2004:074

LULEÅ UNIVERSITY OF TECHNOLOGY

MASTER'S THESIS

Audio Classification and Content Description

TOBIAS ANDERSSON

MASTER OF SCIENCE PROGRAMME

Department of Computer Science and Electrical Engineering
Division of Signal Processing

2004:074 CIV • ISSN: 1402 - 1617 • ISRN: LTU - EX - - 04/74 - - SE

Audio classification and content description

Tobias Andersson

Audio Processing & Transport
Multimedia Technologies
Ericsson Research, Corporate Unit
Luleå, Sweden

March 2004

Abstract

The rapid increase of information imposes new demands of content management. The goal of automatic audio classification and content description is to meet the rising need for efficient content management.

In this thesis, we have studied automatic audio classification and content description. As description of audio is a broad field that incorporates many techniques, an overview of the main directions in current research is given. However, a detailed study of automatic audio classification is conducted and a speech/music classifier is designed. To evaluate the performance of a classifier, a general test-bed in Matlab is implemented.

The classification algorithm for the speech/music classifier is a k-Nearest Neighbor, which is commonly used for the task. A variety of features are studied and their effectiveness is evaluated. Based on feature's effectiveness, a robust speech/music classifier is designed and a classification accuracy of 98.2 % for 5 seconds long analysis windows is achieved.

Preface

The work in this thesis was performed at Ericsson Research in Luleå. I would like to take the opportunity to thank all people at Ericsson for an educating and giving time. I especially want to thank my supervisor Daniel Enström at Ericsson Research for valuable guidance and suggestions to my work. I am very grateful for his involvement in this thesis. I would also like to thank my examiner James P. LeBlanc at the division of Signal Processing at Luleå University of Technology for valuable criticism and support. And finally, thank you Anna for all the nice lunch breaks we had.

Contents

Chapter 1

Introduction

1.1 Backgound

Our daily life is highly dependent on information, for example in formats as text and multimedia. We need information for common routines as watching/reading the news, listening to the radio, watching a video et cetera. However, we easily run into problems when a certain type of information is needed. The immense flow of information makes it hard to find what you are looking for.

The rapid increase of information imposes new demands of content management as the media archives and consumer products begin to be very complex and hard to handle. Currently people perform searches in databases with different meta-tags, which only describe whole chunks of information with brief constellations of texts. The meta-tags are constructed by different individuals and one realizes that the interpretation of meta-tags can differ from individual to individual. An automatic system that describes audio would systematize labelling and would also allow searches on the actual data, not just on labels of it. Better content management is the goal of automatic audio description systems. Some commercial content management applications are already out but many possible applications are still undeveloped.

For example, a person is listening to the radio and wants to listen to jazz. Unfortunately, all the radio stations play pop music mixed with advertisements. The listener gives up searching for jazz and gets stuck with pop music.

The above example can be solved with an automatic audio description system. The scenario may then change to the following. The person that wants to listen to jazz only find pop music on all tuned radio stations. The listener then press a "search for jazz"-button on the receiver and after a couple of seconds the receiver change radio station and jazz flows out of the speakers. This example shows how content management may be an efficient tool that simplifies daily routines involving information management.

1.2 Thesis Objectives

The objectives of this thesis form a coherent study of audio classification and content description. The objectives are divided into four parts:

- To get an understanding of audio content classification and feature extraction. Many areas are closely linked with audio classification and feature extraction. Hence, a broad literature study on different research areas within content description is done.
- To implement a test-bed for testing content classification algorithms in Matlab. The aim is to make a general framework that can be used for evaluation of different classification schemes.
- To examine existing content classification and feature extraction solutions and to determine their performance.
- To investigate further enhancements of content classification algorithms.

1.3 Thesis Organization

Chapter 2 gives an overview of current research and what techniques that are used in audio classification and content description. In chapter 3 the current standard for description of audio content, MPEG-7, is presented and a brief overview of the standard is given. In chapter 4 classification in general is explained. Some applications that use content management techniques are presented in chapter 5. The design and evaluation of a speech/music classifier is presented in chapter 6 and the conclusion is found in chapter 7 as well as considerations for future work. An explanation of the implemented test-bed is given in Appendix A and a list of used abbreviations in the report can be found in Appendix B.

Chapter 2

Current Research

The need for more advanced content management can clearly be seen in the research community. Many research groups work with new methods to analyze and categorize audio. As the need for different content management systems grow, different applications are developed. Some applications are well developed today where speech recognition is one good example. However, most content management applications are quite complex and a lot of work has to be done to implement stable solutions.

In this chapter we present an outline of some of the main directions in current audio description research.

2.1 Classification and Segmentation

Audio classification and segmentation can provide powerful tools for content management. If an audio clip automatically can be classified it can be stored in an organized database, which can improve the management of audio dramatically. Classification of audio can also be useful in various audio processing applications. Audio coding algorithms can possibly be improved if knowledge of the content is known. Audio classification can also improve video content analysis applications, since audio information can provide at least a subset of the information presented.

An audio clip can consist of several classes. It can consist of music followed by speech, which is typical in radio broadcasting. Hence, segmentation of the audio clip can be used to find where music and speech begin. This is practical for applications as audio browsers, where the user may browse for particular classes in recorded audio. Segmentation can also improve classification, when classification is coarse at points where the audio content type changes in an audio clip.

2.1.1 General Audio Classification and Segmentation

The basis for many content management applications is that knowledge automatically can be extracted from the media. For instance, to insert an audio clip with speech at the right place in a sorted database it has to be known what characteristics this clip of speech has. There is a clear need for techniques that can make these classifications and many have been proposed.

However, general audio clips can contain countless differences and similarities that make the complexity of general audio classification very high. The possible content of audio has to be known to make a satisfactory classification. This makes a hierarchical structure of audio classes a suitable model, which allows classification of audio to be made into a few set of classes at each level. As an example, a classifier can first classify audio between speech and music. The music can then be classified into musical genres and speech into genders of the speakers.

Zhang [1] describes a general audio classification scheme that can be used to segment an arbitrary audio clip. The classification is divided into two steps. The first step discriminate the audio between speech and nonspeech segments. In the second step speech segments are further processed to find speaker changes and nonspeech segments are discriminated between music, environmental sounds and silence. Audio features, which give information about the examined audio, are analyzed in each step of the analysis to make a good discrimination. This approach proves to work well and a total accuracy rate of at least 96% is reported.

Attempts to classify general audio into many categories at once can be difficult. Li [2] achieves accuracy over 90 % in a scheme that classify the audio into seven categories consisting of silence, single speaker speech, speech and noise, simultaneous speech and music, music, multiple speaker's speech and environmental noise.

Tzanetakis [3] also propose a hierarchy of audio classes to limit the number of classes that a classifier has to classify between. First, audio is classified into speech and music. Then, speech and music is classified into even more genres. A reported classification accuracy of 61% for ten music genres is achieved.

A speech/music classifier is presented in Scheirer's [4] work, where a number of classification schemes are used. A classification accuracy of 98.6 % is achieved on 2.4 seconds long analysis lengths.

2.1.2 Music Type Classification

Classification of music is somewhat vague. How detailed can one categorize music? What is the difference between blues influenced rock and blues? That can be a difficult question even for the most trained musicians to answer. Limitations have to be done in order to get a realistic classification scheme that can be implemented. Simple categorization of music into a few set of classes is often used.

When we listen and experience music many concepts give us a feeling of what we hear. Features like tempo, pitch, chord, instrument timbre and many more make us recognize music types. These features can be used in an automatic music type classification if they can be extracted from media. It is however difficult to extract these features. Spectral characteristics are easier to work with and give some promising results.

Han [5] classifies music into the three types; popular, jazz and classical music. A set of simple spectral features in a nearest mean classifier gives reasonable accuracy. A classification accuracy of 75%, 30% and 60% between popular, jazz and classical music respectively is achieved.

Pye [6] look for features that can be extracted directly from encoded music. He extracts features from MPEG layer 3 encoded music. As MPEG layer 3 use perceptual encoding techniques, an audio clip of this format may be more descriptive than a raw audio clip since it only contains what humans here. The features are fast to compute and give good performance with classification accuracy of 91%.

