2008)*, pages 113-120, Vancouver, Canada, October 30-31, 2008.

[P8] M. Helén and T. Virtanen, “Audio Query by Example Using Similarity Measures Between Probability Density Functions of Features,” submitted to *EURASIP Journal on Audio, Speech, and Music Processing*.

[P9] M. Helén, T. Lahti, and A. Klapuri, “Tools for Automatic Audio Management,” *Open Information Management: Applications of Interconnectivity and Collaboration* (S. Niiranen, J. Yli-Hietanen, and A. Lugmayr, Eds.), pp. 251–273, IGI Global, May, 2009.

Chapter 1

Introduction

The explosive growth of digital multimedia information has generated the need for automatic content analysis tools to manage the huge amounts of data. Multimedia data used to be designed mostly for professional use but during this decade the Internet and mobile devices have rapidly increased the amount of personal and on-line digital information. As a consequence, managing all the data manually has become an overwhelming task.

Traditionally, the emphasis on multimedia management has been in image and video analysis, but lately the importance of audio content has become apparent. In addition to pure audio signals, video analysis can also benefit from inclusion of the audio content [85]. For example, in a soccer game it can be difficult from the video to detect a goal, but it is easy to track the crowd cheering from the audio content. There are also cases when a certain event occurring in the background is impossible to detect without the audio. For example, a car approaching from behind which can not be observed visually, but auditory clues are obvious. For example, Li *et al.* used both audio and visual clues for video skimming [48].

Nowadays, audio management has also gained increasing commercial value. A good example is music information retrieval which is used in many ways in commercial applications such as Musipedia [58], Midomi [54], and Last.fm [44]. These applications retrieve music pieces based on a user profile or a given query.

This thesis deals with general audio retrieval and especially with query by example (QBE) applications. Probably the single most important aspect of such applications is the similarity estimation between audio samples. Thus, the focus of this thesis is on similarity measures, which are the crucial part in, for example, classification, segmentation, highlight extraction, and QBE tasks.

1.1 Overview of Audio Retrieval

Expectations with respect to audio retrieval applications are enormous, since people hope for a system that can perform the same operations as human being. However, the human ability to process information from the environment is astonishingly well developed. In addition to hearing and recognizing sounds, a human can also find semantic meaning in them and he/she has decades of experience on how to process different kinds of information. A human can, for example, select from a mixture of several sound sources the interesting one and follow it. Automatic systems often try to mimic the properties of human hearing but are still far behind. A comprehensive study in content based retrieval was made in the MUVIS framework [21, 38, 59], which proposed an overall system performing both audio and visual segmentation, classification, indexing, and retrieval. Kiranyaz *et al.* have also proposed progressive query [39] and database indexing [40] to achieve faster retrieval results.

The typical problem in audio retrieval applications is how the user can define the kind of audio clips, referred here as samples, that he/she is searching for. For a normal user, it might be difficult to verbally describe the properties of an audio sample. For this reason the query by example (QBE) method has attracted increasing interest lately. In QBE a user can give an example of the audio sample he/she is searching for. The example could be, for example, humming [50], clapping [34], [74], or a real audio sample [35]. An innovative method for audio collection navigation was proposed by Stewart *et al.* [74]. In their system, a user can navigate through a music collection in two or three dimensional space using ambisonics and binaural technology.

One challenge nowadays in multimedia retrieval is that significant amounts of recordings and processing are made using mobile devices. These devices are limited in processing power, memory, storage space, and battery duration. These limitations have to be considered when designing applications for mobile devices. Iftikhar and Gabbouj [31] proposed a client-server approach, where the user can make an audio or a visual query through the client which is a mobile device but the database is stored on the server which performs the actual query operations.

1.2 Problem Definition

Audio retrieval is a wide research area and can be approached from several different directions. In this section the approach used in this thesis and the sub problems are introduced and the focus is aimed at certain areas.

1.2.1 Signal Representation

Prior to analysis, an audio signal must be represented in an appropriate form. The purpose of the representation is to express the original signal in a more compact form and the aim is to minimize the perceptual quality loss in the process. Typical representations used nowadays in everyday data storage are, for example, wave format (.wav) in CDs, MP3 is used in mobile audio devices and computer environments, and AMR format is used for speech signals on mobile devices.

One goal for audio signal representation is to separate the original signal into sound objects and find representations for these different sound sources. One such method is introduced in this thesis. A music signal can be separated into a harmonic part and a transient part. A very efficient representation for the harmonic part of the signal, in terms of compression, is also introduced here.

Signal representations can also be used to assist the query by example applications. In Chapter 4 a similarity measure is introduces which uses modern audio codecs as a preprocessing step for the similarity measure. The other possibility is to take advantage of the sound source separation. For example, if trying to find similar rhythmic patterns from the music it would be beneficial to first separate drums and then compare only the drum parts of the original signals. This way the non-rhythmic content does not disturb the process.

For audio analysis purposes, the signal is normally represented using frame-wise acoustic features. The signal is divided into short, typically 10-60 ms frames and a set of features is extracted from each frame. The features usually aim at representing the most important perceptual characteristics of the original signals. Then, based on these features the properties of the signal can be compared to the properties of other signals. The signal representations will be discussed more thoroughly in Chapter 3.

1.2.2 Similarity of Audio Samples

Similarity between two audio samples as such is an ill-defined problem. It is impossible to know which properties of the sample the user is emphasizing. For example, if the query signal contains both music and speech, should the similarity be measured relative to speech, music, or both. Another question is whether the temporal sequence should be observed. For example, is it important that speakers in the sample are speaking in the certain order.

In music information retrieval (MIR), two approaches have been suggested in the literature to solve this problem. One approach is query by humming (QBH) [72]. This is applied in a situation when the user does not remember the name of the song he/she is searching for. The user can hum, sing, or whistle the song to the QBH application as an input. Then the system extracts the melody from the input humming and retrieves pieces having a similar melody from the database.

The second approach is query by beatboxing (QBB) [34]. In such an application, the user can beatbox or just tap the music piece. The system extracts the rhythm from the input and retrieves pieces having a similar rhythm. The problem in QBH and QBB applications is that the user may not be able, naturally, to hum the melody or tap the rhythm perfectly, so the system tries to retrieve the pieces which are most similar to the query.

Another alternative is QBE [P3] [P4] [P5]. This is especially applicable in a music recommendation system or in a situation when the user has a certain song and he/she wants to find other music pieces that have similar properties. In QBE applications, the user provides an example signal to the system. The properties of the input signal are analyzed and signals having similar properties are retrieved from the database. The results from such an application could be, for example, retrieving music from the same band or speech samples from the same speaker.

In addition to MIR, QBE is also useful with a general audio database when there is no way of anticipating the content of the database. The only restriction is the set of features used in the application since the similarity is defined according to these features. Obviously the features can be more specialized if there is some information on the contents of the database. However, audio databases collected with mobile phones, for example, can contain any kind of audio samples. In such cases features which describe general properties of audio samples must be used.

1.3 Author's Contributions to the Publications

The main results of the thesis are the following.

- Separating pitched instruments and drums from a music signal and representing harmonic sounds effectively [P1] [P2]
- Developing several novel similarity measures for audio signals based mostly on the PDFs of the feature vectors [P3] [P4] [P5] [P8]
- Developing a practical QBE application for a general audio database in which the proposed similarity measures are tested and proven to outperform previous measures [P7]
- Developing a transformation which enables the use of clustering methods to accelerate QBE in large audio databases [P6] [P9].

The publications used in this thesis are summarized below and the author's contributions are identified.

[P1] Parametric representation for harmonic sounds

The publication proposes a representation of harmonic sounds for data compression purposes. The representation significantly reduces the number of parameters while preserving the most important perceptual features of sounds. The proposed representation is used as part of an object-based audio coding system. The implementation, evaluation, and most of the writing work was done by the author.

[P2] Drum separation from music

The publication introduces a novel algorithm for the separation of pitched musical instruments and drums from a music signal. This is based on two-stage processing in which the signal is first separated into time-frequency components and then these components are organized into sound sources. Here we used the method for drum separation, but it can also be applied to other types of sound sources. The idea of the system and the implementation of non-negative matrix factorization came from Tuomas Virtanen.

[P3] Methods for query by example

This publication introduces three methods for QBE. The first method is based on hidden Markov models (HMMs). There is a model for the example signal and a universal background model (UBM). First, the likelihood for each database signal belonging to these two models is estimated. The second method is likelihood ratio test. The example and the database signal are concatenated and then n-component and $2n$-component Gaussian mixture models (GMMs) are generated. If the n-component model gives a higher likelihood for the concatenation, the signals are considered similar. The idea behind likelihood ratio test came from Tommi Lahti. The third method is called the histogram method in which feature histograms are estimated from each sample and the distances between these histograms are used as a similarity measure.

[P4] Euclidean distance between Gaussian mixture models

The publication introduces an improvement to the histogram method [P3] in which feature vectors were quantized before distance calculation. Here we model the continuous PDFs of the samples using Gaussian mixture models and derive a closed form solution for Euclidean distance between GMMs. The distance measure is applied to the QBE application. The derivation and implementation of closed-form Euclidean distance was made by Tuomas Virtanen.

[P5] Similarity measure based on compression ratios

The publication proposes a novel similarity measure for audio files which is based on their compression ratios. The main advantage of this method is that it is practically parameter free, thus it can easily be applied to a wide range of tasks and the user cannot affect the results as much as in parameter-laden algorithms. The method is compared with parameter-laden algorithms and considering the simplicity of the method, it gives very good results. The idea of using compression ratios for audio similarity came from Tuomas Virtanen. Implementation, evaluation, and most of the writing work was done by the author.

[P6] Accelerating the query by example via clustering

The publication proposes a novel transformation which enables the use of traditional clustering algorithms with samples which are modeled using a series of feature vectors. QBE becomes an exhausting task in large databases if the distance from the example signal to all database signals is calculated. Here we propose that the database could be clustered offline, thus in the query only the distances between the example and the samples in the nearest clusters are calculated. We accomplish major savings in computational costs with only a minor decrease in accuracy. Most of the research and writing work was performed by the author.

[P7] Techniques for Personal Audio Content Management

The publication introduces an audio management system, which has several co-operating management tools. The audio samples are first classified into four classes (silence, speech, music, and noise). Then, analysis tools are applied inside these classes. The tools include speaker change detection, gender and emotion recognition, and similarity estimation. The main contribution of this publication is to generate a system where different analysis tools can benefit from each other. The author's contribution was with the QBE algorithm and finding proper similarity measures for the task.

[P8] Audio Similarity Measures

The publication introduces the distance measures for audio signals, focus being on probability distribution based measures. The Euclidean distance between PDFs is derived for the full covariance matrix GMMs, and cross-likelihood ratio test and approximations of Kullback-Leibler (KL) divergence between GMMs are applied to audio similarity. The idea and implementation of Euclidean distance and idea of applying cross-likelihood ratio test came from Tuomas Virtanen.

[P9] Automatic Audio Management Tools

The publication introduces the concept of metadata and how this data is used in applications for automatic audio content management. The publication includes an introduction to audio classification and segmentation, query-by-example, music retrieval and recommendation, and computational issues. The main contribution is the idea that metadata is first extracted from the original data and then the metadata is used in applications. Tommi Lahti wrote the section 'Automatic information management for speech' and assisted on 'background'. Anssi Klapuri contributed in sections 'Music retrieval and recommendation' and 'Locality sensitive hashing'. The rest of the work was undertaken by the author.

1.4 Outline of the Thesis

This thesis is organized as follows. Chapter 2 gives an overview of the metadata extraction process and Chapter 3 introduces methods for acoustic signal modeling. These operations are preprocessing stages for further audio analysis and processing. Chapter 4 discusses the novel distance measures, which are used to estimate the similarity between audio samples. The similarity estimates are tested here in query by example tasks. This becomes overly time-consuming if the distance between all samples in the database is calculated. Chapter 5 briefly introduces the clustering method to lower the computational cost of such an application. Chapter 6 presents the evaluation criteria, databases, simulation results, and refers to comparative evaluations of the methods in literature. Chapter 7 summarizes the main conclusions of the thesis and outlines future directions for the development and the applications.

Chapter 2

Metadata Extraction

Prior to any multimedia signal analysis, there is a need to have information about the samples which can then be further processed. Such "data about data" is referred as metadata. Metadata can be derived for different aspects; for example, metadata concerning the content of the samples or metadata about the relationships between different samples. Multimedia analysis applications can then benefit from the extracted metadata and operate even without the original samples, which decreases the need for large storage capacity or fast file transfer.

2.1 Metadata Types for Multimedia Content

Basically there are five different types of metadata (for further information, see [46]):

- **Content metadata** describes the media content and it is deduced from the original data without any information from outside. For example, acoustic features like tempo or loudness can be directly analyzed from the musical content. Multimedia files also often hold tags, which are simple attributes that are included in the files. MP3 files, for example, have an ID3 container in conjunction with the content file, in which the user can insert the information about the artist, title of the song, and the name of the album. Also, the name of the file can contain information about the file.

- **Context metadata** is data about the situation where the multimedia sample is recorded or where it is consumed. For example, when taking a picture, the matters that are not captured on the film are context metadata. This might be the location where the picture is taken or the persons who are nearby. Contextual data would be especially important when organizing or grouping the media samples.

  Also, the information about the situation in which the user is consuming the media is considered as context metadata. For example, when, where, or what mood, does the user usually listen to a certain song.

- **Relationship metadata** describes the connections between a single media sample and other media samples. For example, all images taken by the same person have a relationship and if such a relationship is stored as a metadata it allows accessing the object with the help of other objects. If you have one such sample it is easy to access the others. All relationships are two-way relations, implying that if the sample A is used with sample B, it automatically implies that sample B uses sample A. The relationship can also exist between samples from different media types. For example, an audio clip may be related to an image.
- **Usage history metadata** describes how often the user accesses the content and how the content is used. Based on, for example, how often a certain song is played by the user, we can estimate how valuable this song is for the user. Usage history information also provides the possibility of creating a dynamically adapted playlist, based on which songs are most often played.
- **Community metadata** describes the relationships between the users. Using the community metadata, it is possible, for example, to recommend songs based on playlists by other people with similar taste.

In this thesis, content and relationship metadata are of most interest, since in a QBE task, the first step is to extract certain features from the content and then find which samples have the closest relationships based on these features.

2.2 Acoustic Features

Prior to any audio content analysis, there is a need to extract features from raw data signals. Features represent different properties of the content and thus, the choice of features depends on the task. In MIR, the musical features like rhythm or tempo should be emphasized whereas, in speech recognition the Mel-frequency cepstral coefficients are important.

When talking about feature extraction, the question arises: which features should be used? To answer this question, several feature selection methods have been developed. These can be used to reduce the dimensionality of the original feature space, because there might be features which are not relevant for the task or features which represent the same property of the signal. The most popular of these methods are principal component analysis (PCA) [77] and linear discriminant analysis (LDA) [6]. Both of these algorithms reduce the dimensionality through linear transformations. PCA assumes that most of the information is contained in the feature dimensions having the largest variation. Thus, it optimizes the transformation matrix to maximize the variance in the projected space. LDA, on the other hand, optimizes the ratio of between-class variation and within-class variation for the best discrimination of the classes

Below is a short introduction to the features used in this thesis.

2.2.1 Frequency Domain Features

- **Mel-frequency cepstral coefficients** (MFCCs) (used in [P1]-[P9]) are probably the most popular feature used in audio signal processing. They represent the most important properties of the audio sample, based on human sound perception. MFCCs were originally applied to speech analysis [15]. They are calculated using the following procedure. First, the input signal is filtered using frequency-dependent weights, which simulate the response of the human auditory system. Then, the spectrum is emphasized using a filterbank, which consists of triangular filters spaced uniformly in the Mel-frequency scale and with the heights scaled to unity. The Mel-scale is defined as

  $$\mathrm{Mel}(f) = 2595\log_{10}(1+\frac{f}{700}), \tag{2.1}$$

  where f is the original frequency in Hertz. The energies in each filterbank channel are summed and a logarithm is used to compress the dynamic range.

$$m_j = \log_{10} \sum_{n=1}^{N_{fft}} W(f_n) H_j(f_n) a_n^2, \tag{2.2}$$

where N_{fft} is the number of FFT bins, $W(f)$ is the frequency dependent weight at f, $H_j(f)$ is the frequency response of the j^{th} triangular filter at frequency f, and a_n is the amplitude of n^{th} fast Fourier transform (FFT) bin. Finally, the discrete cosine transform (DCT) is taken from channel magnitudes to obtain cepstral coefficients

$$c_{mel}(i) = \sum_{j=1}^{N_{mel}} m_j \cos\left(\frac{\pi}{N_{mel}}(j - \frac{1}{2})i\right), \tag{2.3}$$

where N_{mel} is the number of bands. [15]

- **Spectral centroid** (used in [P2]-[P9]) defines where the center of mass of the spectrum resides [62]

$$\text{SC} = \frac{\sum_{n=1}^{N_{fft}} n \cdot a_n}{\sum_{n=1}^{N_{fft}} a_n}. \tag{2.4}$$

- **Spectral spread** (used in [P3]-[P9]) is the variance of spectral centroid [62]

$$\text{SS} = \frac{\sum_{n=1}^{N_{fft}} (n - \text{SC})^2 a_n}{\sum_{n=1}^{N_{fft}} a_n}. \tag{2.5}$$

  The feature measures how concentrated the energy is around the centroid of the spectrum. It separates tone-like signals from noise signals.

- **Spectral flux** (used in [P3]-[P9]) is an average variation between two consecutive frames in a 1-s window. It measures how quickly the spectrum of the signal is changing. It is measured as the Euclidean distance between the normalized spectra of two consecutive frames [51, 70].

- **Noise likeness** (used in [P2]-[P9]) is the correlation coefficient between the spectrum of the original signal, and the convolution between the local maxima of a spectrum and a Gaussian impulse with zero mean. The Gaussian impulse is given by

$$G(\mathbf{x}) = \exp(-\frac{\mathbf{x}^2}{2\sigma^2}). \tag{2.6}$$

The feature represents the noisiness of the signal, which is intended to describe percussive sounds. [78]

- **Roll off point 85%** (ROP) (used in [P2]) is the frequency below which 85 % of the spectrum energy resides [62]

$$\sum_{n=1}^{\mathrm{ROP}} a_n^2 = 0.85 \sum_{n=1}^{N_{fft}} a_n^2. \tag{2.7}$$

2.2.2 Time Domain Features

- **Peak time** (used in [P2]) is an average length of peaks in a time domain gain curve. Peak is defined as an area where the signal value is at least 0.2 times the maximum value.
- **Peak fluctuation** (used in [P2]) means the standard deviation in peak times.
- **Harmonic ratio** (used in [P3]-[P9]) is used to separate music from noise. It estimates the ratio between harmonic and non-harmonic components in the signal. It is calculated as a maximum of the normalized autocorrelation function of the signal, defined as:

$$\Gamma_j(l) = \frac{\sum_{n=0}^{N_s-1} s_j(n) s_j(n-l)}{\sqrt{\sum_{n=0}^{N_s-1} s_j(n)^2 \sum_{n=0}^{N_s-1} s_j(n-l)^2}} (j \leq l \leq L; 0 \leq j \leq N_{fr}-1), \tag{2.8}$$

where j is the frame index, N_s is the number of sample frames in original signal $s(n)$, l is the lag index, $s_j(n)$ is defined as $s(j \cdot N_{hop} + n)$, N_{hop} is the hop between consecutive frames, L is the maximum fundamental period and the total number of frames in $s(n)$ is N_{fr}.

Harmonic ratio is defined as the maximum autocorrelation within frame.

- **Maximum autocorrelation lag** (used in [P3]-[P9]) is the index where autocorrelation reaches the maximum value. It estimates the fundamental period of the signal [37, pp. 34-35]. For periodic signals the maximum autocorrelation lag corresponds to multiples of fundamental period. This feature is especially useful in detecting drum patterns which are periodic by nature.

- **Crest factor** (used in [P3]-[P6] and [P8], [P9]) describes the magnitude of percussions occurring in the waveform of the frame [62]. It is defined as:

$$\mathrm{CrestFactor} = \frac{\mathrm{Peak}}{\mathrm{RMS}}, \tag{2.9}$$

  where *Peak* is the maximum absolute amplitude and *RMS* is the root mean square value of the waveform.

- **Standard deviation** (used in [P2]) describes how far the observations are on average, from the mean value. It is defined as

$$\sigma = \sqrt{\frac{1}{N_{obs}-1}\sum_{n=1}^{N_{obs}}(x_n-\bar{x})^2}, \tag{2.10}$$

  where $\bar{x}$ is the mean of observations x_n and N_{obs} is the number of observations in a frame.

- **Total energy** (used in [P3]-[P9]) is the average power of all observations in a frame [62]

$$\mathrm{TE} = \frac{\sum_{n=1}^{N_{obs}} x_n^2}{N_{obs}}. \tag{2.11}$$

- **Variance of instantaneous power** (used in [P3]-[P9]) is the variance from the average total energy calculated in a one-second window.

- **Zero crossing rate** (used in [P3]-[P9]) defines how often the signal changes from positive to negative or back [62].

- **Pause rate** (used in [P3]) measures the amount of pauses in the signal. It is defined as the ratio between silent frames and the total number of frames. Pause rate is typically used to distinguish a speech signal from a music signal. In speech there are a lot of short breaks whereas typically in music the number of breaks is close to zero [41]. A closely related feature, called the transition rate, measures the frequency of transition between silent and non-silent segments [41].

Further discussion and detail on audio features can be found in [62].

2.3 Limitations in Audio Recordings

Previously most of the multimedia content was created by professionals using professional equipment. Nowadays, more and more of the multimedia is recorded by amateurs using mobile phones or other personal devices. This has decreased the quality of the recordings which should be taken into account when designing multimedia retrieval applications. Here are some shortcomings which appear in audio recordings:

- The manufacturers of mobile devices aim at low production costs and thus acoustic components are not capable of capturing high quality audio. Frequency response is typically highly irregular and the dynamic range is very narrow. The microphones, especially in mobile phones, are very directional. They are designed to capture speech and are optimized for very short distances between source and microphone.

- Recordings are often made by complete amateurs and thus the recording conditions are not well set up. Recordings are also often made without any planning which leads to poor quality. There is often traffic or wind noise or other background voices in the recordings. Variation in the volume level is also high. All these factors together lead to poor signal-to-noise ratio.

- Mobile phones are focused on capturing speech signals at the expense of other audio information. Typically, there are audio enhancements made to the input signal, in order to maximize the quality of recorded speech in terms of pleasantness and intelligibility. Such enhancement is, for example, achieved by concentrating only on speech frequency bands and removing everything outside that. Also, dynamic compression and discontinuous transmission algorithms fade out the properties which would have been valuable for some audio content.

- Nowadays, there are several widely used audio codecs which are not compatible with each other. It would be preferable for audio processing tools to be able to operate independently of the representation format.

- Volume levels and sampling rates differ between the recordings. Hence normalization and resampling are required.

  This thesis concentrates on similarity measures which are applied to a query by example task. The measures are applicable to all audio recordings. However, the shortcomings mentioned above reduce the accuracy of the methods. The representation of the signal is irrelevant as long as acoustic features can be extracted from the signal. Furthermore, despite the noisy or otherwise poor recording conditions, the query by example application retrieves the most similar samples from the database, based on the acoustic features.

Chapter 3

Signal Modeling

Signal modeling, for the purpose of audio storage, aims at representing the original signal in a way that minimizes the number of parameters required but preserves the most important perceptual properties of the signal. This is an important factor in all practical audio analysis applications. Storage capacity is always limited and efficient methods for representing the original signals are required. For example, in query by example application, typically, the whole database has to be queried. Thus, it would be beneficial to represent the signals in efficient manner instead of storing all the original data. The query by example application can also use the acoustic features calculated in the signal modeling phase instead of calculating everything by itself. Next, audio modeling methods are introduced, some of which are used for audio similarity calculation in Chap. 4.

3.1 Perceptual Audio Coding

Perceptual audio codecs are commonly used in a wide range of commercial applications. Practically, all audio on the internet, on portable devices, or on digital television is perceptually coded. The most familiar codecs are MP3 [56] and AAC [10] which are optimized for music signals, but there are also specialized codecs for other audio, for example, AMR [68] is optimized for speech signals.

Audio codecs aim at compressing the original audio file without losing the perceptual quality of the signal [60]. The above mentioned codecs are lossy algorithms thus they lose information from the source signal. However, they utilize the properties of the human auditory system to achieve compression without audible changes. An especially useful property is the masking phenomenon. It means that when the sounds which are close to each other in the frequency domain occur simultaneously, the louder sound masks the other sound making it inaudible. Thus, the silent sound can be ignored without losing perceptual quality. In practice, codecs achieve compression via quantization of the files. First, a filter bank or a short-time frequency transform is used to form a time-frequency representation. Second, an auditory model estimates how coarse quantization can be applied without affecting audibility. More bits are allocated to the parts which are perceptually the most important. Furthermore, after quantization, redundancy in codewords can be reduced by entropy coding [83].

These audio codecs will be applied to compression based similarity measures in Chap. 4, where they are used in combination with entropy coding to estimate similarity.

3.2 Separation of Drums and Pitched Instruments

One possible signal modeling method is to represent the audio as sound objects. This is a challenging operation, but it also paves the way for several audio processing tools. For example, song separation from a polyphonic music signal can be used in karaoke applications [67] and instrument separation can be used for automatic transcription [55]. Next a method to separate drums and pitched instruments from a music signal is introduced. A similar approach could be applied for other instruments as well. This algorithm was proposed in [P2].

Fig. 3.1 illustrates a drum separation system. The input signal is first separated into components using non-negative matrix factorization (NMF) [45]. Second, several features are extracted from the components. Features used here are MFCCs, spectral centroid, standard deviation, roll off point, noise likeness, maximum autocorrelation lag, peak time, and peak fluctuation. Third, the components are classified into either pitched or drums using support vector machine (SVM) which was trained using examples from both classes. Finally, the components within both classes are synthesized and summed to obtain drum part and pitched part from the original signal.

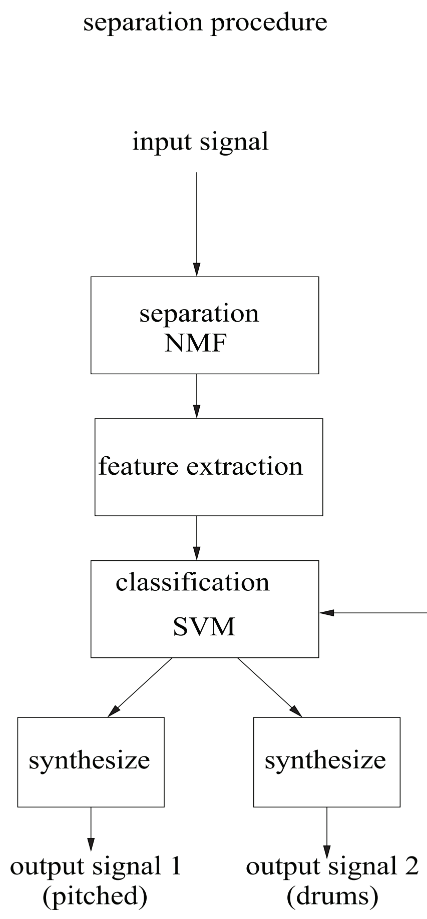

Figure 3.1: A block diagram of the drum separation system.

3.2.1 Non-Negative Matrix Factorization

Audio signals in the real world, for example music, are usually mixtures of several sound sources. The operation, which divides the mixture into sources, is referred as sound source separation. Such a process has several applications in audio data analysis and manipulation. NMF is a signal separation method, which learns structures from the data without prior information. Such algorithms are referred as data-driven methods and they have been applied to music signal separation and analysis [1, 12, 78, 80, 81]. NMF models the input signal $\mathbf{X}$ as

$$\mathbf{X} \approx \mathbf{AS}, \tag{3.1}$$

where $\mathbf{X} = \left[\mathbf{x}_1, \ldots, \mathbf{x}_{N_{fr}}\right]$, $\mathbf{x}_i$ is the short-time spectrum in frame i, N_{fr} is the number of frames, $\mathbf{S} = \left[\mathbf{s}_1, \ldots, \mathbf{s}_{N_{comp}}\right]$, $\mathbf{s}_j$ is the spectrum of component j, N_{comp} is the number of components, and $\mathbf{A}_{j,i} = a_{j,i}$ is the gain of j^{th} component in frame i.

The components of NMF are estimated by minimizing the divergence between $\mathbf{X}$ and $\mathbf{AS}$ denoted as:

$$D(\mathbf{X}||\mathbf{AS}) = \sum_{i=1}^{N_f}\sum_{j=1}^{N_{fr}}(\mathbf{X}_{i,j}\log(\frac{\mathbf{X}_{i,j}}{[\mathbf{AS}]_{i,j}}) - \mathbf{X}_{i,j} + [\mathbf{AS}]_{i,j}). \tag{3.2}$$

The minimization is done using the update rules given in [45] and [P2]. The measure is always non-negative and it reduces to Kullback-Leibler (KL) divergence [45] if $\sum_{i=1}^{N_f}\sum_{j=1}^{N_{fr}}\mathbf{X}_{i,j} = \sum_{i=1}^{N_f}\sum_{j=1}^{N_{fr}}[\mathbf{AS}]_{i,j} = 1$.

The other data-driven method, which we tested for separating the signal into components, was independent component analysis (ICA) [30]. However, in our simulations NMF outperformed ICA and thus the former was chosen for further studies. It was observed, when listening to the separated components, that the separation quality of ICA in our task was poor.

3.2.2 Support Vector Machines

Classification of data is an important operation in many data mining applications. Having an unknown dataset, we would often like to classify the items into categories. In our case NMF separates a source signal into $n = 20$ components and we would like to know which of these components belong to drums. Since we have prior information about the categories of the components, we can use a supervised classifier, meaning that example items from categories are provided for the algorithm and when a new item is presented, the algorithm can decide in which category it belongs. For this pattern recognition task we applied SVMs [26].

SVM is first trained with feature vectors from both classes. From the training procedure, the SVM learns the characteristics of the data sets and finds a hyperplane in the feature space which separates the two classes. This is an optimization problem, which is described more thoroughly in [26]. The shape of the hyperplane depends on the kernel function used in the algorithm. The common kernels are, for example, polynomial, sigmoid, and radial basis function. The most suitable kernel depends on the task. In our task the polynomial kernel provided

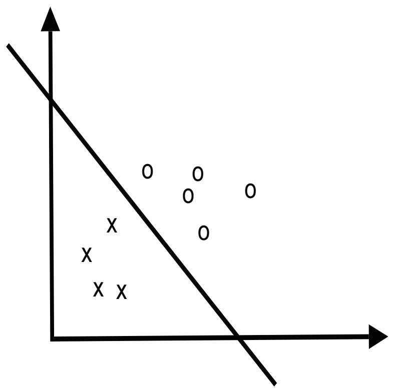

Figure 3.2: An example of linear classification task.

the best separability. An example of linear separability is illustrated in Fig. 3.2.

Finally, the component spectrograms are summed within both classes. Phases are obtained from the original mixture spectrogram and the transformation back to the time-domain is performed using inverse Fourier transform. To reduce discontinuities, the frames are windowed using a Hanning window and overlap-add.

3.2.3 Experimental Results

The performance of the proposed method was simulated and compared to alternative preprocessing methods. Quantitative evaluation of the separation performance would have required reference signals. We did not have access to any material from which the pitched instruments and drums could be obtained separately. Therefore, test signals were generated by mixing harmonic signals and drums from various sources. The detailed explanation of the experimental setup is in [P2].

100 samples were used for the testing. The samples were generated by mixing harmonic and drum signals at equal energy levels. The samples were different in the training and testing. The original samples were stored as references before mixing to allow the evaluation of the separation quality. Each test sample was separated into components using the NMF, and classified using the SVM. The components within both classes were summed and synthesized. The separation quality of harmonic and drum signals was measured using the signal-to-noise

method	SNR / dB
sinusoidal model (drums)	1.35
onset detection (drums)	3.05
NMF + GMM (drums)	7.0
NMF + SVM (drums)	7.33
NMF + SVM (harmonic)	2.46

Table 3.1: Average signal-to-noise ratios (SNR) obtained using different separation algorithms.

ratio (SNR) of a separated signal defined as

$$SNR = 10 \log_{10} \frac{\sum_n s(n)^2}{\sum_n (s(n) - \hat{s}(n))^2}, \tag{3.3}$$

where $s(n)$ is the original signal and $\hat{s}(n)$ is the separated signal. The SNR is calculated for all separated harmonic and drum signals.

The proposed method was compared against other approaches. Table 3.1 presents the average SNRs obtained with different methods. Sinusoidal modeling was done using an algorithm which detects the prominent peaks in the spectrum using the sinusoidal likeness measure [65]. The analysis and synthesis of the sinusoids was done without continuation between frames. The harmonic part of the signal was analyzed with a sinusoidal model, and the estimate of the drum part was obtained by subtracting the synthesized sinusoids from the original signal. The threshold value for the detection of the sinusoids was tuned to maximize the SNR of the separated parts.