Zhang [7] argues for a new feature, Octave-based Spectral Contrast. Octave-based Spectral Contrast can be a measure of the relative distribution of harmonic and non-harmonic components in a spectrum. It measures spectrum peaks and valleys in several sub-bands and classification is done into baroque, romantic, pop, jazz and rock. The total accuracy is about 82.3 % for classification of 10 second long clips.

2.1.3 Content Change Detection

Content change detection, or segmentation, techniques analyze different audio features and highlight regions where possible content changes occur. Automatic content change detection can be used to simplify browsing in recorded news programs, sports broadcasts, meeting recordings et cetera. Possible scenarios are option to skip uninteresting news in a news broadcast, summarize a football game to see all goals and so on.

Kimber [8] describes an audio browser that segments audio. Speaker and acoustic classes (e.g. music) changes give indices that render segmentation possible. The accuracy of the segmentation is highly dependent on the audio's quality and error rates are reported between 14% to 1%.

A common and very efficient way to increase positive segmentation indices is to combine audio and visual data. Albiol [9] describe a method that finds indices when specific persons speak in a video clip. The system has been extensively tested and accuracy of around 93% is achieved for audio only. With combined audio-visual data accuracy at 98% is achieved.

2.2 Recognition

One major field within audio classification and content description is recognition of audio. Automatic recognition of audio can allow content management applications to monitor audio streams. When a specific signal is recognized the application can make a predefined action, as sounding an alarm or logging an event. Another useful application is in various search and retrieval scenarios, when a specific audio clip can be retrieved based on shorter samples of it.

2.2.1 Music Recognition

As automatic recognition of audio allow monitoring of audio streams and efficient searches in databases, music recognition is interesting. Monitoring of audio streams to recognize music can be of interest for record companies that need to get statistics of how often a song is played on the radio. Efficient searches in databases can simplify management of music databases. This is important because music is often stored in huge databases and the management of it is very complex. Music recognition can simplify the management of large amount of content.

Another interesting use of automatic music recognition is to identify and classify TV/Radio commercials. This is done by searching video signals for known background music. Music is an important way to make consumers recognize different companies and Abe [10] propose a method to recognize TV commercials. The method finds 100% of known background music with as low SNR as -10dB, which makes recognition of TV commercials possible.

2.2.2 Speech Recognition

Many applications already use speech recognition. It is possible to control mobile phones, even computers, with speech. However, in content management applications the techniques aim to allow indexing of spoken content. This can be done by segmenting audio clips by speaker changes and even the meaning of what is said.

Speech recognition techniques often use hidden Markov models (HMM), where a good introduction to HMM is given by Rabiner [11]. HMM are widely used and recent articles [12] [13] on speech recognition are also based on them.

2.2.3 Arbitrary Audio Recognition

Apart from music and speech recognition, other useful recognition schemes are being developed. They can build a basis of tools for recognition of general scenes. These recognition schemes can of course also be used in music recognition, since music consists of several recognizable elements.

Gouyon [14] uses recognizable percussive sounds to extract time indexes of their occurrences in an audio signal. This is used to calculate rhythmic structures in music, which can improve music classification. Synak [15], [16] use MPEG-7 descriptors to recognize musical instrument sounds.

The central issue of arbitrary audio recognition is that detailed recognition can give much information about what context it is in. As Wold [17] proposes, monitoring sound automatically including the ability to detect sounds can make surveillance more efficient. For instance, it would be possible to detect a burglar if the sound of a window crash is detected in an office building at nighttime.

2.3 Content Summarization

To find new tools to help administering data, techniques are being developed to automatically describe it. When we want to decide whether to read a book or not we can read the summary on the back. When we decide whether to see a movie or not we can look at a trailer. Analogous to this, we may get much information of an audio file from a short audio clip that describes it. To manually produce these short clips can be time consuming and because of the amount of media available automatic content summarization is necessary. Automatic content summarization would also allow a general and concise description that is suitable for database management.

2.3.1 Structure detection

Structure detection can be informative about how the content in an audio clip is partitioned. The technique aims to find similar structures within the audio stream and label the segments. Casey [18] proposes a method, which is part of the MPEG-7 standard, that structures audio. It is shown to be efficient in structuring music[1]. The main structures of a song can then be labelled into the intro, verse, chorus, bridge et cetera.

2.3.2 Automatic Music Summarization

Music genres as pop, rock and disco are often based on a verse and chorus pattern. The main melody is repeated some times and that is what people memorize. Therefore, music summarization can be done by extracting these repetitive sequences. This works for repetitive music genres but is difficult to achieve for jazz and classical music, which involves significant variations of the main melody.

Logan [19] proposes a method that achieves this in three steps. First, the structure of the song is found. Then, statistics of the structure are used to produce the final summarization. Although the first tests are limited the method gives promising results.

Xu [20] presents an algorithm for pure music summarization. It uses audio features for clustering segmented frames. The music summary is generated based on the clustering results and domain-specific music knowledge. This method can achieve better results than Logan's [19]. However, no tests were made on vocal music.

2.4 Search and Retrieval of Audio

Extensive research is conducted to develop applications for search and retrieval of audio. This is a very complex task as recognition, classification and segmentation of audio has to be used. Further, advanced search algorithms for databases have to be used for efficient management. In spite of the complexity, the applications have huge potential and are interesting to develop because of their intuitive approach to search and retrieval.

2.4.1 Content Based Retrieval

Retrieval of a specific content can be achieved by searching for a given audio clip, characteristics etc. Other ways to search in databases is to use a set of sound adjectives that characterize the media clip wanted. Wold [17] describes an application that can search for different categories as laughter, animals, bells, crowds et cetera. The application can also search for adjectives to allow searches as "scratchy" sound.

[1]Casey has an interesting page on the internet about structuring of music. See www.musicstructure.com for more information about his work.

2.4.2 Query-by-Humming

In order to develop user friendly ways to retrieve music, researchers strive to get away from traditional ways to search for music. People, in general, do not memorize songs by title or by artists. People often remember certain attributes of a song. Rhythm and pitch form a melody that people can hum for themselves. A logical way to search for music is therefore by humming, which is called query-by-humming (QBH).

Song [21] presents an algorithm that extracts music melody features from humming data and matches the melody information to a music feature database. If a match is found retrieval of the desired clip can be done. This shows good promise for a working QBH-application.

Liu [22] propose a robust method that can match humming to a database of MIDI files. Note segmentation and pitch tracking is used to extract features from the humming. The method achieves an accuracy of 90% with a query of 10 notes.

Chapter 3

MPEG-7 Part 4: Audio

MPEG-7 Part 4 provides structures for describing audio content. It is build upon some basic structures in MPEG-7 Part 5. For a more detailed discussion about structures in MPEG-7, study the MPEG-7 overview [23]. These structures use a set of low-level Descriptors (LLDs) that are different audio features for use in many different applications. There are also high-level Description Tools that include both Descriptors (Ds) and Description Schemes (DSs), which are designed for specific applications.

The MPEG-7 standard is continuously being developed and enhanced. Version 2 is being developed but an overview of the currently available tools of the MPEG-7 Audio standard is seen below.

3.1 Introduction to MPEG-7

In the MPEG-7 overview [23] it is written that MPEG (Moving Picture Experts Group) members understood a need for content management and called for a standardized way to describe media content. In October 1996, MPEG started a new work item that is named "Multimedia Content Description Interface". In the fall 2001 it became the international standard ISO/IEC 15398. MPEG-7 provides a standardized set of technologies for describing multimedia content. The technologies do not aim for any particular application; the standardized MPEG-7 elements are aimed to support as many applications as possible.

A list of all parts of the MPEG-7 standard can be seen below. For a more detailed discussion about any specific part, see the overview of the MPEG-7 standard [23].

The MPEG-7 consists of 8 parts:

1. MPEG-7 Systems - the tools needed to prepare MPEG-7 descriptions for efficient transport and storage and the terminal architecture.

2. MPEG-7 Description Definition Language - the language for defining the syntax of the MPEG-7 Description Tools and for defining new Description Schemes.