The other method tested is based on the onset detection algorithm proposed by Klapuri [43]. The onsets were estimated by finding sharp increases of the signal energy within 21 frequency bands. A short-duration segment of the signal after each onset was judged to belong to the drum signal. The harmonic part was obtained by removing the estimated onset segments from the original signal. The threshold for the onset detection and the duration of the segments were tuned to maximize the SNR of the separated signals. The optimal segment duration was found to be 66 ms.

For the drum part, the average SNR obtained using the proposed method is clearly higher than others. However, the SNR of the harmonic part is not as high. This can be explained by the signal model, which suits drum signals better. However, by calculating the harmonic part by subtracting the synthesized drums from the original signal, the SNR for the harmonic part becomes equal to the SNR of the drum part.

In order to measure the classification performance, each separated component in the test data was labeled as harmonic instruments or drums. Since a component may have both harmonic and drum content, it was put in the class, where it most resembles the reference signal. For each component, the residuals between the synthesized component and original signals of pitched instruments and drums were calculated. The energy ratios between the residuals and the original signals were calculated and the one with the smaller residual-to-signal ratio was chosen. That is, the component was labeled as drums if

$$\frac{\sum_n (d(n) - \hat{s}(n))^2}{\sum_n d(n)^2} \leq \frac{\sum_n (h(n) - \hat{s}(n))^2}{\sum_n h(n)^2}, \tag{3.4}$$

where $h(n)$ and $d(n)$ are the original pitched-instrument and drum signals respectively, and $\hat{s}(n)$ is the separated component. This labeling was used as a reference in the classification.

The percent of correct classifications was used to measure the quality of the classification. The measure is an average of all the test samples. When testing different feature sets, the features calculated from the spectrum seem to work better than the ones calculated from gain. The best combination of features in our simulations included MFCCs, noise-likeness, centroid, roll off point, standard deviation, periodicity, peak time, and peak fluctuation. With this feature set, the percent of correct classifications using SVM was up to 93 percent and using GMMs 92 percent. However, almost equally good results (80-90 percent) were achieved with several different feature sets. Other features with good classification capability were percussiveness and crest factor. Even MFCCs alone gave the classification percent of 82. Furthermore, the components clustered in the wrong class usually had a considerable amount of content from both classes. For these components, it is hard even for a human to decide which class they belong in.

Gillet and Richard [22] tested the drum separation algorithm in [P2] against other state of the art drum separation approaches. They stated that our method “obtained a high signal-to-interference ratio - illustrating the ability of this algorithm to strongly discriminate drum components”. Furthermore, Moreau and Flexer [55] extended our work making drum transcription from separated drum sources.

3.3 Parametric Representation of Harmonic Instruments

The separation of instruments in a polyphonic music signal enables the representation of each instrument using their special characteristics. In [P1] we propose one such representation for harmonic sounds. In the case of harmonic sounds, the characteristic properties are:

- Perfectly harmonic sound contains the fundamental frequency component (F_0) and overtones which are integral multiples of F_0.
- the phases are not perceptually important and thus random phases could be used in synthesis.
- The time-frequency spectrum is smooth, hence resulting in slowly-varying amplitudes of harmonic sounds.

An example of an amplitude slope of a harmonic component is illustrated in Fig. 3.3.

The harmonic part of the sound of the j^{th} frame can be expressed as a sum of sinusoids

$$s_j(t) = \sum_{i=1}^{N_{sin}} a_i cos(2\pi f_i t + \varphi_i), \tag{3.5}$$

where t is the time index, N_{sin} is the number of sinusoids, a_i, f_i, and φ_i are the amplitude, frequency, and phase of the i^{th} sinusoid, respectively. A single harmonic sound can similarly be represented in frame j as a sum of the harmonic partials

$$h_j(t) = \sum_{n=1}^{N_h} a_n cos(2\pi n F_0 t + \varphi_n), \tag{3.6}$$

where N_h is the number of harmonics including the fundamental frequency F_0.

In this thesis, a representation for the amplitude spectrum of harmonic sounds is proposed. The rough shape of the spectrum is modeled using MFCCs and the temporal evolution of MFCCs is further modeled using the attack-decay-sustain-release (ADSR) representation [32, pp. 51-66].

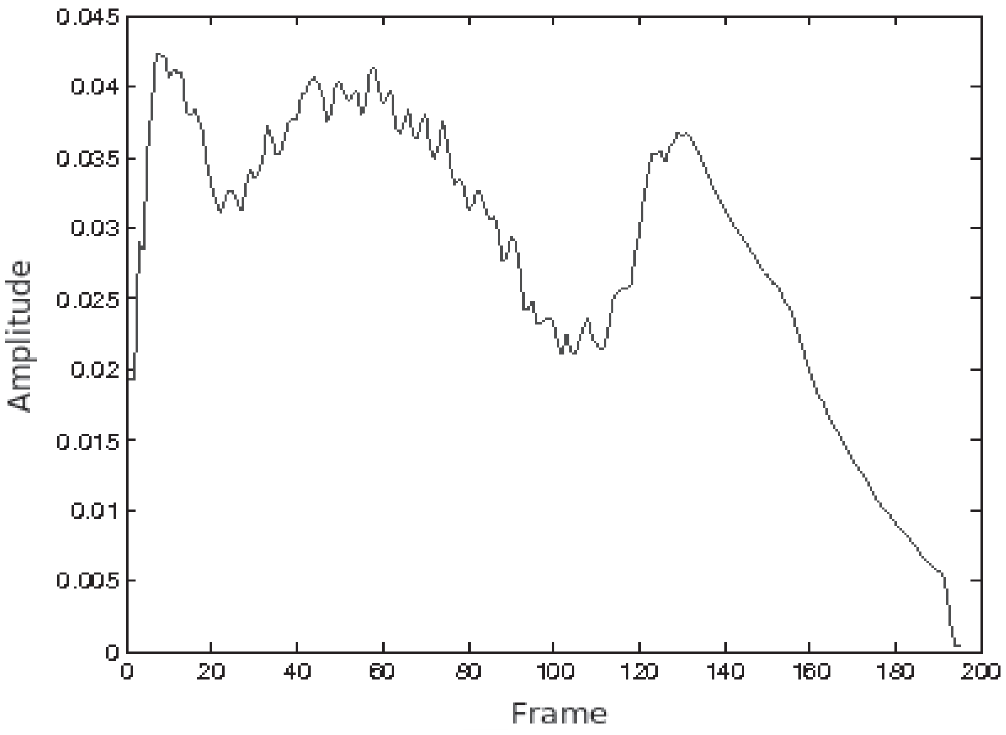

Figure 3.3: An example of harmonic sound.

3.3.1 MFCC Representation

MFCCs can be used to represent the rough shape of the amplitude spectrum [P1] and they are explained in Section 2.2. Fig. 3.4 illustrates the shape of the MFCC slope as a function of time for a harmonic sound.

When MFCCs are transformed back to amplitudes, the operations are performed backwards. First inverse discrete cosine transform (IDCT) is taken from the MFCCs to retrieve the energies in each frequency band. Then the energies within each band are divided uniformly inside the band. Details from the original signal are lost, but the rough shape of the amplitude spectrum is retained. The compression is achieved by removing the highest MFCCs.

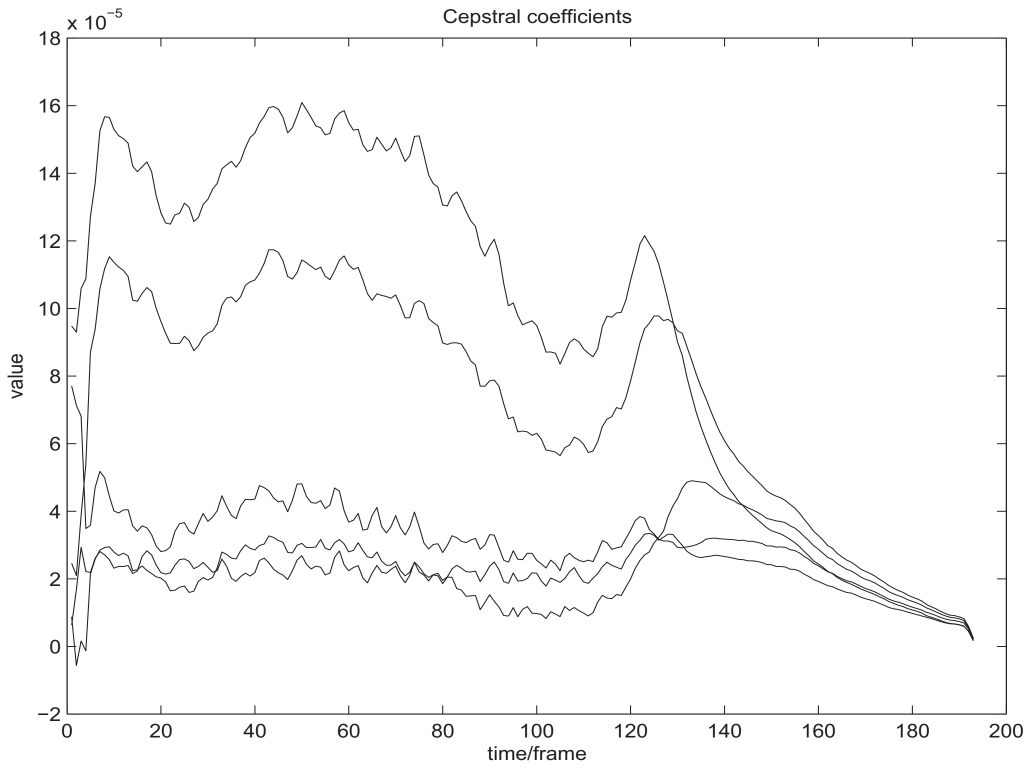

Figure 3.4: Five first MFCCs of the bowed cello as a function of time.

3.3.2 ADSR Representation

ADSR is often used in synthesizers to modulate the loudness of sound over time. In [P1] we propose using a similar approach to model the temporal evolution of the amplitude slope. The justification for such modeling is that the natural sounds usually have a smooth time-frequency spectrum resulting in slowly varying amplitudes of harmonic components. The shape of the amplitude slope depends on the instrument being played. However, this slope can roughly be represented with amplitude values in four instants of time, which define four stages. These are attack, decay, sustain, and release. First the amplitude rises to its highest value which is the attack time. Then, at the decay stage the amplitude drops to the sustain level. Sustain level continues until the note is released. Finally, the release time describes how fast the sound fades after the note ends. An example of ADSR curve is illustrated in Fig. 3.5.

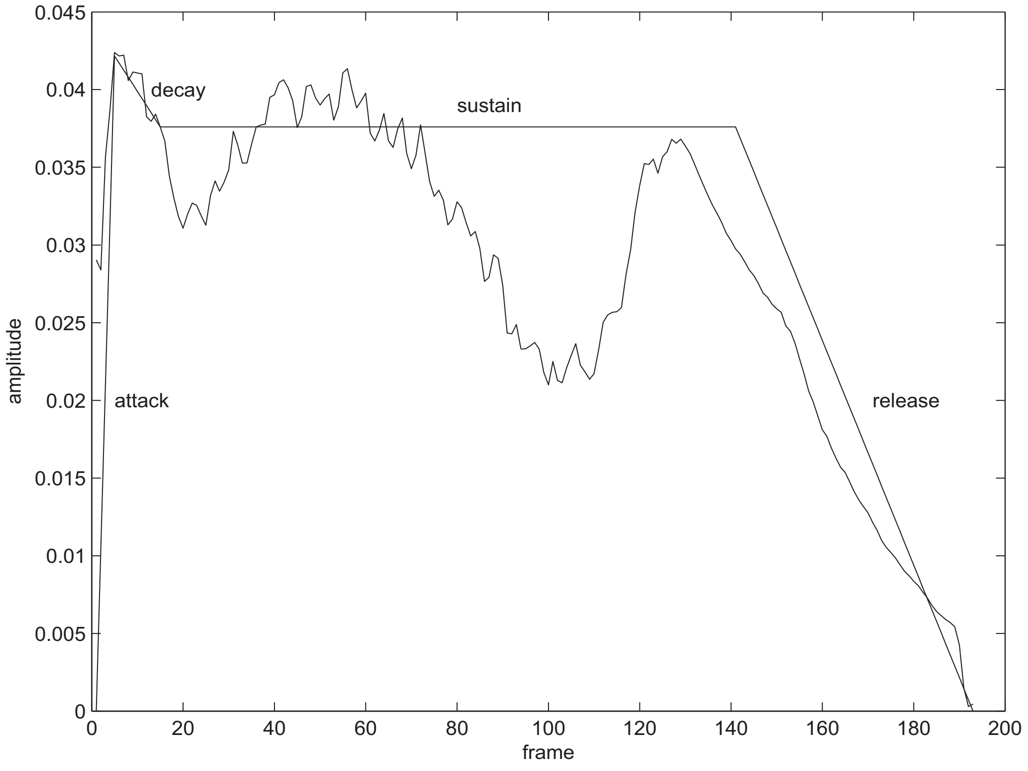

Figure 3.5: Amplitude slope of one harmonic component and the resulting ADSR slope.

The problem with the ADSR method is, how to estimate the time intervals for each stage. We applied the method which finds the maximum amplitude and the stages are defined with predetermined percent values of the maximum [32, pp. 54-55]. The attack starts when the value first rises above 10% of the maximum and ends at the maximum value where the decay starts. The sustain starts when the amplitude drops below 90% of the maximum and lasts until the amplitude is below 70% of the maximum. Finally, the release ends when amplitude drops below 10% of the maximum.

The ADSR model was improved by replacing linear lines in the original model with exponential curves. The curves are defined as

$$f(x) = v_0 + (v_1 - v_0)(1 - (1 - x)^n)^{\frac{1}{n}}, \tag{3.7}$$

where v_0 and v_1 are start and end points, respectively. x is the time parameter which is normalized between $[0, 1]$ and n is the parameter

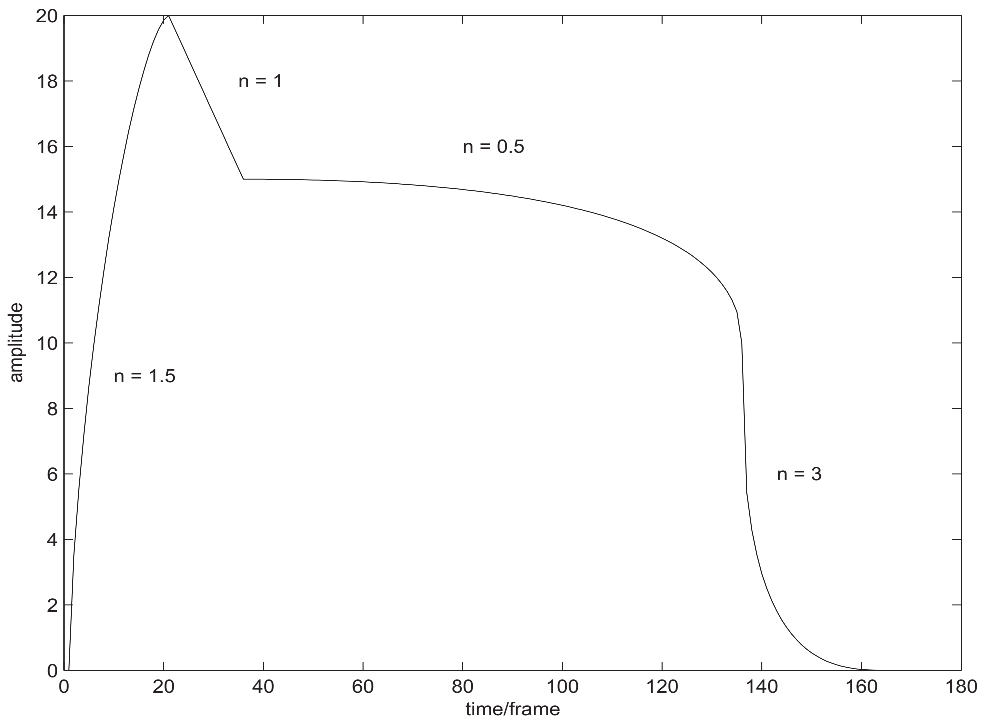

Figure 3.6: The effect of parameter n on ADSR curve model.

which defines the shape of the curve. If $0 \leq n < 1$ the curve is exponential, if $n = 1$ the curve is linear, and if $n > 1$ the curve is logarithmic. Fig. 3.6 clarifies the shapes.