3. MPEG-7 Visual - the Description Tools dealing with (only) Visual descriptions.

4. MPEG-7 Audio - the Description Tools dealing with (only) Audio descriptions.

5. MPEG-7 Multimedia Description Schemes - the Description Tools dealing with generic features and multimedia descriptions.

6. MPEG-7 Reference Software - a software implementation of relevant parts of the MPEG-7 Standard with normative status.

7. MPEG-7 Conformance Testing - guidelines and procedures for testing conformance of MPEG-7 implementations.

8. MPEG-7 Extraction - and use of descriptions informative material (in the form of a Technical Report) about the extraction and use of some of the Description Tools.

3.2 Audio Framework

The Audio Framework consists of seventeen low level Descriptors for temporal and spectral audio features. This set of Descriptors is as generic as possible to allow a wide variety of applications to make use of them. They can be divided into six groups and, as seen in figure 3.1. Apart from these six groups there are the very simple MPEG-7 silence Descriptor. It is however very useful and complete the Audio Framework. Each group is briefly explained below, and the explanations are derived from the MPEG-7 overview [23] and the text of International Standard ISO/IEC 15938-4 [24]

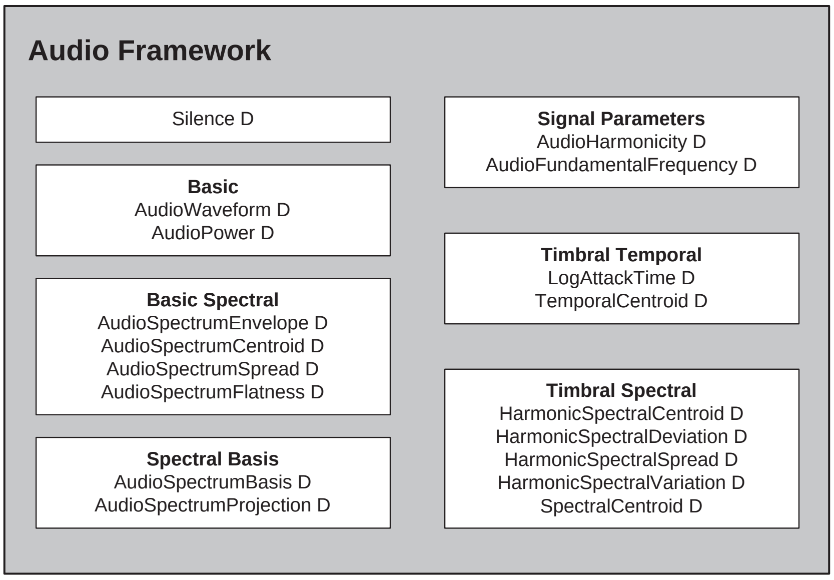


Figure 3.1: Overview of the MPEG-7 Audio Framework

3.2.1 Basic

The two basic audio Descriptors are temporally sampled scalar values. The AudioWaveform Descriptor describes the audio waveform envelope by sampling the maximum and minimum value in an analysis window of default size 10 ms. The AudioPower Descriptor describes the temporal-smoothed instantaneous power. These can give a quick and effective summary of a signal, especially for display purposes.

3.2.2 Basic Spectral

The four basic spectral audio Descriptors all have the central link that they are derived from a time-frequency analysis of the audio signal.

The AudioSpectrumEnvelope describes the short-term power spectrum of an audio signal as a time-series of spectra with a logarithmic frequency axis. It may be used to display a spectrogram, to synthesize a crude "auralization" of the data or as a general-purpose descriptor for search and comparison.

The AudioSpectrumCentroid describes the center of gravity of the log-frequency power spectrum. The AudioSpectrumCentroid is defined as the power weighted log-frequency centroid. It indicates whether the power spectrum is dominated by low or high frequencies.

The AudioSpectrumSpread describes the sound moment of the log-frequency power spectrum. It indicates whether the log-frequency power spectrum is concentrated in the vicinity of its centroid or if it is spread out over the spectrum. It allows differentiating between tone-like and noise-like sounds.

The AudioSpectrumFlatness describes the flatness properties of the short-term power spectrum of an audio signal. This Descriptor expresses the deviation of the signal's power spectrum over frequency from a flat shape. The spectral flatness analysis is calculated for a desired number of frequency bands. It may be used as a feature vector for robust matching between pairs of audio signals.

3.2.3 Spectral Basis

The two spectral basis Descriptors represent low-dimensional projections of a high-dimensional spectral space to aid compactness and recognition. A power spectrum has many dimensions and these Descriptors reduce the dimensionality of a power spectrum representation. This allows use of effective classification algorithms.

The AudioSpectrumBasis Descriptor contains basis functions that are used to project high-dimensional spectrum descriptions into a low-dimensional representation.

The AudioSpectrumProjection Descriptor represents low-dimensional features of a spectrum, derived after projection against a reduced basis given by AudioSpectrumBasis.

3.2.4 Signal Parameters

The two signal parameter Descriptors apply chiefly to periodic or quasi-periodic signals.

The AudioFundamentalFrequency Descriptor describes the fundamental frequency of the audio signal. Fundamental frequency is a good predictor of musical pitch and speech intonation.

The AudioHarmonicity Descriptor describes the degree of harmonicity of an audio signal. This allows distinguishing between sounds that have a harmonic spectrum and non-harmonic spectrum, e.g. between musical sounds and noise.

3.2.5 Timbral Temporal

The two timbral temporal Descriptors describe the temporal characteristics of segments of sounds. These are useful for description of musical timbre.

The LogAttackTime Descriptor estimates the "attack" of a sound. The Descriptor is the logarithm (decimal base) of the time duration between the time the signal starts to the time it reaches its stable part.

The TemporalCentroid Descriptor describes where in time the energy is focused, based on the sound segment's length.

3.2.6 Timbral Spectral

The five timbral spectral Descriptors describe the temporal characteristics of sounds in a linear-frequency space. This makes timbral spectral combined with timbral temporal Descriptors especially useful for description of musical instrument's timbre.

The SpectralCentroid Descriptor is very similar to the AudioSpectrumCentroid with its use of a linear power spectrum as the only difference between them.

The HarmonicSpectralCentroid Descriptor is the amplitude-weighted mean of the harmonic peaks of a spectrum. It is similar to the other centroid Descriptors but applies only to harmonic parts of the musical tone.

The HarmonicSpectralDeviation Descriptor is the spectral deviation of log-amplitude components from a global spectral envelope.

The HarmonicSpectralSpread Descriptor is the amplitude weighted standard deviation of the harmonic peaks of the spectrum, normalized by the instantaneous HarmonicSpectralCentroid.

The HarmonicSpectralVariation Descriptor is the normalized correlation between the amplitude of the harmonic peaks of two adjacent frames.

3.3 High Level Tools

The Audio Framework in the MPEG-7 Audio standard describes how to get features. These are clearly defined and can give information about audio. Whether the information has meaning or not depends on how the feature is analyzed and in what context the features are used in. MPEG-7 Audio describes five sets of audio Description Tools that roughly explain and give examples on how Low Level Features can be used in content management.

The "high level" tools are Descriptors and Description Schemes. These are either structural or application oriented and make use of low level Descriptors from the Audio Framework.

The high level tools cover a wide range of application areas and functionalities. The tools both provide functionality and serve as examples of how to use the low level framework.

3.3.1 Audio Signature Description Scheme

The Audio Signature Description Scheme is based on the LLD spectral flatness. The spectral flatness Descriptor is statistically summarized to provide a unique content identifier for robust automatic identification of audio signals.

The technique can be used for audio fingerprinting, identification of audio in sorted databases and, these combined, content protection. A working solution based on the AudioSignature DS is the AudioID system developed by Fraunhofer Institute of Integrated Circuits IIS. It is a robust matching system that gives good results with large databases and is tolerant to audio alternations as mobile phone transmission. That is, the system can find metatags to audio submitted via a mobile phone.

3.3.2 Musical Timbre Description Tools

The aim of the Musical Timbre Description Tools is to use a set of low level descriptors to describe the timbre of instrument sounds. Timbre is the perceptual features that make two sounds having the same pitch and loudness sound different.