We also further developed the state model. In [42] we proposed an interpolating state model for time-varying spectra modeling. The sequence of the acoustic feature vectors is represented by interpolating between states. The states are estimated by iteratively merging and deleting states from the state model. Details can be found in [42]. The algorithm provided approximately two times better accuracy compared to the vector quantization approach used as a reference method.

3.3.3 Combining MFCC and ADSR

In [P1] we propose a combined use of MFCC and ADSR representations for modeling the time-varying amplitude slope of harmonic sound. The

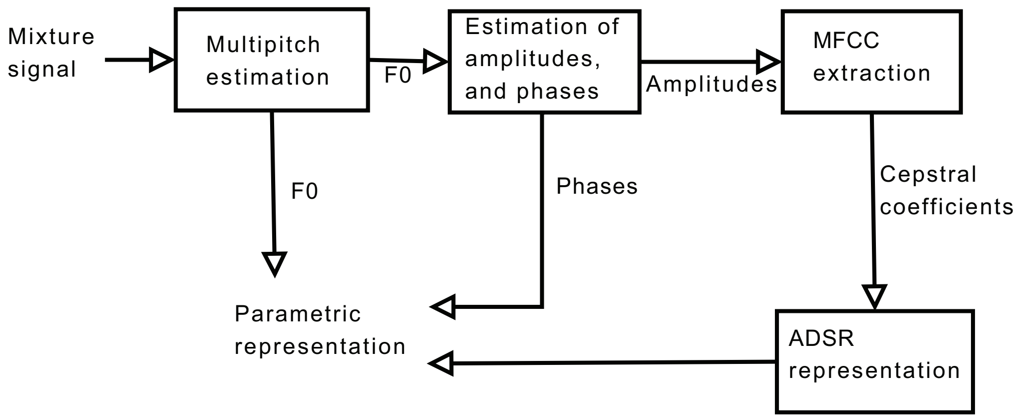

Figure 3.7: Block diagram of the harmonic sound representation algorithm.

block diagram of the method is illustrated in Fig. 3.7. First, the fundamental frequencies (F_0) of harmonic sounds were identified in the original signal with the multipitch estimator of Tolonen and Karjalainen [76]. The method calculates the enhanced summary autocorrelation function (ESACF) and the frequency of the F_0 is assumed to be the highest peak in the ESACF. Second, the amplitudes and phases of the harmonic components are estimated from the signal spectrum. Since the input is expected to be a clean harmonic sound the lower harmonics are found from integer multiples of the F_0 component. Third, if the same fundamental frequencies are found from consecutive frames, these are combined to one sound object. Finally, the MFCC-ADSR modeling is applied to the sound objects to obtain the final parametric representation. The harmonic components are synthesized from the parameters and summed to generate the resulting time domain signal.

The quality of the proposed method was measured using Perceptual Audio Quality Measure (PAQM) [5] and informal listening tests. Table 3.2 presents quality evaluations of some monophonic sounds. The values are logarithms of noise disturbance thus the smaller the value the better quality. It is observed that each stage of the processing usually degrades the quality of the sound. However, by listening to the results the difference is quite small and there are also cases in the table where the sound quality actually improves in terms of PAQM. For example, in violin sample the ADSR processed sample is better quality than unprocessed sample, but in this case already the unprocessed

	Unprocessed	MFCC	ADSR	MFCC+ADSR
CELLO	-3.2	-2.0	-2.9	-1.9
GUITAR	-5.1	-3.6	-4.5	-3.6
VIOLIN	-1.0	-0.4	-1.1	-0.4
BASS	-3.3	-2.7	-3.3	-3.0

Table 3.2: PAQM values (log(noise disturbance)) for monophonic sounds using different codings.

sample was rather low quality. The demonstrations of monophonic signals are available at http://www.cs.tut.fi/˜heln/demopage.html.

Chapter 4

Similarity Measures

This chapter introduces the measures which are used in the thesis to estimate the similarity between two audio samples. An important factor that has to be considered when estimating similarity between samples is that similarity and distance are two different things. Distance is an objective measure. On the other hand, similarity is subjective. It depends on the application at hand and also on personal preferences. For example, which one is more similar, the same song played by a different band or a different song played by the same band? Distance measures are, however, used to approximate the similarity of the samples.

One possibility for solving this "similarity" problem, which is left for future studies, would be asking for feedback from the user or the taking of several samples, instead of only one, as an example. Such approaches have been followed in image retrieval [75, 87], but not yet in content based audio retrieval. An application to audio would also require a large amount of work with user interfaces. On the other hand, there is software on the Internet such as Last.fm [44], which compares the playlists of users and can recommend songs for people having similar playlists and thus, similar musical taste. Such applications, however, do not base their search on the actual content of the signals, but only on usage history.

Traditional distance measures use simple statistics (mean, covariance, etc) of the features. For example, Mandel and Ellis [52] used the Mahalanobis distance for supervised music classification

$$\mathrm{d_M}(f,g) = (\boldsymbol{\mu}_f - \boldsymbol{\mu}_g)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_f - \boldsymbol{\mu}_g), \tag{4.1}$$

where f and g are feature distributions characterized with means $\boldsymbol{\mu}_f \in \Re^n$ and $\boldsymbol{\mu}_g \in \Re^n$, respectively. $\boldsymbol{\Sigma} \in \Re^{n\times n}$ is the covariance matrix of the features across all samples in the database. The Mahalanobis distance is independent of the scale of the features since it uses the variances of feature dimensions. If $\mathbf{\Sigma}$ is an identity matrix, the Mahalanobis distance reduces to the Euclidean distance.

The distance measure, which also considers the covariances of individual samples, is called the Bhattacharyya distance [9] and is defined as

$$\mathrm{d_B}(f,g) = \frac{1}{4}(\boldsymbol{\mu}_f - \boldsymbol{\mu}_g)^{\mathrm{T}}(\mathbf{\Sigma}_f + \mathbf{\Sigma}_g)^{-1}(\boldsymbol{\mu}_f - \boldsymbol{\mu}_g) + \frac{1}{2}\ln\frac{|\frac{\mathbf{\Sigma}_f+\mathbf{\Sigma}_g}{2}|}{\sqrt{|\mathbf{\Sigma}_f||\mathbf{\Sigma}_g|}}, \quad (4.2)$$

where $\mathbf{\Sigma}_f$ and $\mathbf{\Sigma}_g$ are the covariances of multivariate feature distributions f and g, respectively. $\ln$ is the natural logarithm. It should be noted that the Bhattacharyya distance reduces to the Mahalanobis distance (up to a constant summation term) if $\mathbf{\Sigma}_f = \mathbf{\Sigma}_g$.

Especially in speech segmentation and clustering, the Bayesian information criterion (BIC) has been used as a distance measure [86]. The BIC difference is defined as

$$\begin{aligned}\Delta\mathrm{BIC} = {} & (N_{fr}^A + N_{fr}^B)\log_{10}(|\mathbf{\Sigma}_{AB}| - N_{fr}^A\log_{10}(|\mathbf{\Sigma}_A|) - N_{fr}^B\log_{10}(|\mathbf{\Sigma}_B|) \\ & - \frac{1}{2}\lambda(N_f + \frac{1}{2}N_f(N_f+1))\log_{10}(N_{fr}^A + N_{fr}^B),\end{aligned} \quad (4.3)$$

where N_{fr}^A and N_{fr}^B are the numbers of observations in sample A and B, respectively. $\mathbf{\Sigma}_A$, $\mathbf{\Sigma}_B$, and $\mathbf{\Sigma}_{AB}$ are the covariances of sample A, sample B, and the concatenation of these samples, respectively. N_f is the length of the feature vector and λ is a penalty factor to compensate for small sample sizes.

The similarity of two samples is estimated here using distances. That is, the smaller the distances, the more similar the samples are. Features define the perspective sense in which the samples are similar.

4.1 Probability Distribution Based Similarity Measures

In [P4] and [P8], we proposed similarity measures based on PDFs of the frame-wise acoustic features. The problem when using PDFs is that values of audio features are normally continuous and thus their PDF cannot be calculated exactly from a finite number of observations.

One possibility to overcome this is to quantize the frame-wise feature-vectors and generate feature histograms. Kashino *et al.* [35] proposed such an approach to detect specific audio events within a long input signal. In [P3] we applied a similar approach to the QBE task. First, the Linde-Buzo-Gray vector quantization algorithm [49] was applied to find the centers of the quantization levels. Second, the amount of frame-wise features falling into each level was calculated resulting in feature histograms for each sample. Finally, the distance between histograms can be calculated using, for example, $\mathcal{L}_1$- or $\mathcal{L}_2$-norm.

However, based on simulations presented in [P8] and summarized in Chap. 6, more accurate results were obtained when the PDF was modeled using GMMs or HMMs and the distance measures between these models are found.

4.1.1 Gaussian Mixture Model

GMM is a weighted sum of Gaussian components which is used to model the distribution $p(\mathbf{x})$ of the features of the sample. It is defined as

$$p(\mathbf{x}) = \sum_{i=1}^{I} w_i \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{4.4}$$

where w_i is the non-negative weight of the i^{th} Gaussian component. The sum of the weights $i = 1, \ldots, I$ is 1. I is the number of components, and

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{N_f/2}\sqrt{|\boldsymbol{\Sigma}_i|}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^{\mathrm{T}} \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)\right] \tag{4.5}$$

is the multivariate normal distribution with mean vector $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$. The parameters of GMMs can be estimated using, for example, the expectation maximization (EM) [16] algorithm or the Parzen window [18, pp. 164-174] approach which assigns a GMM component with a fixed variance for each observation. As a result, I is the number of frames, $\boldsymbol{\mu}_i$ is the feature vector of frame i, and $w_i = 1/I$ (for $\forall i$).

Hidden Markov Models (HMMs) [63] can also be used to consider temporal evolution of the signal. HMMs are widely used, for example, in speech recognition systems for modeling the units [23]. The HMM can move from one state to another between frames. Emission probabilities of the states are modeled with GMMs and the state transitions are controlled by state transition probabilities. The Baum-Welch algorithm [4], which is a special version of the EM algorithm, can be used to estimate the HMM parameters. For solving the most likely state sequence, we used the Viterbi algorithm [20].

4.1.2 Maximum Likelihood Estimation

The most traditional method for estimating the similarity of samples, when operating with continuous PDFs , is to generate a model (e.g., GMM) and calculate the likelihood that the database sample is generated by this model [61]. We developed a modification of this algorithm in [P3], which is illustrated in Fig. 4.1. First we generated a universal background model using the whole database and another model for the example signal, which represents the general audio sample. Then, the likelihood for each database sample to be generated by these two models is estimated through the Viterbi algorithm. If the likelihood for a sample being generated by the example model is higher than the likelihood for the general model, these samples are considered to be similar.

The computational cost of the universal background model is very high depending, of course, about the size of the database. However, the model can be generated offline and it has to be generated only once for a certain database. On the other hand, generating the model for a single sample is rather fast: for 10 second sample, with 4 states and 8 components in each state, took about 2 second. The likelihood estimation with Viterbi algorithm takes about 20 ms per sample.

The results are reported in [P3] and they are rather similar to those obtained by the histogram based method [P3]. A somewhat similar approach was applied also in audio segmentation by Velivelli *et al.* [79].

4.1.3 Kullback-Leibler Divergence

KL divergence is a distance measure between two probability distributions defined as

$$\mathrm{KL}(p_A(\mathbf{x})||p_B(\mathbf{x})) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} p_A(\mathbf{x}) \log \frac{p_A(\mathbf{x})}{p_B(\mathbf{x})} dx_1 \dots dx_N, \qquad (4.6)$$

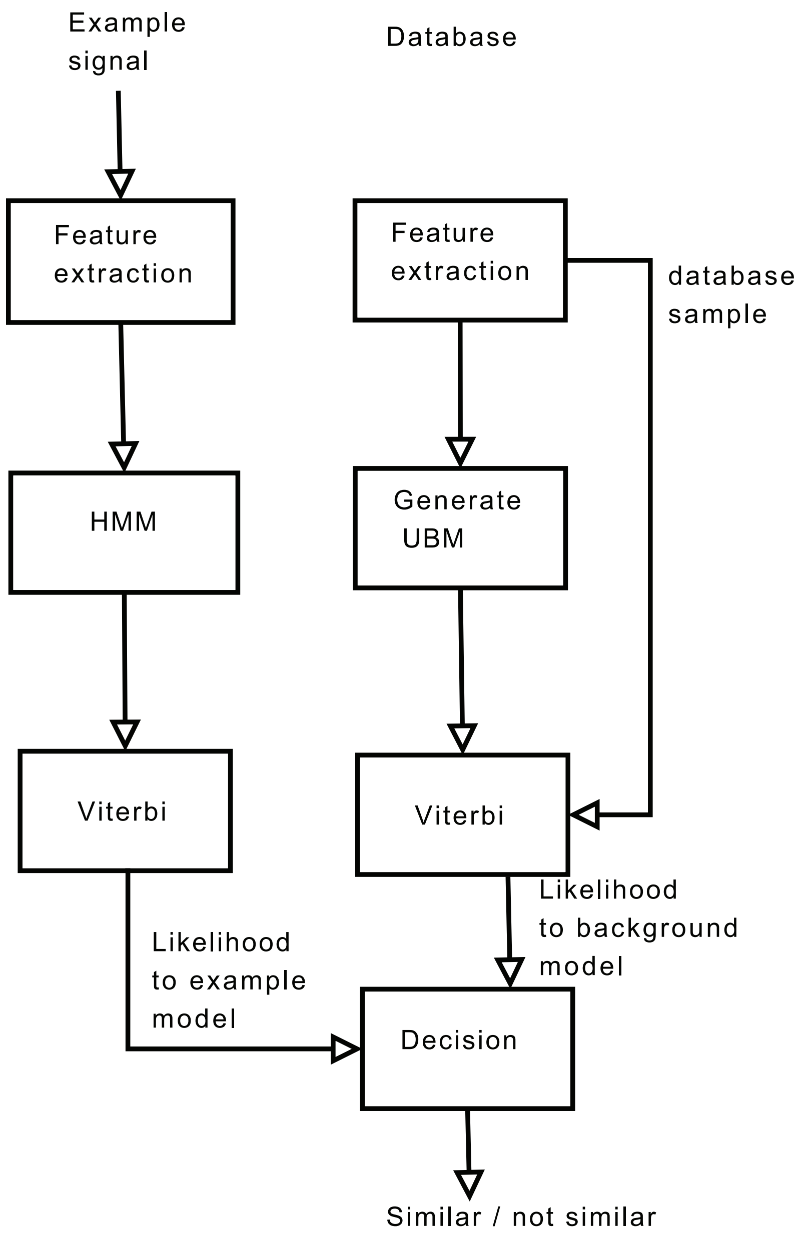

Figure 4.1: An overview of HMM based QBE system.

where $p_A(\mathbf{x})$ and $p_B(\mathbf{x})$ are the distributions of **A** and **B**, respectively. The measure is non-symmetric, but it can easily be made symmetric by adding the term $\mathrm{KL}(p_B(\mathbf{x})||p_A(\mathbf{x}))$.

Furthermore, the KL divergence between two Gaussian distributions is defined as [24]

$$\begin{aligned}\mathrm{KL}(p_A(\mathbf{x})||p_B(\mathbf{x})) = \frac{1}{2}[\log \frac{|\mathbf{\Sigma}_B|}{|\mathbf{\Sigma}_A|} + \mathrm{Tr}(\mathbf{\Sigma}_B^{-1}\mathbf{\Sigma}_A) \\ + (\boldsymbol{\mu}_A - \boldsymbol{\mu}_B)^{\mathrm{T}}\mathbf{\Sigma}_B^{-1}(\boldsymbol{\mu}_A - \boldsymbol{\mu}_B) - N_f],\end{aligned} \tag{4.7}$$

where $\mathrm{Tr}(X)$ is the trace of X. $|A|$ denotes the determinant of matrix A. The problem arises when dealing with GMMs, which have several Gaussian components. There is no closed-form solution for such a case. However, there are approximations which can be used to estimate the KL divergence between GMMs.

Approximations of KL-divergence

Goldberger *et al.* [24] achieved good results on image retrieval application using the approximations for KL-divergence between GMMs. One of them was defined as

$$\mathrm{KL}_{\mathrm{Goldberger}}(p_A(\mathbf{x})||p_B(\mathbf{x})) = \sum_{i=1}^{I_A} w_i^A(\mathrm{KL}(p_A^i(\mathbf{x})||p_B^{m(i)}(\mathbf{x})) + \log \frac{w_i^A}{w_{m(i)}^B}), \tag{4.8}$$

where I_A is the number of components in GMM A, w_i^A and $p_A^i(\mathbf{x})$ are the weight and the distribution of the i^{th} component in GMM A, respectively, and

$$m(i) = \underset{j=1,\dots,I_B}{\arg\min}\{\mathrm{KL}(p_A^i(\mathbf{x})||p_B^j(\mathbf{x})) - \log(w_j^B)\}. \tag{4.9}$$

Hershey and Olsen [28] tested several approximations for KL-divergence. They observed that the Monte Carlo approximation is the most accurate, but computationally too expensive for practical applications. They concluded that from the computationally feasible approximations the variational approximation is the most accurate:

$$\mathrm{KL}_{\text{variational}}(p_A(\mathbf{x})||p_B(\mathbf{x})) = \sum_{i=1}^{I_A} w_i^A \log \frac{\sum_{n=1}^{I_A} w_n^A \exp(-\mathrm{KL}(p_A(\mathbf{x})_i||p_A(\mathbf{x})_n))}{\sum_{j=1}^{I_B} w_j^B \exp(-\mathrm{KL}(p_A(\mathbf{x})_i||p_B(\mathbf{x})_j))}. \tag{4.10}$$

We applied Goldberger's approximation and variational approximation to audio QBE. The results were reported in [P8] and are summarized in Chap. 6.

4.1.4 Euclidean Distance

In [P4] we derived a closed-form solution for the Euclidean distance between two GMMs having diagonal-covariance matrices. [P8] extended the measure to full covariance GMMs.

The squared Euclidean distance e^2 between two GMMs is calculated by integrating the squared difference of these GMMs over the whole feature space:

$$e^2 = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \left[\sum_{i=1}^{I_A} w_i^A p_A(\mathbf{x})_i - \sum_{j=1}^{I_B} w_j^B p_B(\mathbf{x})_j \right]^2 dx_1 \ldots dx_N, \tag{4.11}$$

where x_n denotes the n^{th} feature.

The derivations are given in [P8] but finally the Euclidean distance is

$$e = \sqrt{\sum_{i=1}^{I_A}\sum_{j=1}^{I_A} w_i^A w_j^A Q_{i,j,A,A} + \sum_{i=1}^{I_B}\sum_{j=1}^{I_B} w_i^B w_j^B Q_{i,j,B,B} - 2\sum_{i=1}^{I_A}\sum_{j=1}^{I_B} w_i^A w_j^B Q_{i,j,A,B}}, \tag{4.12}$$

where

$$Q_{i,j,k,n} = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} p_k(\mathbf{x})_i \, p_n(\mathbf{x})_j \, dx_1 \ldots dx_N. \tag{4.13}$$

Chapter 6 presents the experimental results, where the Euclidean distance between two GMMs is used as a distance measure in a QBE system. When measuring the retrieval accuracy, the Euclidean distance is among the top measures. Furthermore, it is a rather low complexity measure, especially when compared with likelihood ratio tests which gave slightly higher accuracy. This is the reason why this measure was chosen for use as a part of our audio managements application [P7].

4.1.5 Cross-Likelihood Ratio Test

In [P8] and [82] we proposed the cross-likelihood ratio test for QBE applications. A likelihood ratio test has previously been used in speech clustering and segmentation tasks [73, 64]. It is defined as the ratio between two hypothesis. First, the feature sequences are generated by two separate models and second, the sequences are generated by the same model:

$$\mathrm{LRT}(\mathbf{A},\mathbf{B}) = \frac{p_A(\mathbf{A})p_B(\mathbf{B})}{p_{AB}(\mathbf{A})p_{AB}(\mathbf{B})}, \tag{4.14}$$

where p_{AB} is a model trained using both feature sequences $\mathbf{A}$ and $\mathbf{B}$.

The problem with the above measure in the QBE task is that in the query phase, the model p_{AB} has to be generated on line for each database sample, resulting in prohibitive computational complexity. Thus, we used a modification of this measure referred to as cross-likelihood ratio test, which is given as

$$\mathrm{CLRT}(\mathbf{A},\mathbf{B}) = \frac{p_A(\mathbf{A})p_B(\mathbf{B})}{p_B(\mathbf{A})p_A(\mathbf{B})}. \tag{4.15}$$

This measure is computationally less expensive since it does not require models for signal combinations. Furthermore, this measure resulted in better retrieval results compared to traditional likelihood ratio test in our studies. This measure has also been used in many clustering applications because of its computational benefits [73, 84].

4.2 Non-Parametric Similarity Measure

In most data mining applications, it would be preferable to have as little user influence and interference as possible. Typically, in parameter-laden algorithms either the end-user or the algorithm developer has to choose several parameters. Such parameters are, for example, which features are being used, how the features are calculated, and which similarity measure is used. Consequently, human expectations and presumptions may have an effect on the results. This may lead to over emphasis of some properties or incorrect adjustment of parameters, which may cause the retrieval task to fail. Thus, it would be better to let the data speak for itself.

In parameter-free methods the number of user-determined parameters is ideally zero. As a consequence, data mining results are completely dependent on the data but independent of the user. Despite the seemingly simple algorithm, parameter-free methods have proven to give very good results in various different tasks, including book classification by the author [13], optical character recognition [13], building an evolutionary tree based on mitochondrial genomes [13], and in fetal heart rate tracing [14]. Keogh [36] showed that parameter-free or parameter-light algorithms are able to outperform or at least give competitive results in many tasks compared with traditional methods. However, the most advantageous aspect of parameter-free algorithms is that they do not require any specific knowledge from the user about the problem at hand and can be applied, as such, in a wide range of tasks.

Next, a non-parametric similarity measure, based on compression ratios, will be discussed. Then this similarity measure is modified to make it applicable to audio similarity [P5].

4.2.1 Normalized Compression Distance

The concept of normalized compression distance (NCD) was introduced by Cilibrasi and Vitányi [13]. The measure is based on Kolmogorov complexity $K(s)$ which is the minimal amount of information required to represent a string s. Furthermore, conditional Kolmogorov complexity $K(s_1|s_2)$ is defined as the shortest binary program to compute s_1 if s_2 is given as an auxiliary input. As a consequence, we can define the concept of information distance [7]:

$$\mathrm{ID}(s_1, s_2) = \max\{K(s_1|s_2), K(s_2|s_1)\}. \tag{4.16}$$

However, this measure does not take into account the lengths of the samples and therefore, normalization is required. The normalized information distance (NID) [47] is given as:

$$\mathrm{NID}(s_1, s_2) = \frac{\max\{K(s_1|s_2), K(s_2|s_1)\}}{\max\{K(s_1), K(s_2)\}}. \tag{4.17}$$

The drawback using NID is that Kolmogorov complexities are non-computable. However, they can be approximated using lossless compressors, which aim at finding the minimum amount of data to represent a given sample. Instead of NID, we can apply NCD [47]

$$\mathrm{NCD}(s_1, s_2) = \frac{C(s_1 s_2) - \min\{C(s_1), C(s_2)\}}{\max\{C(s_1), C(s_2)\}}, \tag{4.18}$$

where $C(s_1)$ and $C(s_2)$, are the sizes of compressed s_1 and s_2, respectively, and $C(s_1 s_2)$ is the compressed size of concatenated s_1 and s_2.

However, there are some parameters that need to be set even when using NCD. Firstly, when dealing with audio signals, there is a need to choose a signal representation format which would, preferably, retain the perceptual quality of the signal but also quantize the signal to produce identical codewords in two signals. This will be discussed in the next section. Secondly, a compression algorithm has to be chosen. The aim, however, is to approximate the Kolmogorov complexity and the algorithm having the best compression ratio gives the best approximation.

A rather similar idea is behind the earth mover's distance [66] which has also been used successfully in several areas. The earth mover's distance calculates the minimal cost of transforming the feature distribution of one sample to the feature distribution of the other sample.

Application to Audio Similarity

When applying the compression based similarity measure to audio signals, the signal presentation is an essential factor. The presentation should be such that near similar samples generate similar codewords but at the same time it should preserve the perceptual properties of the sound. Frequency components in the sound which are below or above the threshold of hearing can be removed without audible effect. Also masking phenomena are usually employed in the codecs. In simultaneous masking a louder sound hides a weaker sound which is occurring simultaneously. Forward and backward masking happens when a louder sound masks a weaker one even though the latter occurs either before or after the louder sound [60]. In [P5] we tested different signal representations for this purpose. We found that the low-bitrate audio codecs (AAC, AMR, and MP3) gave the best results in our test environment. For the second phase of the algorithm, a few different lossless codecs, including gzip (Lempel Ziv [88] and Huffman [29] coding), bzip (Burrows-Wheeler [11] transform and Huffman coding), were tested but the difference between them was negligible.

In the simulations, we noticed that higher bitrates gave worse results. This is due to the fact that, in order for lossless compression algorithms to compress the files, identical codewords are required in the two signals. When the bit rate is high there are more quantization steps, which means that the original frames need to be more similar

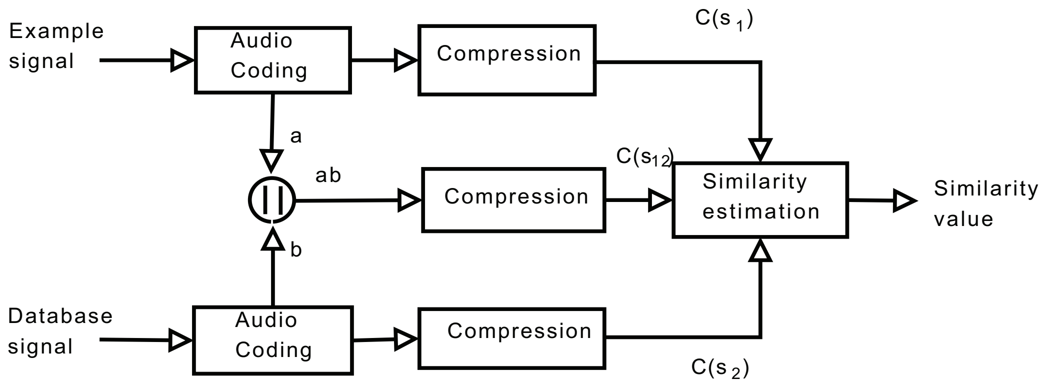

Figure 4.2: An overview of non-parametric audio similarity measure.

to each other to produce the same codeword than when lower bitrates, and thus fewer quantization levels, are used. An extreme case was to compress the audio files (wave format) without the lossy coding, which did not work at all, since there is no quantization, aside the one in wave format.

Fig. 4.2 illustrates the structure of the normalized compression distance (NCD) based similarity evaluation for audio signals.

4.3 Summary

In this chapter, several similarity measures for audio signals were reviewed. Similarity is typically calculated from the frame-wise features extracted from the original signals. The most traditional methods are based on simple statistics, but more complex measures between feature distributions were also discussed. These PDF based similarity measures preserve more information from the original signal and therefore provide more accurate results than measures from simple statistics [P4][P8]. In addition, parameter-free NCD was introduced, which gives competitive results and this is especially useful since it does not require the user to set any parameters. As a consequence, the user does not need to be a specialist and the user's expectations do not have an effect on the results.

Table 4.1 summarizes the distance measures. Second column of the table shows the computational time of a single distance calculation. It should be noted, that different distance measures require varying

Table 4.1: The properties of different similarity measures.

Method	Comp. time	Parameters
GMM cross-likelihood ratio test (8 comp.)	16.6ms	$p_A(\mathbf{A})$,$p_B(\mathbf{B})$,$p_A(\mathbf{B})$,$p_B(\mathbf{A})$
HMM cross-likelihood ratio test (8 comp.)	39.3 ms	$p_A(\mathbf{A})$,$p_B(\mathbf{B})$,$p_A(\mathbf{B})$,$p_B(\mathbf{A})$
KL-Goldberger, GMM (8 comp.)	9.30 ms	$p_A(\mathbf{x})$,$p_B(\mathbf{x})$
Euclidean distance, GMM (8 comp.)	0.87 ms	$p_A(\mathbf{x})$,$p_B(\mathbf{x})$
KL-variational, GMM (8 comp.)	20.2 ms	$p_A(\mathbf{x})$,$p_B(\mathbf{x})$
KL-Gaussian, GMM (1 comp.)	0.19 ms	$p_A(\mathbf{x})$,$p_B(\mathbf{x})$
Compression distance (AAC)	22.9 ms	$C(s_1s_2)$,$C(s_1)$,$C(s_2)$
Compression distance (AMR)	23.1 ms	$C(s_1s_2)$,$C(s_1)$,$C(s_2)$
Bhattacharyya	6.5 ms	$\boldsymbol{\mu}_f$,$\boldsymbol{\mu}_g$,$\boldsymbol{\Sigma}_f$,$\boldsymbol{\Sigma}_g$
Mahalanobis	0.013 ms	$\boldsymbol{\mu}_f$,$\boldsymbol{\mu}_g$,$\boldsymbol{\Sigma}$

amount of preprocessing. However, the preprocessing can be done offline. Third column summarizes the parameters which each measure uses for distance calculation.

Every distance measure has it's pros and cons. Mahalanobis distance is really simple and thus the computational complexity is very low. Bhattacharyya is an extension to Mahalanobis distance. Instead of using the covariance calculated over the whole database, it considers the covariances calculated from the individual samples. This is advantageous in audio query, since the feature distribution can vary a lot between different samples. The GMM and HMM based measures provide even more accurate modeling of feature distributions. KL divergence has information theoretical justifications but it is slower to compute than Euclidean distance. Furthermore, Euclidean distance has a closed form solution for multiple component GMMs when KL divergence has to be approximated if the number of GMM components is more than one. Cross-likelihood ratio tests are computationally complex, but they gave the highest accuracy in the evaluations. Likelihood ratio test has previously been used, especially, in speech segmentation tasks. The compression based distance measures are aimed for the applications which should operate without setting any parameters. The proposed method takes advantage of audio codecs, which makes it suitable for audio query tasks. However, the computational complexity is quite high.

Chapter 5

Reducing Computational Cost via Clustering

The aim of this chapter is to make the QBE application more computationally feasible. If the database is large, searching through the whole database becomes a time consuming task but the users expect results almost immediately. Nowadays, the usability of software and devices has a high priority and thus speed is an essential factor when designing a practical application.

Several methods exist in the literature to alleviate this computational burden. For example, progressive query [39], which partitions the search space into sub-sets, searches one sub-set at a time and fuses the results of the sub-set query with the previous overall retrieval results. Resulting samples can be retrieved after each sub-set query is complete. Database indexing is another method where the database is organized in a certain order as a preprocessing step in which case the database is not queried in random order. An example of the effective indexing technique is a hierarchical cellular tree [40], which is a self-organized tree and the items in the tree are partitioned offline, based on their relative distances. Here a novel method for accelerating the query is proposed [P6]. It utilizes clustering techniques in order to narrow the query to include only the most relevant clusters.

5.1 Computational Issues

In a QBE application, if optimal search accuracy is needed, the requirement is to go through the entire database and compare the example to all the database samples. Optimal accuracy means here that in all cases the database samples which have the shortest distance to the example are retrieved. This operation requires excessive computing capacity especially when it is done using mobile devices. Although, mobile devices often work as a client in such operations and the server performs the heavy calculations, the databases used nowadays may contain millions of samples and the query then becomes time-consuming even for high-performance processors. However, in practical applications users usually require fast response times even if it is at the expense of accuracy.

Fundamentally, there are two different approaches to reduce the computational complexity in multimedia retrieval [19]. The first possibility is to decrease the number of comparisons, which are made during the operation. The second is to make the single comparison faster. This is typically done by reducing the size of the feature vectors. Kanth et al. [33] achieved this by reducing the dimensionality of the feature space. Guldogan and Gabbouj [25] reduced the size of feature vectors by choosing the most relevant features from the larger feature set.

This chapter concentrates on reducing the number of comparisons during the query. The reduction is accomplished through clustering the database prior to the search. The clustering itself is a time consuming operation but the advantage is that it can be done offline and it has to be done only once for a particular database. Although, when the items are inserted into or removed from the database, the clusters will also need updating. [P6] proposes the use of transformation which reduces the dimensionality of the data in the clustering phase. This enables the use of less complex similarity measures and therefore, makes clustering faster.

5.2 Key-sample Distance Transformation

Clustering is an operation where the samples are organized into groups of samples which have similar properties. One of the most typical methods for clustering is k-means clustering [27]. The k-means algorithm iteratively explores the centroid for each cluster, such that the sum of distances between the samples and their representative cluster centroids is minimized. The clustering of database can also be considered as an unsupervised audio classification task. In classification, there are predefined classes which are used to learn the features associated with these classes in the data. Based on these associations, the algorithm generates rules which are used to define the class of new samples. In clustering task, there are no predefined classes, but instead the algorithm uses only the given data to group the samples into clusters. Samples inside each cluster has similar properties. For example, in [53] k-means clustering algorithm is used in co-operation with support vector machines for unlabeled data classification.