Applications can be various identification, search and filtering scenarios where specific types of sound are considered. The technique can be used to simplify and improve performance of, for example, music sample database management and retrieval tools.

3.3.3 Melody Description Tools

The Melody Description Tools are designed for monophonic melodic information and forms a detailed representation of the audio's melody. Efficient and robust matching between melodies in monophonic information is possible with these tools.

3.3.4 General Sound Recognition and Indexing Description Tools

The General Sound Recognition and Indexing Description Tools are a set of tools that support general audio classification and content indexing. In contrast to specific audio classification systems, which can work well for the designed task, these General Sound Recognition Tools aim to be a tool useable for diverse source classification [25]. The tools can also be used to efficiently index audio segments into smaller similar segments.

The low level spectral basis Descriptors is the foundation to these tools. The AudioSpectrumProjection D, with the AudioSpectrumBasis D, is used as the default descriptor for sound classification. These descriptors are collected for different classes of sounds and a SoundModel DS is created. The SoundModel DS contain a sound class label and a continuous hidden Markov Model. A

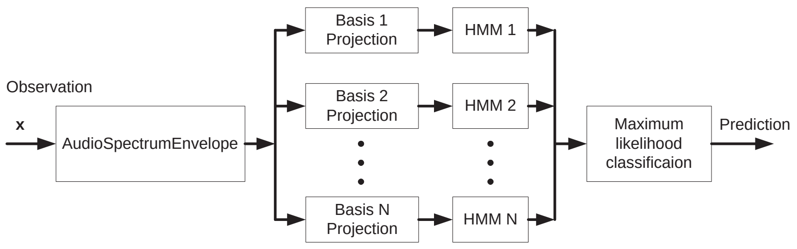


Figure 3.2: Multiple hidden Markov models for automatic classification. Each model is trained for a separate class and a maximum likelihood classification for a new audio clip can be made.

SoundClassificationModel DS combine several SoundModels into a multi-way classifier, as can be seen in Figure 3.2.

3.3.5 Spoken Content Description Tools

The Spoken Content Description Tools are a representation of the output of Automatic Speech Recognition. It is designed to work at different semantic levels. The tools can be used for two broad classes of retrieval scenario: indexing into and retrieval of an audio stream, and indexing of multimedia objects annotated with speech.

Chapter 4

Audio Classification

As described, audio classification and content description involves many techniques and there are many different applications. The focus in this thesis is on classification of audio and, therefore, this chapter will outline general classification techniques and also present features applicable for audio classification.

4.1 General Classification Approach

There are many different ways to implement automatic classification. The available data to be classified may be processed in more or less efficient ways to give informative features of the data. Another variation between classifiers is how efficient the features of unknown samples are analyzed to make a decision between different classes. Many books regarding machine learning and pattern recognition is written and many ideas presented here comes from Mitchell's [26] and Duda's [27] books, which also are recommended for further reading.

A general approach is shown in Figure 4.1 to illustrate a classifier's scheme. First a sequence of data has to be observed, which is stored in $\mathbf{x}$. The observed sequence, $\mathbf{x}$, does not say much about what information it contains. Further processing of the sequence can isolate specific characteristics of it. These characteristics are stored in a feature vector $\mathbf{y}$. The feature vector contains several descriptive measures that can be used to classify the sequence into defined classes, which is done by the classifier.

Many thinkable applications can use this basic approach to accomplish different classification tasks. An example is outlined below that show how a medical application classify cells in a tissue sample to normal and cancerous cells.

A tissue sample is observed and a feature is measured. As most cancerous

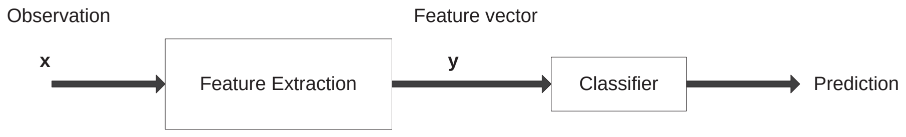
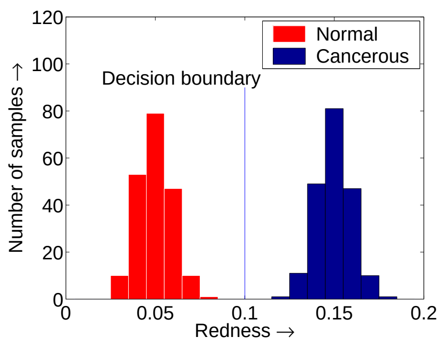
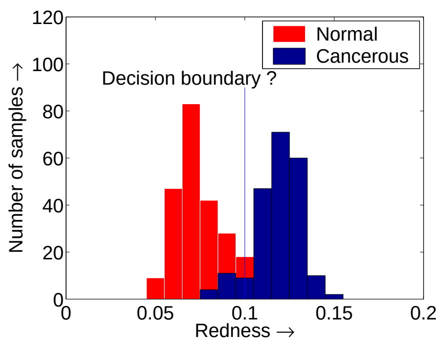


Figure 4.1: Basic scheme of automatic classification



(a)



(b)

Figure 4.2: Histograms for the redness of cells. Two classes of cells are considered; normal and cancerous ones. In plot (a), the two classes are completely separable and a decision boundary can be chosen to discriminate between them. Plot (b) exemplifies a more realistic scenario, where the two considered classes cannot be completely separable by one feature. Here, the placement of the decision boundary is not obvious.

cells are red, the feature describes the redness of the cells. The histogram of the measured values for a set of samples can be seen to the left in Figure 4.2. The figure shows that normal and cancerous cells are clearly separable and a *decision boundary* is chosen so that a perfect decision is done between the two classes. This example is however an idealized classification scenario. In most practical scenarios, the classes are not completely separable and to the right in Figure 4.2 a more realistic histogram can be seen. Here, it is clear that the normal and cancerous cells cannot be completely separable. It is no longer obvious how to choose the decision boundary. Despite how the decision boundary is chosen, there will always be cells that are misclassified.

To minimize the number of cells that are misclassified, or to use typical terminology, to get better *classification accuracy* more features can be measured and considered. As, for example, the size of normal and cancerous cells usually differs a suitable second feature is the size of the cells. The feature vector is composed by the two features, size and redness of cells. The two features are by themselves descriptive but combined they can give even more information about the cells. In Figure 4.3 measured values of the two features are plotted against each other. To make a classification based on these points a straight line can be drawn in the figure, which then serve as a decision boundary. The classification accuracy is higher but there will still be some cells misclassified.

As seen, a classification scheme consists of several steps. To make a good classifier that have high classification accuracy all steps have to relate. The way observations are done affects how to measure features and the way features are measured affects how to implement the classifier. This imposes many problems and each step in the classification scheme has to be carefully designed.

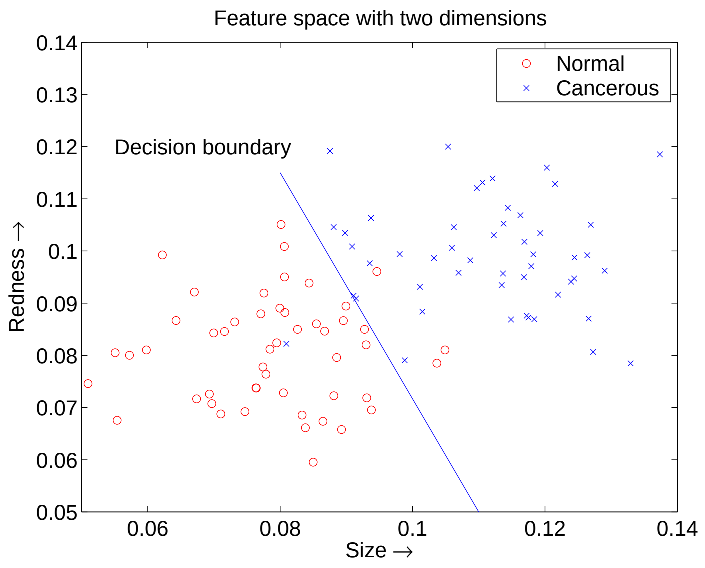


Figure 4.3: The two measured features, size and redness, for a set of cells. The normal cells are smaller and less red. A decision boundary can be chosen to be a straight line that, compared with decisions based on one feature, drastically improves the classification accuracy.