The k-means algorithm has previously been used also in nearest neighbor searching [19]. The samples were represented as feature vectors which were clustered. The cluster centroid having the smallest distance to query sample was then found and the search was restricted to the representative cluster. The problem, however, is that in our case each sample is represented by a series of feature vectors. As a result the k-means algorithm is not applicable as such, since the representation can not be inserted in any dimensional feature space and thus finding the centroid becomes impossible.

One solution for handling the problem would be to use some statistical measure like mean or median to reduce the input into a single feature vector. This would, however, loose a lot of information from the original sample. The other solution would be to concatenate the feature vectors. This could work if the samples have the same length, but here we are dealing with varying-length samples. The third solution is the use of a clustering algorithm which does not require the calculation of the centroids, but instead finds existing samples from the database which are then used as a cluster centers. The drawback of this approach is that the distance measures which operate on series of feature vectors tend to be complex [P8] which then leads to very long clustering time.

The other type of solution would be to make a transformation of the original sample to convert a series of feature vectors into a single feature vector. One possibility is to select k samples and calculate the distance from each database sample to these key-samples. Then the distances can be used as a k-dimensional feature vector. The question is how to choose the key-samples which are to be used as a base for the transform space? Shapiro [71] reported that the best choice of key-samples (Shapiro referred them as reference points) is to select them so that they are as far from each other as possible. In a practical situation this is difficult to accomplish as there is no information about where in the feature space, the database sample are located. Berenzweig et al. [8] used key-samples (which they referred to as anchors) for music mapping purposes. They chose key-samples from different musical genres and from different artists and the similarity between samples was estimated based on the distances to these samples. Barrington et al. used

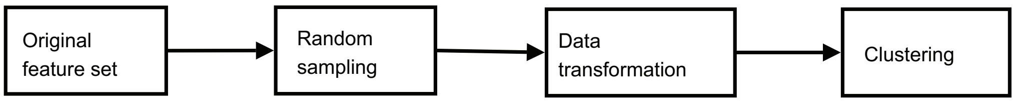

Figure 5.1: Transformation based clustering method.

a similar approach for general audio [3]. They chose key-samples from several different audio classes. Again, in a general audio database it is not always possible to choose samples that are in different parts of the space, because the database may contain only speech samples, only music samples or any other samples. This is why we decided to use random samples from the database as the key-samples. This also is not an optimal solution, but it is probable that the key-sample set includes samples from different parts of the feature space.

In [P6] we proposed using key-sample transformation as a preprocessing step to clustering based query by example. The algorithm is illustrated in Fig. 5.1. First, k samples are chosen randomly from the database to serve as a key-sample set. Then the distance from each sample in the database to these key-samples is calculated and the length k distance vector is used as a feature vector in clustering. Only k distance calculations have to be done for each sample and thus an accurate PDF based distance measure can be used. The number of key-samples determines the accuracy of the algorithm but also the computational complexity. In our simulations, $k = 10$ turned out to be a good compromise. In the clustering phase, we used k-means clustering but any other algorithm could be used instead. The transformation is formally expressed as

$$T(\mathbf{x}, \mathcal{O}, d) = \mathcal{F} \rightarrow \Re^k, \tag{5.1}$$

where $\mathbf{x}$ is the original series of feature vectors, $\mathcal{O}$ is the set of k key-samples, d is the distance measure, $\mathcal{F}$ is the original feature space, and $\Re^k$ is the k-dimensional feature space in which i^{th} element is the distance from $\mathbf{x}$ to i^{th} key-sample ($i = 1, ..., k$).

5.3 Accelerating the Query via Offline Clustering

The overall QBE system is illustrated in Fig. 5.3. In the offline phase, the features are first extracted from the database samples and the feature distribution is modeled using, for example, GMMs. Next the key-samples are chosen from the database files and distances between each sample and key-samples are calculated. Finally, the database is clustered using the k-means algorithm. When the query sample is inputted into the system, the same operations are completed. Features are extracted, the GMM is generated, and distances to key-samples are calculated. Then the resulting distance vector is compared to cluster centroids using the Euclidean distance. After this, the actual query can be restricted only to nearest cluster. The distances inside the cluster are calculated using the PDF based distance measures between the original feature distributions. The samples having the shortest distance to the example are outputted to the user.

Significant acceleration can be achieved via the clustering approach without serious accuracy loss as demonstrated in [P6]. If the clustering is completed offline, the computational time of the query is inversely proportional to the number of clusters. If the clusters are of equal size, the number of required distance calculations is

$$\mathrm{NC}_{fast} = \frac{1}{N_c}\mathrm{NC}_{full} + N_c, \tag{5.2}$$

where N_c is the number of clusters and NC_{full} is the number of distance calculations in full search. The last term is required since the nearest cluster has to be searched although it should be noted that the sizes of clusters are not usually the same, since samples are added and removed. To overcome this, in [P7] we used an update method which splits over-sized clusters and merges the small ones [19]. Figure 5.2 illustrates the speedup which is achieved in the experimental setup via the offline clustering of the database. The speedup is significant within the first ten clusters, but above that the benefit, in terms of computational time, is quite low.

The accuracy of the system can be controlled by the number of clusters into which the database is separated. Less clusters results in larger size of clusters and thus more accurate results but also longer query times. Accuracy can also be increased by expanding the search from the nearest cluster to n nearest clusters. More accurate results

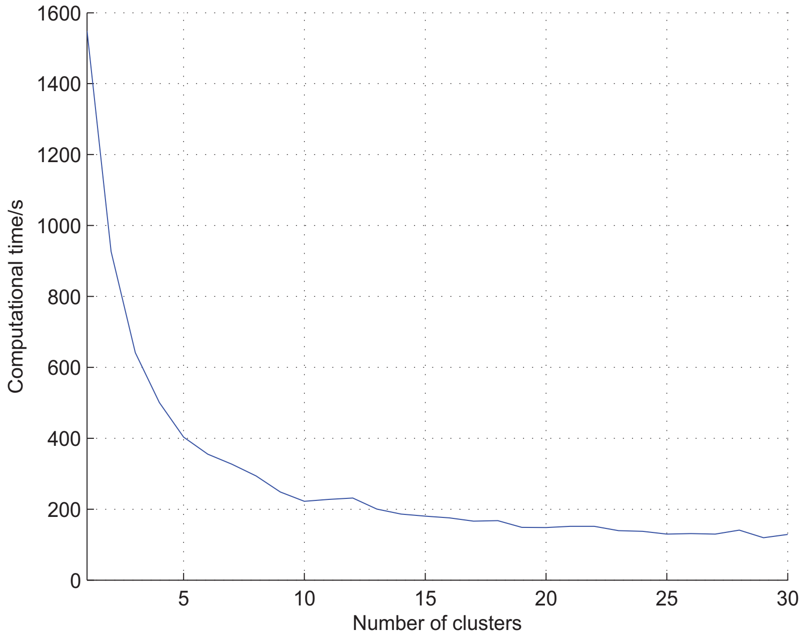

Figure 5.2: Speedup achieved via clustering the database.

are achieved but the computational cost is increased. This approach also makes it possible, in a practical application, to retrieve samples from the nearest cluster first and then to widen the search to more distant clusters. In this way the application can offer first results quickly and then make the search more accurate and complete.

The issue that is not covered in this study is the choice of how many clusters should be used. We used a fixed number of clusters but it might be advantageous to use some algorithm to choose the proper amount of clusters. In the literature there are several methods of achieving this. Aghagolzadeh *et al.* applied a top-down hierarchical clustering algorithm which begins with a large number of clusters and during iterations, removes some of the clusters [2]. They measured the information potential to define the final number of clusters. Salvador and Chan used an evaluation graph, where the x-axis is the number of clusters and the y-axis is the clustering error to search for a “knee” point in the curve [69]. Ding and He explored several methods for cluster merging and splitting in hierarchical algorithms [17].

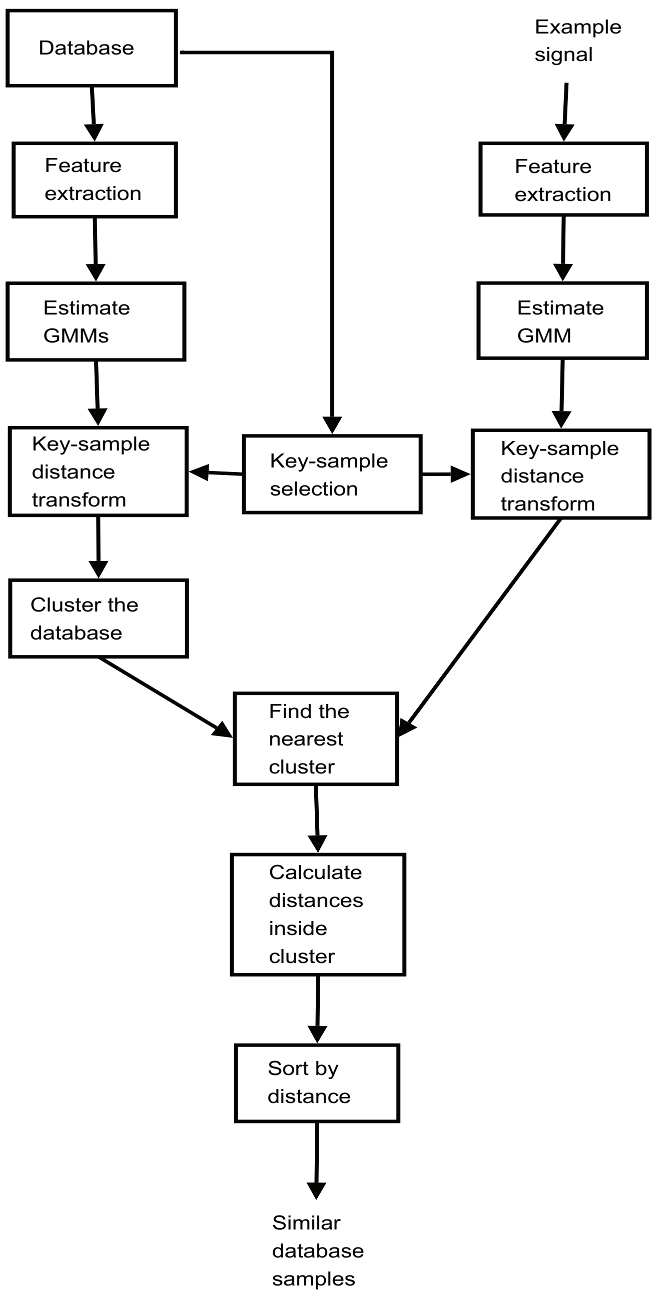

Figure 5.3: Overview of the QBE system which uses clustering for speedup.

Chapter 6

Experimental Retrieval Results

This chapter summarizes the main experimental retrieval results in the thesis. Performance is compared with other similarity measures from the literature. Firstly the evaluation procedure, the dataset, and evaluation measures are explained.

6.1 Evaluation Procedure

In the evaluation of similarity measures we used a query by example (QBE) application and "leave one out" testing. One database sample at a time was used as an example query for the rest of the database. The distance between the query and each database sample was calculated and the samples having the shortest distance were retrieved as the most similar ones. The decision between correct and false retrieval was made based on the annotations of the samples (annotation procedure is explained in [P6]). If the query and the retrieved sample were annotated into the same category, the sample was considered to be correctly retrieved, otherwise it was considered to be a false retrieval.

Two separate query methods were used: k-nearest neighbor (k-NN) query and ϵ-range query. In a k-NN query a fixed number of the most similar samples were retrieved to the user, whereas in ϵ-range query all the samples having a distance to the example shorter than a fixed threshold were retrieved. Typically a k-NN query is more practical since no matter what the example query is, it always retrieves the same number of samples. On the other hand, ϵ-range query can retrieve samples for the user as soon as similar samples are found at any time during the query process.

Features were extracted in 46 ms frames and the feature set used in the following evaluations were MFCCs (the first three coefficients), spectral spread, spectral flux, harmonic ratio, maximum autocorrelation lag, crest factor, noise likeness, total energy, and variance of instantaneous power.

6.1.1 Acoustic Material

The dataset used in QBE simulations contained 1332 samples with a 16 kHz sampling rate. The samples were from 4 main categories and 17 sub categories. The samples in the database were 10 seconds long. The categories are listed in Table 6.1.

This thesis aims at query by example application, which could be applied in any kind of audio. Thus, the database aims at being as general as possible. It contains samples from different music styles, environmental sounds, speech samples from several speakers, etc. More detailed explanation regarding the database can be found in [P5] and [P6].

6.1.2 Evaluation Measures

The performance of query by example application was measured using precision P, recall R, precision error P_e, the combination of these referred to as F-measure F, and average normalized modified retrieval rank (ANMRR) [57]. These are defined as:

$$P = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ retrieved\ items}, \tag{6.1}$$

$$R = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ relevant\ items}, \tag{6.2}$$

$$P_e = 1 - P, \tag{6.3}$$

$$F = 2RP/(R+P), \tag{6.4}$$

$$ANMRR = \frac{1}{Q}\sum_{j=1}^{Q} \frac{\sum_{i=1}^{G(j)} \frac{r(i)}{G(j)} - \frac{1}{2}(G(j)+1)}{\min(4G(j), 2max(G(j))) + 0.5 - \frac{1}{2}G(j)}, \tag{6.5}$$

Table 6.1: Audio categories and the number of samples in the database.

Main category	Sub category
Environmental (231)	Inside a car (151) In a restaurant (42) Road (38)
Music (620)	Jazz (264) Drums (56) Popular (249) Classical (51)
Singing (165)	Humming (52) Singing (60) Whistling (53)
Speech (316)	Speaker1 (50) Speaker2 (47) Speaker3 (44) Speaker4 (40) Speaker5 (47) Speaker6 (38) Speaker7 (50)

where *relevant item* means sample from the same category as the example, $G(j)$ is the number of retrieved samples from query j, Q is the number of queries, and $r(i)$, $i = 1, 2, \ldots, G(i)$ is the ranking of relevant samples retrieved (lower number indicating more similar sample) and for relevant samples missed $r(i) = \min(4G(i), 2max(G(i))) + 1$.

6.2 Simulation Results

The values in the following simulations are mean values from the results of each category, unless otherwise stated. Table 6.2 demonstrates the performance of different similarity measures in k-NN query. From the table we can observe that the distribution based similarity measures which use multiple GMM components provide low precision errors. The cross likelihood ratio test between HMMs and the Goldberger's approximation of KL divergence give the lowest precision error with a small margin relative to the others. The Mahalanobis distance is also competitive, especially since it is the quickest to calculate. The histogram method, which is actually a quantized version of a single component distribution, is the most inaccurate.

Table 6.2: The P_e for k-NN queries for the main and sub categories. The number of retrieved samples was 10.

Method	Main	Sub
Histogram	7.7 %	24.3 %
Mahalanobis	1.2 %	6.8 %
Compression distance (AMR)	2.2 %	14.6 %
Compression distance (AAC)	1.0 %	9.6 %
KL-Gaussian, GMM (1 comp.)	2.3 %	14.0 %
KL-Goldberger, GMM (8 comp.)	0.8 %	4.8 %
KL-variational, GMM (8 comp.)	1.2 %	6.0 %
Euclidean distance, GMM (8 comp.)	0.8 %	5.9 %
GMM cross-likelihood ratio test (8 comp.)	0.8 %	5.3 %
HMM cross-likelihood ratio test (8 comp.)	0.7 %	4.8 %

Table 6.3 presents F-measure and ANMRR values for each method. F-measure was calculated from the ϵ-range query with different values of ϵ. The maximum value is reported in the table. The ANMRR value was calculated from k-NN query with k=10. Both quality measures provide rather similar rank for the distance measures. In F-measure the higher value corresponds to a better quality, whereas in ANMRR, a lower value corresponds to a higher quality. The exception are the compression based distance measures which give high F-measure and low ANMRR compared to other measures.

Fig. 6.1 presents the precision and recall in ϵ-range queries with different values of ϵ. In most parts of the figure, the cross-likelihood ratio test between GMMs produced the highest accuracy. However, in low recall rates, which refers to small ϵ, the cross-likelihood ratio test using HMMs, the Euclidean distance, and the Mahalanobis distance had the highest precision.

Table 6.3: Maximum F-measure and ANMRR values for different methods in sub categories.

Method	F-measure	ANMRR
GMM cross-likelihood ratio test (8 comp.)	58.3	0.049
HMM cross-likelihood ratio test (8 comp.)	52.1	0.044
KL-Goldberger, GMM (8 comp.)	49.3	0.045
Euclidean distance, GMM (8 comp.)	47.7	0.055
Mahalanobis	53.3	0.065
KL-variational, GMM (8 comp.)	45.7	0.056
KL-Gaussian, GMM (1 comp.)	40.1	0.133
Compression distance (AAC)	59.1	0.090
Compression distance (AMR)	50.1	0.141
Histogram	35.4	0.236

Fig. 6.2 illustrates the precision of k-NN queries when k was varied from 1 to 35. Here we can observe, that when k was large (k>28) the Euclidean distance was the most accurate measure. When the k was small (k<10) the likelihood ratio using HMMs gives highest precision and in the mid section (10<k<20) Goldberger's approximation of the KL divergence and likelihood-ratio test using HMMs are the best in terms of precision. Thus, it can be stated that none of the similarity measures is clearly the most accurate. The most suitable measure depends, instead, about the task.

We can also observe that PDF based measures, generally, give higher precision compared to compression based measures. However, the results of compression based measure using AAC as a signal representation gives promising results considering that this measure does not require any feature extraction or other parameters.

Similar results, for the PDF based and reference measures, from the simulations, where the query is made to the whole database instead of "leave one out", is illustrated in Fig. 6.3. The natural consequence is that the results are slightly better, since the query sample is now included to the database.

Table 6.4 demonstrates the confusion matrix of the proposed query by example algorithm. Here the Euclidean distance was used as a distance measure and 10 nearest samples were retrieved from the database. The values in the matrix are the percentage of the signals retrieved from each category (rows) when the example was from the certain category (columns). The most confusion was between the music

Table 6.4: Confusion matrix for Euclidean distance when 10 nearest neighbors were retrieved. The values in the matrix are the percentage of the signals retrieved from each category (rows) when the example was from the certain category (columns).

	Inside a car	In a restaurant	Road	Jazz	Drums	Popular	Classical	Humming	Singing	Whistling	Speaker1	Speaker2	Speaker3	Speaker4	Speaker5	Speaker6	Speaker7
Inside a car	**99.7**	1.9	7.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
In a restaurant	0	**97.9**	1.1	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0
Road	0.2	0	**91.8**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Jazz	0.1	0.2	0	**91.3**	0	11.5	11.2	5.7	4.2	0.4	0	0	0	0	0	0	0
Drums	0	0	0	0	**99.5**	0	0	0	0	0	0	0	0	0	0	0	0
Popular	0	0	0	7.3	0.2	**88.4**	5.7	0	0.2	0.9	0	0	0	0	0	0	0
Classical	0	0	0	0.5	0	0	**82.8**	0	0	0.4	0	0	0	0	0	0	0
Humming	0	0	0	0.4	0	0	0.4	**90.8**	3.8	0.5	0	0	0	0	0	0	0
Singing	0	0	0	0.4	0	0.1	0	3.3	**91.8**	0.4	0	0	0	0	0	0	0
Whistling	0	0	0	0	0.3	0	0	0	0	**95.3**	0	0	0	0	0	0	0
Speaker1	0	0	0	0	0	0	0	0	0	0	**100**	0	0	0	0	0	0
Speaker2	0	0	0	0	0	0	0	0	0	0	0	**99.8**	2.5	0	0	0	0
Speaker3	0	0	0	0	0	0	0	0	0	0	0	0.2	**97.1**	0	0	0.5	0
Speaker4	0	0	0	0	0	0	0	0	0	0	0	0	0	**100**	0	0	0
Speaker5	0	0	0	0	0	0	0	0.2	0	0	0	0	0	0	**100**	0	0
Speaker6	0	0	0	0	0	0	0	0	0	0	0	0	0.4	0	0	**99.5**	0
Speaker7	0	0	0	0	0	0	0	0	0	2.1	0	0	0	0	0	0	**100**

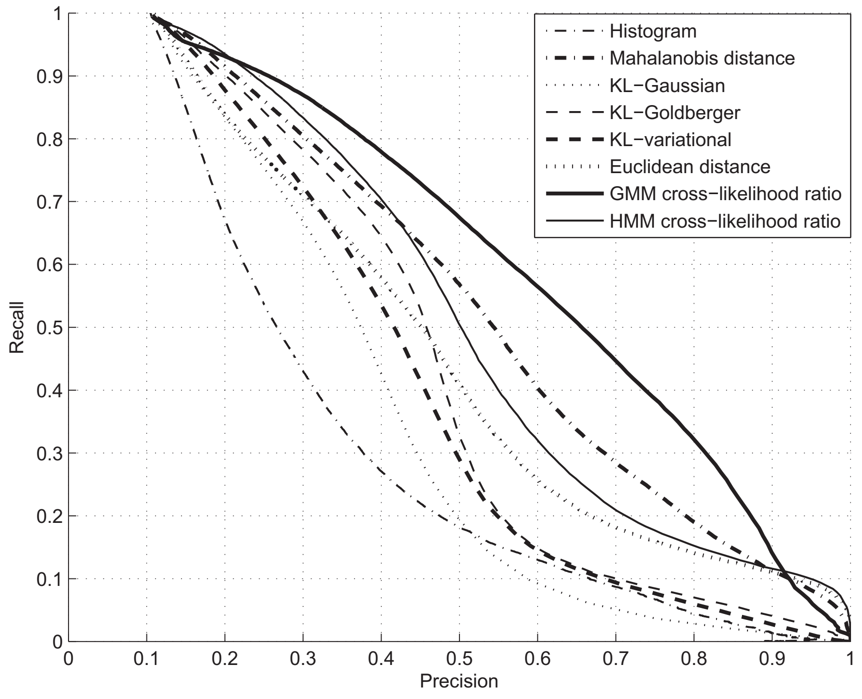

Figure 6.1: ϵ-range query results when the value of ϵ is changed.

sub categories, especially with jazz, popular, and classical music. However, these categories contain samples which were close to each other also from the human perspective. On the other hand, the speakers were separated from each other almost perfectly.

6.2.1 Fast Search

Chapter 5 introduced a clustering method to improve the speed of the QBE. The drawback of the method is that the accuracy is reduced since the query example is not compared with all the samples in the database. However, simulations reveal that the accuracy loss is only few percentage units compared to full search. Euclidean distance between PDFs was used here as distance measure when searching inside the clusters.

Fig. 6.4 illustrates the effect of changing the number of key-samples. The number of clusters in this test was 17 and the 5 nearest

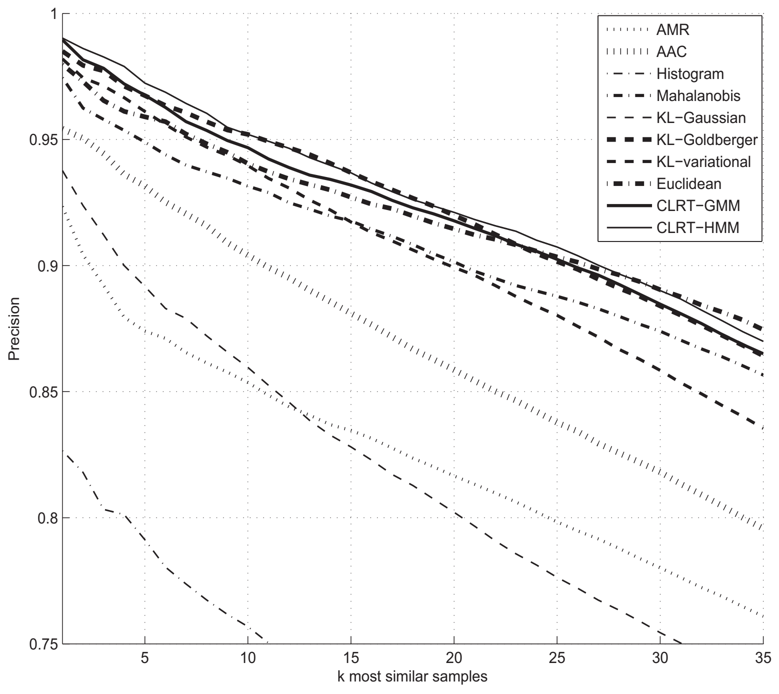

Figure 6.2: Results for different similarity measures with k-NN queries. k is varied between 1 and 35.

neighbors were retrieved. It can be seen that the higher key-sample number gives higher precision. However, the improvement in precision above 10 key-samples is quite small and thus, in the other simulations we used 10 as the number of key-samples since the number of key samples should be as low as possible as the extra samples make the search and especially the clustering take more time. As can be seen from the figure, even when using all the samples as key-samples the algorithm does not reach the same precision as the full search. This is due to the fact that the clustering is still being made using less accurate similarity measure, as discussed in Chapter 5. In the following simulations the Euclidean distance between the PDFs is used as a similarity measure, when searching inside the nearest cluster.

In Fig. 6.5 the effect of changing the number of overall clusters and the effect of searching similar samples from several nearest clusters, was tested. 10 key-samples were used and the 5 nearest neighbors

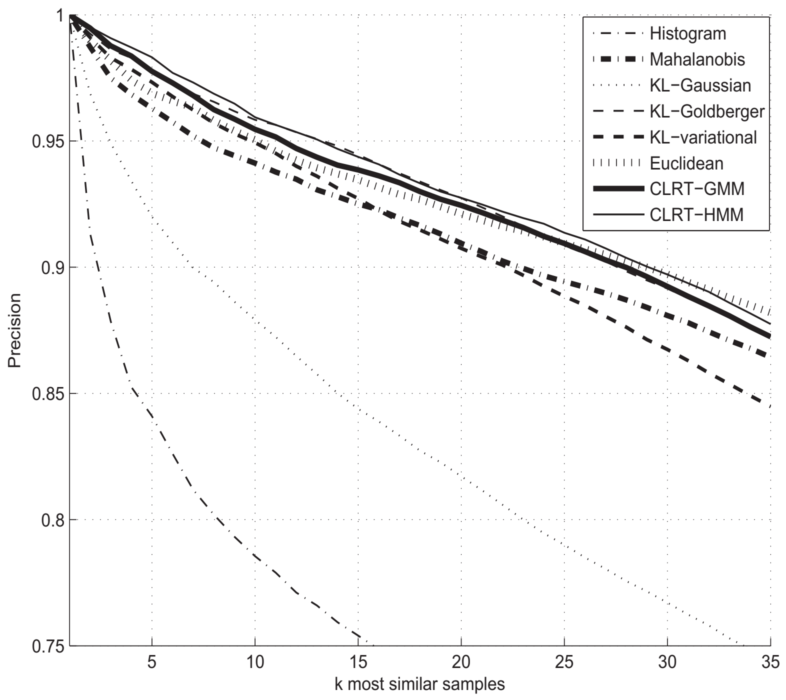

Figure 6.3: Results for different similarity measures in k-NN query. k is changed from 1 to 35. The query is made from the whole database.

were retrieved. The query precision degrades more or less linearly as the number of overall clusters increases. However, the query speed accelerates at the same time, thus the number of clusters is again a compromise between accuracy and speed. If the number of clusters in the original dataset is known beforehand, then this would be a good choice. However, usually this information is not available. In our database there were samples from 17 clusters, which explains the rapid drop in precision after 17 clusters, when the search is done only in the nearest cluster.

Different curves in Fig. 6.5 illustrate the effect of widening the query to the two or three nearest clusters. The top curve is the full search, which corresponds to the situation where the query is run over the whole database, i.e. using all the clusters. The next is searching the three nearest clusters, then searching from the two nearest clusters, and finally searching only from the nearest cluster. Expanding

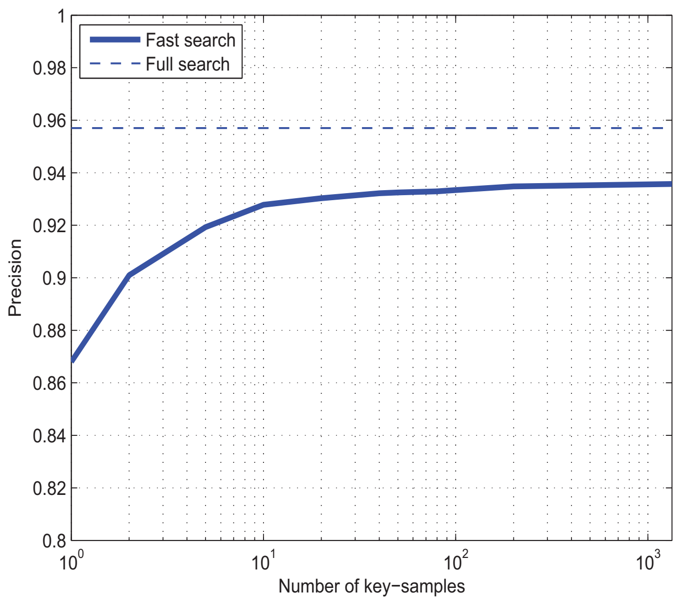

Figure 6.4: Precision performance with different numbers of key-samples.

the search into a few of the nearest clusters turned out to be very effective way of increasing the accuracy of the query. When searching three nearest clusters, the precision is very close to that of a full search even when the number of clusters is as high as 50. It can be seen that searching in the two nearest clusters instead of only one is more effective than decreasing the overall number of clusters.

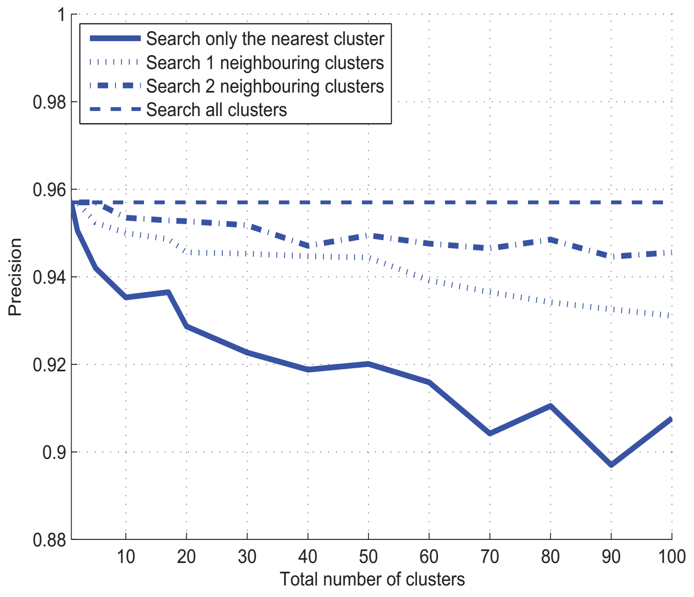

Figure 6.5: The system performance with different numbers of clusters.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Similarity measures and signal representations proposed in this thesis will aid the development of ever more accurate audio retrieval applications. Already, retrieval tools are useful in both professional and commercial search engines, but they are still far from perfect. Similarity, as a concept, is problematic when dealing with general audio samples. This is due to the fact that when based on only one example signal and without any additional information, it is impossible to know which characteristics of the sample are of interest.

This thesis proposed a method which can separate drums from the polyphonic music signal. The algorithm separates the original sound into components using NMF and finds the components belonging to the drums using a SVM. The sound source separation is an important preprocessing step for further audio processing. The thesis also proposed a parametric representation for harmonic sounds, which is based on MFCCs and ADSR representation. It preserves the most important characteristics of the signal while representing it in a highly compact form.

Most importantly, this thesis demonstrates that the similarity of audio samples can be estimated effectively using PDFs . Several distance measures between PDFs were proposed for estimating the similarity between audio samples. These were compared against traditional statistical measures in query by example task and were found to provide more accurate results although they are, of course, more complex. A parameter-free similarity measure was also proposed for audio samples. The sample is first represented using a perceptual lossy audio codec and then similarity is measured based on lossless compression ratios. For practical audio retrieval applications, the clustering approach was proposed. This reduces the number of samples from among which the similar samples are to be selected. This is an important procedure in modern databases which may contain millions of samples.

7.2 Discussion and Future Work

Although, the performance of the proposed query by example application was already good, there exist several possibilities to improve audio retrieval systems, which were not covered in this thesis.

- Utilizing feedback from the user. It would be profitable to take either positive or negative feedback about the retrieval results. Users could, for example, indicate that certain samples were exactly what he/she was looking for or that certain samples were completely wrong. Using this information the system could learn and provide better results in the future.

- Finding better methods to choose the key-samples in key-sample transformation and the number of clusters. In our implementation the key-samples are chosen randomly, since the system was targeted for an application which does not have any information about the content of database. However, random sampling may not be the best solution. The number of clusters is set manually. Automatic methods for finding the number of clusters [2, 17, 69] would be useful, since the number of clusters is not always known.