4.1.1 Feature Extraction

The first step in the classification scheme, shown in Figure 4.1, is critical to the classification scheme's accuracy. The feature vector, $\mathbf{y}$, that is composed by several features should be as discriminative between considered classes as possible. Ideally, the feature vectors should clearly separate all measured samples from different classes. In reality, this is not possible but the aim for the feature extraction step is to gain as much information as possible about the observed sequence, $\mathbf{x}$.

Features are extracted by different signal processing algorithms to get as much discriminative information as possible from the observed sequence. If a classification between music and speech is considered a feature that tells how much energy there is in the GHz band is irrelevant. That feature is not able to discriminate between speech and music, when the frequency range is too high. However, a feature as the spread of the frequency spectrum in the range 0 Hz to 22050 Hz gives discriminative information and can be used for classification.

How the feature vector, $\mathbf{y}$, is composed is important for the classification accuracy. An effectively composed feature vector simplifies classification and therefore simplifies the classifier's design. Hence, what features to extract depends on the context.

4.1.2 Learning

To make classification of new samples possible, the classifier's decision boundaries has to reflect the considered classes. Relevant decision boundaries are achieved by *learning*, which is the process when a set of samples is used to tune the classifier to the desired task. How the classifier is tuned depends on what algorithm the classifier use. However, in most cases, the classifier has to be given a *training set* of samples, which the classifier uses to construct decision boundaries.

In general, there are two ways to learn classifiers depending on what type of classifier that is considered. The first method to learn a classifier is called *supervised learning.* In this method, the training set consists of pre-classified samples that the classifier uses to construct decision boundaries. The pre-classification is done manually. Pre-classification can, in some scenarios, be difficult to make. Much experience and expertise is needed to pre-classify cell samples to be normal or cancerous in our previous example. In music genre classification, the pre-classification can be problematic when the boundaries between different classes can be somewhat vague and subjective. A training method without the manual pre-classification is named *unsupervised learning.* In this method, the training set consists of unclassified samples that the classifier uses to form clusters. The clusters are labelled into different classes and this way the classifier can construct decision boundaries.

4.1.3 Classification

Once the classifier has been trained it can be used to classify new samples. A perfect classification is seldom possible and numerous different classification algorithms are used with varying complexity and performance. Classifier algorithms based on different statistical, instance-based and clustering techniques are widely used.

The main problem for a classifier is to cope with feature value variation for samples belonging to specific classes. This variation may be large due to the complexity of the classification task. To maximize classification accuracy decision boundaries should be chosen based on combinations of the feature values.

4.1.4 Estimation of Classifiers Performance

Analysis of how well a specific classification scheme work is important to get some expectation of the classifiers accuracy. When a classifier is designed, there are a variety of methods to estimate how high accuracy it will have on future data. These methods allow comparison to other classifiers performance.

A method described in Mitchell's [26] and Duda's [27] books, which also is used by many research groups [1] [3] [4], is *cross-validation.* Cross-validation is used to maximize the generality of estimated error rates. The error rates are estimated by dividing a labelled data set into two parts. One part is used as *training set* and the other as a *validation set* or *testing set.* The training set is used to train the classifier and the testing set is used to evaluate the classifiers performance. A common variation is the "10-fold cross-validation" that divides the data set into 10 separate parts. Classification is performed 10 times, each time with a different testing set. 10 error rates are estimated and the final estimate is simply the mean value of these. This method further generalizes the estimated error rates by iterating the training process with different training and testing sets.

4.2 Feature Extraction

Due to the complexity of human audio perception extracting descriptive features is difficult. No feature has yet been designed that with 100% certainty can distinguish between different classes. However, reasonably high classification accuracy, into different categories, has been achieved with a combination of features. Below are several features, suitable for various audio classification tasks, outlined.

4.2.1 Zero-Crossing Rate

Values of the Zero-Crossing-Rate (ZCR) is widely used in speech/music classification. It is defined to be the number of time-domain zero-crossings within a processing window, as shown in Equation 4.1.

$$ZCR = \frac{1}{M-1} \sum_{m=0}^{M-1} |sign(x(m)) - sign(x(m-1))| \tag{4.1}$$

where *sign* is 1 for positive arguments and 0 for negative arguments, M is the total number of samples in a processing window and *x(m)* is the value of the *mth* sample.

The algorithm is simple and has low computational complexity. Scheirer [4] use the ZCR to classify audio between speech/music, Tzanetakis [3] use it to classify audio into different genres of music and Gouyon [14] use it to classify percussive sounds.

Speech consists of voiced and unvoiced sounds. The ZCR correlate with the frequency content of a signal. Hence, voiced and unvoiced sounds have low respectively high zero-crossing rates. This results in a high variation of ZCR. Music does not typically have this variation in ZCR but it has to be said that some parts of music has similar variations in ZCR. For instance, a drum intro in a pop song can have high variations in ZCR values.

4.2.2 Short-Time-Energy

Short-time energy (STE) is also a simple feature that is widely used in various classification schemes. Li [2] and Zhang [1] use it to classify audio. It is defined to be the sum of a squared time domain sequence of data, as shown in Equation 4.2

$$STE = \sum_{m=0}^{M-1} x^2(m) \tag{4.2}$$

where M is the total number of samples in a processing window and *x(m)* is the value of the *mth* sample.

As STE is a measure of the energy in a signal, it is suitable for discrimination between speech and music. Speech consists of words and mixed with silence. In general, this makes the variation of the STE value for speech higher than music.

4.2.3 Root-Mean-Square

As the STE, the root-mean-square (RMS) value is a measurement of the energy in a signal. The RMS value is however defined to be the square root of the average of a squared signal, as seen in Equation 4.3.

$$RMS = \sqrt{\frac{1}{M}\sum_{m=0}^{M-1} x^2(m)} \tag{4.3}$$

where M is the total number of samples in a processing window and *x(m)* is the value of the *mth* sample.

Analogous to the STE, the variation of the RMS value can be discriminative between speech and music.

4.2.4 High Feature-Value Ratio

Zhang [1] propose a variation of the ZCR that especially designed for discrimination between speech and music. The feature is called High Zero-Crossing-Rate Ratio (HZCRR) and its mathematical definition can be seen in Equation 4.4

$$HZCRR = \frac{1}{2N}\sum_{n=0}^{N-1}[sgn(ZCR(n) - 1.5avZCR)) + 1] \tag{4.4}$$

where N is the total number of frames, n is the frame index, *ZCR(n)* is the zeros-crossing rate at the *nth* frame, *avZCR* is the average *ZCR* in a 1 second window and *sgn(.)* is a sign function. That is, HZCRR is defined as the ratio of number of frames whose ZCR is above 1.5-fold average zero-crossing rate in a 1 second window.

The motive of the definition comes directly from the characteristics of speech. Speech consists of voiced and unvoiced sounds. The voiced and unvoiced sounds have low respectively high zero-crossing rates, which results in a high variation of ZCR and the HZCRR. Music does not typically have this variation in ZCR but it has to be said that some parts of music has similar values in HZCRR. For instance, a drum intro in a pop song can have high HZCRR values.

The HZCRR will be used in a slightly different way. The feature is generalized to work for any feature values. Hence, high feature-value ration (HFVR) is defined as Equation 4.5

$$HFVR = \frac{1}{2N}\sum_{n=0}^{N-1}[sgn(FV(n) - 1.5avFV)) + 1] \tag{4.5}$$

where N is the total number of frames, n is the frame index, *FV(n)* is the feature value at the *nth* frame, *avFV* is the average *FV* in a processing window and *sgn(.)* is a sign function. That is, HFVR is defined as the ratio of number of frames whose feature value is above 1.5-fold average feature value in a processing window.

4.2.5 Low Feature-Value Ratio

Similar to the HZCRR, Zhang [1] propose a variation of the STE feature. The feature they propose is called Low Short-Time Energy Ratio. Its mathematical definition can be seen in Equation 4.6

$$LSTER = \frac{1}{2N}\sum_{n=0}^{N-1}[sgn(0.5avSTE - STE(n)) + 1] \tag{4.6}$$

where N is the total number of frames, n is the frame index, *STE(n)* is the short-time energy at the *nth* frame, *avSTE* is the average *STE* in a 1 second window and *sgn(.)* is a sign function. That is, LSTER is defined as the ratio of number of frames whose STE is below 0.5-fold average short-time energy in a 1 second window.