- The usability of audio retrieval applications, and especially browsing in audio databases. Browsing is usually based on file names or some tags, but more innovative approaches are needed. Stewart *et al.* [74] proposed one such method, in which the user can browse through the database in two or three dimensional space based mostly on audio information. Such innovative usability improvements are also required for audio retrieval applications.

Bibliography

[1] S. A. Abdallah and M. D. Plumbley. An independent component analysis approach to automatic music transcription. In *Proc. 114th AES Convention*, Amsterdam, The Netherlands, Mar. 2003.

[2] M. Aghagolzadeh, H. Soltanian-Zadeh, B. N. Araabi, and A. Aghagolzadeh. Finding the number of clusters in a dataset using an information theoretic hierarchical algorithm. In *Proc. IEEE International Conference on Electronics, Circuits and Systems (ICECS 2006)*, volume 2, pages 693–696, New York, USA, July 2000.

[3] L. Barrington, A. Chan, D. Turnbull, and G. Lanckriet. Audio information retrieval using semantic similarity. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, volume 2, pages 725–728, Honolulu, Hawaii, USA, Apr. 2007.

[4] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, July 1970.

[5] J. G. Beerends and J. A. Stemerdink. A perceptual audio quality measure based on a psychoacoustic sound representation. *J. Audio Eng. Soc.*, 40:963–978, Dec. 1992.

[6] P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. In *Proc. Fourth European Conference on Computer Vision*, volume 1, pages 45–58, Cambridge, UK, Apr. 1996.

[7] C. H. Bennett, P. Gács, M. Li, P. M. B. Vitányi, and W. H. Zurek. Information distance. *IEEE Transactions on Information Theory*, 44(4):1407–1423, July 1998.

[8] A. Berenzweig, D. P. W. Ellis, and S. Lawrence. Anchor space for classification and similarity measurement of music. In *Proc. International Conference on Multimedia and Expo (ICME 2003)*, pages 29–32, 2003.

[9] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by probability distributions. *Bull. Calcutta Math. Soc*, 35:99–109, 1943.

[10] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fielder, K. Akagiri, H. Fuchs, and M. Dietz. Iso/iec mpeg-2 advanced audio coding. *Journal of Audio Engineering Society*, 45(10):789–814, Oct. 1997.

[11] M. Burrows and W. D. J. A block-sorting lossless data compression algorithm. Technical Report 124, Digital Equipment Corporation, 1994.

[12] M. A. Casey and A. Westner. Separation of mixed audio sources by independent subspace analysis. In *Proc. International Computer Music Conference*, Berlin, Germany, 2000.

[13] R. Cilibrasi and P. M. B. Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, Apr. 2005.

[14] C. Costa-Santos, J. Bernandes, P. M. B. Vitányi, and L. Antunes. Clustering fetal heart rate tracings by compression. In *Proc. 19th IEEE International Symposium on Computer-Based Medical Systems*, Salt Lake City, Utah, USA, June 2006.

[15] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuosly spoken sentences. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-28(4):357–366, 1980.

[16] A. P. Dempster and D. B. B. Laird, N. M. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39(1):1–38, Nov. 1977.

[17] C. Ding and X. He. Cluster merging and splitting in hierarchical clustering algorithms. In *Proc. IEEE International Conference on*

Data Mining (ICDM 2002), pages 139–146, Maebashi City, Japan, Dec. 2002.

[18] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., second edition, 2001.

[19] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. El Abbadi. Approximate nearest neighbor searching in multimedia databases. In *Proc. 17th IEEE International Conference on Data Engineering (ICDE)*, pages 503–511, Heidelberg, Germany, Apr. 2001.

[20] G. D. Forney. The Viterbi algorith. *Proceedings of the IEEE*, 61(3):268–278, Mar. 1973.

[21] M. Gabbouj, E. Guldogan, M. Partio, O. Guldogan, M. Birinzi, A. Iftikhar, S. Uhlmann, and S. Kiranyaz. An extended framework structure in MUVIS for content-based multimedia indexing and retrieval. In *Proc. 4th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications (SETIT 2007)*, Tunisia, Mar. 2007.

[22] O. Gillet and G. Richard. Transcription and separation of drum signals from polyphonic music. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3), Mar. 2008.

[23] M. Giurgiu. Maximization of mutual information for training hidden Markov models in speech recognition. In *Proc. Third COST 276 Workshop on Information and Knowledge Management for Integrated Media Communications*, Budapest, Hungary, Oct. 2002.

[24] J. Goldberger, S. Gordon, and H. Greenspan. An efficient image similarity measure based on approximations of KL-divergence between two Gaussian mixtures. In *Proc. 9th IEEE International Conference on Computer Vision*, pages 487–493, Nice, France, Oct. 2003.

[25] E. Guldogan and M. Gabbouj. Feature selection for content-based image retrieval. *Springer Journal on Signal, Image and Video Processing*, 2(3):241–250, Sept. 2008.

[26] C. J. C. Gurges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[27] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979.

[28] J. Hershey and P. Olsen. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2007)*, Honolulu, Hawaii, USA, Apr. 2007.

[29] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the I.R.E.*, 40(9):1098–1101, sep 1952.

[30] A. Hyvärinen. Beyond independent components. In *Proc. Ninth International Conference on Artificial Neural Networks (ICANN 1999)*, volume 2, pages 809–814, Edinburgh, UK, Sept. 1999.

[31] A. Iftikhar and M. Gabbouj. *Audio-Visual Multimedia Retrieval on Mobile Devices. Multimedia Services in Intelligent Environments*. G. A. Tsihrintzis and L. Jain (ed.), Springer Berlin, 1008.

[32] K. Jensen. *Timbre Models of Musical Sounds*. PhD thesis, Department of Computer Science, University of Copenhagen, 1999.

[33] K. Kanth, R., D. agrawal, and A. Singh. Dimensionality reduction for similarity searching in dynamic databases. In *Proc. ACM SIGMOD International Conference on Management of Data*, volume 27, pages 166–176, Seattle, WA, USA, June 1998.

[34] A. Kapur, M. Benning, and G. Tzanetakis. Query-by-beat-boxing: Music retrieval for the DJ. In *Proc. 5th International Conference on Music Information Retrieval*, Oct. 2004.

[35] K. Kashino, T. Kurozumi, and H. Murase. A quick search method for audio and video signals based on histogram pruning. *IEEE Transactions on Multimedia*, 5(3), Sept. 2003.

[36] E. Keogh, S. Lonardi, and C. Ratanamahatana. Towards parameter-free data mining. In *Proc. 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 206–215, Seattle, WA, USA, Aug. 2004.

[37] H.-G. Kim, N. Moreau, and T. Sikora. *MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval*. John Wiley & Sons, 2005.

[38] S. Kiranyaz. *Advanced Techniques for Content-Based Management of Multimedia Databases*. PhD thesis, Tampere University of Technology, Finland, 2005.

[39] S. Kiranyaz and M. Gabbouj. A novel multimedia retrieval technique: Progressive query (why wait?). *Proc. IEE Proceedings Vision, Image and Signal Processing*, 152(3):356–366, May 2005.

[40] S. Kiranyaz and M. Gabbouj. Hierarchical cellular tree: An efficient indexing method for content-based retrieval on multimedia databases. *Proc. IEEE Transactions on Multimedia*, 9(1):102–119, Jan. 2007.

[41] S. Kiranyaz, A. F. Qureshi, and M. Gabbouj. A generic audio classification and segmentation approach for multimedia indexing and retrieval. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(3), May 2006.

[42] A. Klapuri, T. Virtanen, and M. Helén. Modeling musical sounds with an interpolating state model. In *Proc. 13th European Signal Processing Conference (EUSIPCO 2005)*, Sept. 2005.

[43] A. P. Klapuri. Sound onset detection by applying psychoacoustic knowledge. In *Proc. 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing*, June 1999.

[44] Last.fm. http://www.last.fm.

[45] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Proc. Advances in Neural Information Processing 13 (NIPS 2000)*, pages 556–562, 2000.

[46] J. Lehikoinen, A. Aaltonen, P. Huuskonen, and I. Salminen. *Personal content experience: Managing digital life in the mobile age*. London: Wiley & Sons, 2007.

[47] M. Li, X. Chen, X. Li, B. Ma, and P. M. B. Vitányi. The similarity metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004.

[48] Y. Li, S.-H. Lee, C.-H. Yeh, and C.-C. J. Kuo. Techniques for movie content analysis and skimming: tutorial and overview on video abstraction techniques. *IEEE Signal Processing Magazine*, 23(2):79–89, 2006.

[49] Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, COM-28:84–95, Jan. 1980.

[50] L. Lu, H. You, and H.-J. Zhang. A new approach to query by humming in music retrieval. In *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, pages 595–598, Aug. 2001.

[51] L. Lu, H.-J. Zhang, and H. Jiang. Content analysis for audio classification and segmentation. *IEEE Transactions on Speech and Audio Processing*, 10:504–516, Oct. 2002.

[52] M. Mandel and D. Ellis. Song-level features and support vector machines for music classification. In *Proc. 6th International Conference on Music Information Retrieval*, Sept. 2005.

[53] L. Maokuan, C. Yusheng, and Z. Honghai. Unlabeled data classification via support vector machines and k-means clustering. In *Proc. International Conference on Computer Graphics, Imaging and Visualization (CGIV 2004)*, pages 183–186, July 2004.

[54] Midomi. http://www.midomi.com.

[55] A. Moreau and A. Flexer. Drum transcription in polyphonic music using non-negative matrix factorisation. In *Proc. 8th International Conference on Music Information Retrieval (ISMIR2007)*, Vienna, Austria, Sept. 2007.

[56] MPEG. http://www.mpeg.org.

[57] MPEG-7 visual part of experimentation model version 2.0. MPEG-7 Output Document ISO/MPEG, dec 1999.

[58] Musipedia. http://www.musipedia.org.

[59] Muvis. http://muvis.cs.tut.fi.

[60] T. Painter and A. Spanias. Perceptual coding of digital audio. *Proceedings of the IEEE*, 88(4):451–515, Apr. 2000.

[61] E. Pampalk. *Computational Models of Music Similarity and their Applications in Music Information Retrieval*. PhD thesis, Technische Universitat, Wien, 2006.

[62] G. Peeters. A large set of audio features for sound description (similarity and classification) in the cuidado project. Technical report, CUIDADO I.S.T. Project, was accessed in http://recherche.ircam.fr/equipes/analyse-synthese/peeters/ARTICLES/, 2004.

[63] L. Rabiner. A tutorial on hidden markov models and selected applications inspeech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb. 1989.

[64] D. A. Reynolds, E. Singer, B. A. Carlson, G. C. O'Leary, J. J. McLaughlin, and M. A. Zissman. Blind clustering of speech utterances based on speaker and language characteristics. In *Proc. 5th International Conference on Spoken Language Processing (ICSLP98)*, Sydney, Australia, Dec. 1998.

[65] X. Rodet. Musical sound signal analysis/synthesis: Sinusoidal+residual and elementary waveform models. In *Proc. IEEE Time-Frequency and Time-Scale Workshop*, Coventry, Grande Bretagne, 1997.

[66] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2), Nov. 2000.

[67] M. Ryynänen, T. Virtanen, J. Paulus, and A. Klapuri. Accompaniment separation and karaoke application based on automatic melody transcription. In *Proc. IEEE International Conference on Multimedia and Expo (ICME'08)*, Hannover, Germany, June 2008.

[68] R. Salami, B. Bessette, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Jarvinen. The adaptive multi-rate wideband codec: history and performance. In *Proc. IEEE Workshop on Speech Coding*, pages 144–146, Montreal, Quebec, Canada, Oct. 2002.

[69] S. Salvador and P. Chan. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In *Proc. IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)*, pages 576–584, Boca Raton, Florida, USA, Nov. 2004.

[70] E. Scheirer and M. Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP97)*, volume 2, pages 1331–1334, 1997.

[71] M. Shapiro. The choice of reference points in best-match file searching. *Communications of the ACM*, 20(5):339–343, 1977.

[72] J. Song, S.-Y. Bae, and K. Yoon. Query by humming: Matching humming query to polyphonic audio. In *Proc. IEEE International Conference on Multimedia and Expo (ICME'02)*, pages 329–332, Aug. 2002.

[73] T. Stadelmann and B. Freisleben. Fast and robust speaker clustering using the earth mover's distance and mixmax models. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2006)*, Toulouse, France, May 2006.

[74] R. Stewart, M. Levy, and M. Sandler. 3d interactive environment for music collection navigation. In *Proc. 1th International Conference on Digital Audio Effects (DAFx-08)*, pages 13–17, Espoo, Finland, Sept. 2008.

[75] Z. Su, H. Zhang, S. Li, and S. Ma. Relevance feedback in content-based image retrieval: Bayesian framework, feature subspaces, and progressive learning. *IEEE Transactions on Image Processing*, 12(8):924–937, Aug. 2003.

[76] T. Tolonen and M. Karjalainen. A computationally efficient multipitch analyses model. *IEEE Transactions on Speech and Audio Processing*, 8(6):708–716, Nov. 2000.

[77] M. Turk. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[78] C. Uhle, C. Dittmar, and T. Sporer. Extraction of drum tracks from polyphonic music using independent subspace analysis. In *Proc. 4th International Symposium on Independent Component Analysis and Blind Signal Separation (ICA2003)*, Nara, Japan, Apr. 2003.

[79] A. Velivelli, C. Zhai, and T. Huang. Audio segment retrieval using a short duration example query. In *Proc. IEEE International Conference on Multimedia and Expo (ICME'04)*, pages 27–30, Taipei, Taiwan, June 2004.

[80] E. Vincent and X. Rodet. Music transcription with isa and hmm. In *Proc. 5th International Symposium on ICA and BSS (ICA'04)*, pages 59–63, Granada, Spain, 2004.

[81] T. Virtanen. Sound source separation using sparse coding with temporal continuity objective. In *Proc. International Computer Music Conference*, Singapore, 2003.

[82] T. Virtanen and M. Helén. Probabilistic model based similarity measures for audio query-by-example. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA2007)*, New Paltz, New York, Oct. 2007.

[83] M. Yang and N. Bourbakis. An overview of lossless digital image compression techniques. In *Proc. 48th Midwest Symposium on Circuits and Systems*, volume 2, pages 1099–1102, Cincinnati, Ohio, USA, Aug. 2005.

[84] J. Yin and Q. Yang. Integrating hidden Markov models and spectral analysis for sensory time series clustering. In *Proc. 5th IEEE International Conference on Data Mining (ICDM 2005)*, pages 506–513, Houston, Texas, USA, Nov. 2005.

[85] T. Zhang and C.-C. J. Kuo. Audio content analysis for online audiovisual data segmentation and classification. *IEEE Transactions on Speech and Audio Processing*, 9(4):441–457, 2001.

[86] B. Zhou and J. H. L. Hansen. Unsupervised audio stream segmentation and. clustering via the Bayesian information criterion. In *Proc. Int. Conf. on Spoken Language Processing (ICSLP-2000)*, volume 3, pages 714–717, Beijing, China, Oct. 2000.

[87] X. S. Zhou and T. S. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, 8(6):536–544, Apr. 2003.

[88] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, Sept. 1978.

Errata and Clarifications for the Publications

- In[P2], Equation (2) should be:

$$\mathbf{X} \approx \mathbf{AS}. \tag{7.1}$$

- In [P4], Equation (8)-(11) have minor errors. Here are the corrected versions:

$$\begin{aligned}&\int_{-\infty}^{\infty} \exp\left[-\frac{(a-b)^2}{e^2} - \frac{(a-c)^2}{f^2} + g\right] da \\ &= \frac{\sqrt{\pi}|e||f|}{\sqrt{e^2+f^2}} \exp\left[-\frac{(c-b)^2}{e^2+f^2} + g\right],\end{aligned} \tag{7.2}$$

$$\begin{aligned}&\int_{-\infty}^{\infty} \mathcal{N}_1(\mathbf{x}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)\mathcal{N}_2(\mathbf{x}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)\, d\mathbf{x} \\ &= \frac{1}{(2\pi)^{N/2} \prod_{n=1}^{N} \sqrt{\sigma_{1,n}^2 + \sigma_{2,n}^2}} \exp\left[-\frac{1}{2} \sum_{n=1} \frac{(\mu_{1,n} - \mu_{2,n})^2}{\sigma_{1,n}^2 + \sigma_{2,n}^2}\right]\end{aligned} \tag{7.3}$$

$$e_{11} = \sum_{i=1}^{I} \sum_{j=1}^{I} w_i w_j Q_{i,j,1,1}, \tag{7.4}$$

$$e_{22} = \sum_{i=1}^{J} \sum_{j=1}^{J} v_i v_j Q_{i,j,2,2}. \tag{7.5}$$