LSTER is suitable for discrimination between speech and music signals. Speech consists of words mixed with silence. In general, this makes the LSTER value for speech high whereas for music the value is low. The LSTER feature is effective for discriminating between speech and music but since it is designed to recognize characteristics of single speaker speech it can loose it effectiveness if multiple speakers are considered. Also, music can have segments of pure drum patterns, short violin patterns et cetera that will have the same characteristics of speech in a LSTER sense.

As with the HZCRR, the feature will be used in a slightly different way. The feature is generalized to work for any feature values. Hence, low feature-value ration (LFVR) is defined as Equation 4.7

$$LFVR = \frac{1}{2N}\sum_{n=0}^{N-1}[sgn(0.5avFV - FV(n)) + 1] \tag{4.7}$$

where N is the total number of frames, n is the frame index, *FV(n)* is the short-time energy at the *nth* frame, *avFV* is the average *FV* in a processing window and *sgn(.)* is a sign function. That is, LFVR is defined as the ratio of number of frames whose feature value is below 0.5-fold average short-time energy in a 1 second window.

4.2.6 Spectrum Centroid

Li [2] use a feature named spectral centroid (SC) to classify between noise, speech and music. Tzanetakis [3] use the same feature to classify music into different genres. Spectrum centroid is based on analysis of the frequency spectrum for the signal. The frequency spectrum, for use in this feature and several others, is calculated with the discrete Fourier transform (DFT) in Equation 4.8

$$A(n,k) = \left|\sum_{m=-\infty}^{\infty} x(m)w(nL - M)e^{-j(\frac{2\pi}{L})km}\right| \tag{4.8}$$

where k is the frequency bin for the nth frame, *x(m)* is the input signal, *w(m)* is a window function and L is the window length. The frequency spectrum can be analyzed in different ways and the spectrum centroid is calculated as seen in

Equation 4.9

$$SC(n) = \frac{\sum\limits_{k=0}^{K-1} k \cdot |A(n,k)|^2}{\sum\limits_{k=0}^{K-1} |A(n,k)|^2} \tag{4.9}$$

where K is the order of the DFT, k is the frequency bin for the nth frame and *A(n,k)* is the DFT of the nth frame of a signal and is calculated as Equation 4.8. That is, the spectral centroid is a metric of the center of gravity of the frequency power spectrum.

Spectrum centroid is a measure that signifies if the spectrum contains a majority of high or low frequencies. This is correlates with a major perceptual dimension of timbre; i.e. sharpness [23].

4.2.7 Spectrum Spread

Spectrum spread relates closely with the spectrum centroid. Its mathematical definition is seen in Equation 4.10

$$SS(n) = \sqrt{\frac{\sum\limits_{k=0}^{K-1} [(k - SC)^2 \cdot |A(n,k)|^2]}{\sum\limits_{k=0}^{K-1} |A(n,k)|^2}} \tag{4.10}$$

where K is the order of the DFT, k is the frequency bin for the nth frame and *A(n,k)* is the DFT of the nth frame of a signal and is calculated as Equation 4.8.

Spectrum spread is a measure that signifies if the power spectrum is concentrated around the centroid or if it is spread out over the spectrum. Music often consists of a broad mixture of frequencies whereas speech consists of a limited range of frequencies. This make the spectrum spread useful for discrimination between speech and music. The feature can also be applicable for discrimination between different musical genres when, for instance, rock have higher power spectrum spread than calm flute pieces.

4.2.8 Delta Spectrum

Another feature used by Zhang [1] and Tzanetakis [3] to discriminate between speech and music is delta spectrum (SF)[1]. It is also used to discriminate between music and environment sounds.

Its mathematical definition can be seen in Equation 4.11

$$SF = \frac{1}{(N-1)(K-1)} \sum_{n=1}^{N-1} \sum_{k=1}^{K-1} [log(A(n,k) + \delta) - log(A(n-1,k) + \delta)]^2 \tag{4.11}$$

where N is the total number of frames, K is the order of the DFT, δ is a very small value to avoid calculation overflow and *A(n,k)* is the discrete Fourier transform of the nth frame. That is, the spectrum flux is defined as the average

[1]The delta spectrum feature is often called spectrum flux (SF)

variation value of spectrum between the adjacent two frames in a processing window.

Speech consists, in general, of short words and the audio waveform is varying rapidly. Music do not typically have this characteristics, which makes the SF an efficient feature for speech/music classification.

4.2.9 Spectral Rolloff Frequency

Li [2] use a feature named spectral rolloff frequency (SRF) to classify between noise, speech and music. Its mathematical definition can be seen in Equation 4.12

$$SRF(n) = \max\Big(h \mid \sum_{k=0}^{h} A(n,k) < \mathrm{TH} \cdot \sum_{k=0}^{K-1} |A(n,k)|^2\Big) \tag{4.12}$$

where N is the total number of frames, K is the order of the DFT, TH is a threshold set to 0.92 (or something around that value) and *A(n,k)* is the DFT of the nth frame of a signal and is calculated as Equation 4.8. That is, the spectral rolloff frequency is a metric of how high in the frequency spectrum a certain part of the energy lies.

The SRF is an effective feature for classification between speech and music. The characteristics of speech signals tend to have a lower SRF than music. Music signals contain higher frequencies from instruments as flutes, distorted guitars and hi-hats. This result in that music signals tend to have high values of SRF.

4.2.10 MPEG-7 Audio Descriptors

Standardized audio features, as the MPEG-7 Audio Descriptors, are interesting to use due to their exact specification. The coherency of the standardized set of descriptors available allows efficient calculation of several features. And because it is an international standard it is widely used and understood.

The MPEG-7 Descriptor AudioSpectrumEnvelope describes the short-term power spectrum of an audio signal as a time-series of spectra with a logarithmic frequency axis. I refer the reader to the MPEG-7 Part 4: Audio documentation [24] for extraction details. However, this is a different (compared with the DFT) way to represent the spectrum of a signal that "synthesize a crude 'auralization' of the data" [23]. Although more sophisticated perceptual processing of the spectrum can be used it is still an effective estimate.

The AudioSpectrumEnvelope may be used in the above spectral based features to form the AudioSpectrumCentroid, AudioSpectrumSpread and the deltaAudioSpectrumSpread[2].

4.3 Feature Selection

A big problem in all classification tasks is that a small set of features should be chosen to compose the feature vector. As features often are correlated it is

[2]The AudioSpectrumCentroid and the AudioSpectrumSpread is part of the MPEG-7 standard. The deltaAudioSpectrumSpread is not a part of the standard

difficult to choose a few number of features that will perform well. In Duda's book [27], methods to combine features into lower dimensionality feature sets are described. These are methods like the Principal Component Analysis (PCA) and Fisher Linear Discriminant. Because of time limits, this will not be covered in this thesis. However, the interested reader may further study those techniques.

More practical approaches can be seen in Zhang's [28] and Scheirer's [4] work. Zhang compose a "Baseline" feature vector that contains 4 features. The baseline set of features give good classification accuracy alone. Other features are then added to the baseline and the difference in classification accuracy gives a measure of the effectiveness of new features.

Scheirer propose a method that indicates how effective features are alone, which give guidance how to compose the feature vector. The effectiveness of each feature is approximated by a 10-fold cross validation on the training set. Error rates are estimated for each feature and these indicate what features to use, when low error rates implies that the feature is effective. It has to be said that strictly following these values to compose the feature vector would prove bad since some features are highly correlated. However, with some knowledge of what the features measure a good feature vector may be composed.

Both of the above practical approaches do not solve the problem of correlated features but they give guidance to how to compose the final feature set for a classifier.

4.4 Classification

There are many proposed techniques for classifying audio samples into multiple classes. The most commonly techniques use different statistical approaches, where Gaussian Mixture Models (GMM) and Hidden Markov Models (HMM) give good results, and instance-based approaches, where the k-Nearest Neighbor (kNN) algorithm is a good example. A brief outline of the most common classification techniques is found below.

4.4.1 Gaussian Mixture Models

GMM are widely used in speech identification and audio classification scenarios. It is an intuitive approach when the model consists of several Gaussian components, which can be seen to model acoustic features. Much is written about them and the interested reader may study the work published by Reynolds [29]. A general overview of GMM will however be given here.

GMM is based on modelling patterns with a number of Gaussian distributions. The Gaussian distributions are combined as seen in Figure 4.4. Each component density is weighted and summed, which results in a Gaussian mixture density. The Gaussian mixture density function is seen in Equation 4.13

$$p(\overrightarrow{x}|\lambda) = \sum_{i=1}^{M} w_i b_i(\overrightarrow{x}) \tag{4.13}$$

where $p(\overrightarrow{x}|\lambda)$ is the Gaussian mixture density, M is the total number of components, w_i is the weight corresponding to component i and $b_i(\overrightarrow{x})$ are the component densities, which are given by Equation 4.14.

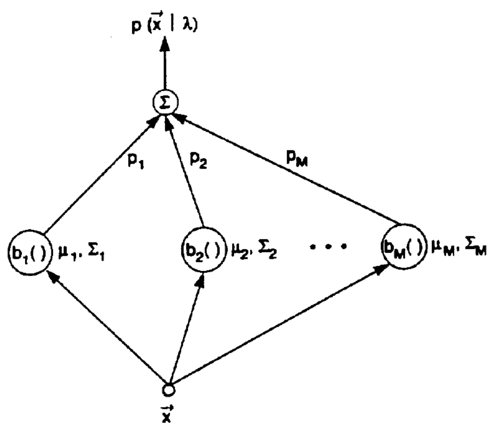

$p(\vec{x} \mid \lambda)$
Σ
p_1
p_2
p_M
$b_1(\,)$ μ_1, Σ_1
$b_2(\,)$ μ_2, Σ_2
$b_M(\,)$ μ_M, Σ_M
$\vec{x}$

Figure 4.4: Scheme of Gaussian mixture model.

$$b_i(\overrightarrow{x}) = \frac{1}{(2\pi)^{\frac{1}{L}} \mid \Sigma_i|^{\frac{1}{2}}} exp\{-\frac{1}{2}(\overrightarrow{x} - \overrightarrow{\mu}_i)^T \Sigma_i^{-1}(\overrightarrow{x} - \overrightarrow{\mu}_i)\} \tag{4.14}$$

The complete Gaussian mixture density is parameterized by the mixture weights, covariance matrices and mean vectors from all component densities. The notation used is seen in Equation 4.15.

$$\lambda = \{w_i, \overrightarrow{\mu}_i, \Sigma_i\},\ i = 1, 2, ..., M \tag{4.15}$$

In classification, each class is represented by a GMM and is referred to its model λ. Once the GMM is trained, it can be used to predict to which class a new sample most probably belong to.

4.4.2 Hidden Markov Models

The theory behind HMM was initially introduced in the late 1960's but it took to late 1980's before they got widely used. Today, many use HMM in sophisticated classification schemes. One example is Casey's work [25], which results are used in the MPEG-7 standard. A good tutorial on HMM is given by Rabiner [11].

4.4.3 k-Nearest Neighbor Algorithm

The k-Nearest Neighbor Classifier is an *instance-based classifier*. The instance-based classifiers make decisions based on the relationship between the unknown sample and the stored samples. This approach is conceptually easy to understand but can nonetheless give complex decision boundaries for the classifiers. Hence, the k-Nearest Neighbor Classifier is suitable for a variety of applications. For example, Zhang [1] use a k-Nearest Neighbor Classifier to classify if an audio signal is speech and Tzanetakis [3] use a k-Nearest Neighbor classifier to discriminate music into different musical genres. Many books about machine learning and pattern recognition often include discussions about k-Nearest Neighbor algorithms, where Mitchell's [26] and Duda's [27] books are recommended for further reading. However, a discussion on how the classifier works will be outlined below.

When training a k-Nearest Neighbor classifier, all training samples are simply stored in a n-dimensional Euclidean space, $\Re^n$. This results in a number of labelled points in the feature space. For example, figure 4.3 can actually represent the 2-dimensional Euclidean space for a k-Nearest Neighbor classifier.

As said before, the k-Nearest Neighbor Classifier makes decisions based on the relationship between a query instance and stored instances. The relationship between instances are defined to be the Euclidean distance, which definition is seen in Equation 4.16

$$d(\boldsymbol{x_i}, \boldsymbol{x_j}) = |\boldsymbol{x_i} - \boldsymbol{x_j}| = \sqrt{\sum_{r=1}^{n}(x_i(r) - x_j(r))^2} \tag{4.16}$$

where $d(\boldsymbol{x_i}, \boldsymbol{x_j})$ is the Euclidean distance between two instances, represented by the two feature vectors $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$, in a n-dimensional feature space.

Because the relationship between different instances is measured by the Euclidean distance, feature values need to be in the same region. If features with large values and low values are used, the feature with low values will have little relevance in the relationship measurement. To enhance the performance of the k-Nearest Neighbor algorithm normalization of feature values may be used. To further enhance the performance of the k-Nearest Neighbor algorithm, weighting the axes in the feature space is proposed by Mitchell [26]. This correspond to stretching axes in the Euclidean space, shortening the axes correspond to less relevant features, and lengthening the axes correspond to more relevant features.

In the case when only one neighbor is considered to make the classification, the decision boundaries are represented by cells encapsulating each instance. That is, each cell represents the class of the particular training instance in it and all new instances in that cell are classified thereafter. The cells are called *Voronoi cells* and a typical pattern is shown in Figure 4.5, which is a *Voronoi diagram.* The figure shows a close-up of the 2-dimensional feature space from the example shown in Figure 4.3. The Nearest Neighbor algorithm produces a complex decision boundary that often performs better than the linear decision boundary. Here, the classification of the new instance, $\boldsymbol{x_i}$, is the class of the instance that the cell correspond to. That is, the new instance is classified as normal.

One problem with classifiers is that they often become overtrained to a specific training set. The decision boundaries should represent the main characteristics of the classes considered, not only the characteristics of the actual training set. A Nearest Neighbor algorithm can sometimes be too precise, with respect to the training set. Smoother decision boundaries are achieved with the k-Nearest Neighbor classifier. Here, the k nearest instances in the training set is considered when a classification is made. Classification is done by a majority vote from these instances, which classes predict the new sample's class. The new instance in Figure 4.5 would, for example, be classified as cancerous if a 3-Nearest Neighbor is used.

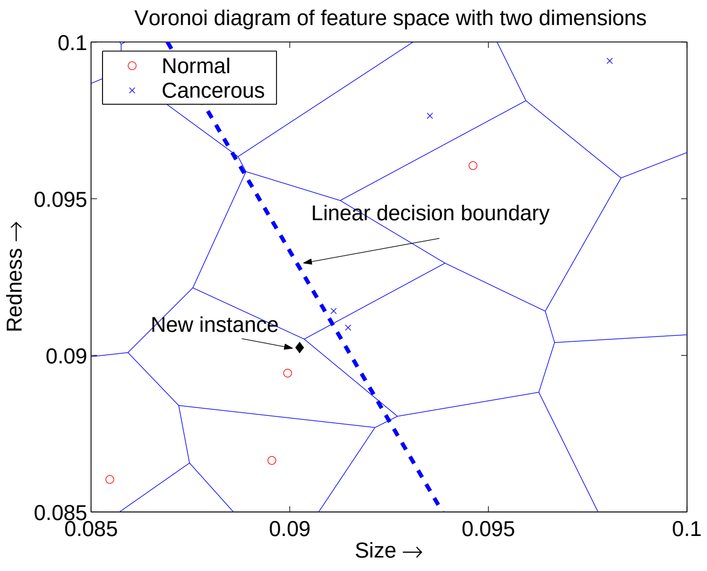


Figure 4.5: Feature space in two dimensions. The Nearest Neighbor algorithm produces voronoi cells, each representing a class determined by the training instance it encapsulate. A training set of normal and cancerous cells is shown. The decision boundaries are shown to be more complex then a linear one but the classification result can nonetheless be the same. A new instance, $\boldsymbol{x_i}$, is shown and would be classified as a normal cell for a Nearest Neighbor algorithm and as a cancerous cell for a 3-Nearest Neighbor algorithm.

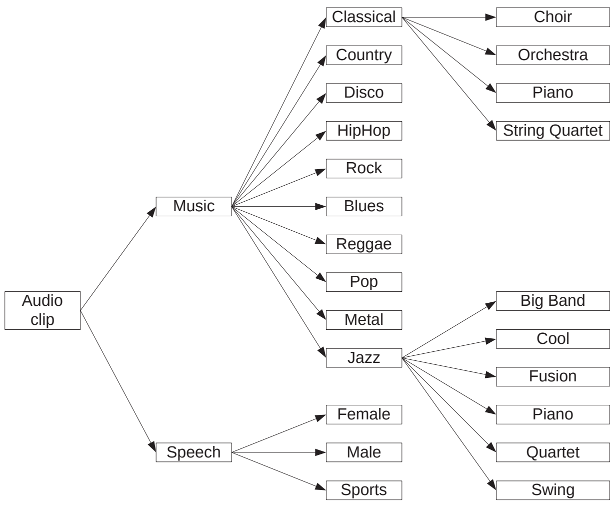


Figure 4.6: Audio classification hierarchy.

4.5 Multi-class Classification

General audio contain audio content from countless of different classes. To cope with the vast amount of possible classes that general sound have, restrictions is made on the number of classes that audio contain. That is, a hierarchical structure of classes is designed that limits the possible classes that audio contain. Many hierarchical structures are proposed and in Figure 4.6 a hierarchy proposed by Tzanetakis [3] is shown. The hierarchical structure contains three levels. First, audio is classified between speech/music. Then, if the audio is classified as speech it is further classified between male/female/sports. This way a classifier can be designed for each level, which makes efficient classification possible at each step.

Chapter 5

Applications

The potential audio classification and content description technologies are attracting commercial interests and many new products will be released in the near future. Media corporations need effective storage and search and retrieval of technologies. People need effective ways to consume media. Researchers need effective methods to manage information. Basically, the information society needs methods to manage itself.

5.1 Available commercial applications

Many commercial products have already been developed to meet the need of content management. Most existing products are either music recognition or search-and-retrieval applications. Two examples of successful (in a technology sense) commercial products are given below.

5.1.1 AudioID

Fraunhofer Institute for Integrated Circuits IIS has developed an automatic identification/recognition system named AudioID. AudioID can automatically recognize and identify audio based on a database of registered work and delivers information about the audio identified. This is basically done by extracting a unique signature from a known set of audio material, which then are stored in a database. Then unknown audio sequence can be identified by extracting and comparing a signature corresponding to it with the known database.

The system proves to work well, and recognition rates of over 99% are reported. The system use different signal manipulations, as equalization and mp3 encoding/decoding, to emulate a human perceptibility that makes the system robust to distorted signals. Recognition works, for example, with signals transmitted with GSM cell phones. The main applications are:

- **Identifying music and linking it to metadata.** The system can automatically link metadata from a database to a particular piece of music.
- **Music sales.** Automatic audio identification can give consumers metadata of unknown music. This makes consumer aware of artists and songs names, which possibly stimulates music consumption.

- **Broadcast monitoring.** The system can identify and monitor broadcasts audio program. This allows verification of scheduled transmission of advertisement and logging of played materials to ensure royalties.

- **Content protection.** Automatic audio identification may possibly be used to enhance copy protection of music.

5.1.2 Music sommelier

Panasonic has developed an automatic music analyzer that classifies music into different categories. The application classifies music based on tempo, beat and a number of other features. Music clips are plotted in an *impression map* that has active factors (Active - Quiet) on one axis and emotional factors (Hard - Soft) on the other axis. Different categories may be defined based on music clip's locations in the impression map. The software comes with three default types: Energetic, Meditative and Mellow.

The technique is incorporated into an automatic jukebox that may run on ordinary PCs. The SD-Jukebox is a tool for users to manage music content at home and it allow users to choose what type of music to listen to.

Chapter 6

Experiments and analysis

As a second part of the thesis, a test-bed in Matlab was implemented and a speech/music classifier was designed and evaluated. The test-bed was to be used to evaluate the performance for different classification schemes. For an explanation of the test-bed, see Appendix A. A speech/music classifier was designed because general audio classification is complex and involves too many steps to be studied in this thesis. Hence, the first level in the hierarchy shown in Figure 4.6 was implemented and analyzed.

In this chapter, a description of the used evaluation and design procedure is outlined. The evaluation procedure aims to get generalized evaluation results. In the design procedure, a comparison of feature's efficiency is conducted and a few feature sets are proposed. The final performance for the speech/music classifier is found at the end of this chapter.

6.1 Evaluation procedure

To evaluate the performance of different features and different classification schemes, a database of audio is collected. The database consists of approximately 3 hours of data collected from CD-recordings of music and speech. To generalize the database, care is taken to collect music recordings from different artists and genres. The speaker content is collected from recordings of both female and male speakers. All audio clips are sampled at 44.1 kHz with 16 bits per sample. The database is divided into a training set of about 2 hours and a testing set of about 1 hour of data. Special care is taken to ensure that no clips from the same artist or speaker are included in both the training and testing set. These efforts are done to get evaluation results as generalized as possible.

During the design process, features are selected to compose feature vectors based on their efficiency. The efficiency is evaluated on the training set only. This ensures that the features are evaluated and chosen by performance on other data than the testing set. That is, the features are not chosen to the specific testing set, which would produce too optimistic results.

The final evaluation of the classifiers performance is done on the testing set. This data has not been used to design the classifier and, therefore, give a good estimate of the accuracy of the classifiers.

6.2 Music/Speech classification

The design of a robust music/speech classifier involves several steps. First, a general scheme to extract features is implemented. The scheme is used to extract many features, which effectiveness can be analyzed. Based on the effectiveness of each feature, a feature vector is composed that is used by a classification algorithm to discriminate between music and speech. This section describes how features are extracted, their effectiveness and how the final classification scheme is designed.

6.2.1 Feature Extraction

Features are extracted in a scheme that uses different parameters to allow higher and lower resolution of the features. The variable resolution allows comparison of performance between different analysis lengths. This is interesting to analyze, since shorter analysis windows have several advantages.

There are basically two main ways to extract features from an audio sequence. First, simpler features can be extracted directly from the time-domain of an audio sequence, as shown in Figure 6.1(a). The extraction of time-domain features are measurements on N_h number of samples long *frames* of the audio sequence. This value is called the *hop size*, when it determines the length between each frame. The hop size is set to 10 ms, which also is the default value of the MPEG-7 standard.

The second way to extract features is from the frequency-domain of an audio sequence, as shown in Figure 6.1(b). Feature extraction in the frequency domain is done by a sliding analysis window. Each sliding window is multiplied with N_{sw} samples, centered at N_h samples long intervals, of the audio sequence. These windows are transformed to the spectral domain by fast Fourier transform (FFTs) and various measurements can be done on each FFT. The size of the sliding analysis window is set to 30 ms, which is the default value of the MPEG-7 standard. This ensures the frequency resolution to be high enough. However, other values might be used.

As typical values on N_h is about 10 ms, classification based on these values are not suitable. A comparison to human perception, which need about 200 ms to perceive what audio actually contain, indicate that automatic audio classification need longer audio segments. This leads to a generalized scheme, which can be seen in Figure 6.1(c), that can summarize and calculate useful feature values on longer audio segments. Summarization is performed on analysis windows of length N_w, which determine how much information that is gathered from the audio clip. The length of the analysis window greatly affects the performance of the classification. If shorter analysis windows are used, better resolution is achieved but as shorter analysis windows are use less information is gathered to base predictions on. Typical summarizations are simply mean values and variances of specific features. Other summarizations can be seen as extracting the temporal characteristics of an audio segment, where the hzcrr feature is a good example. It actually extracts the typical evolvement of speech.

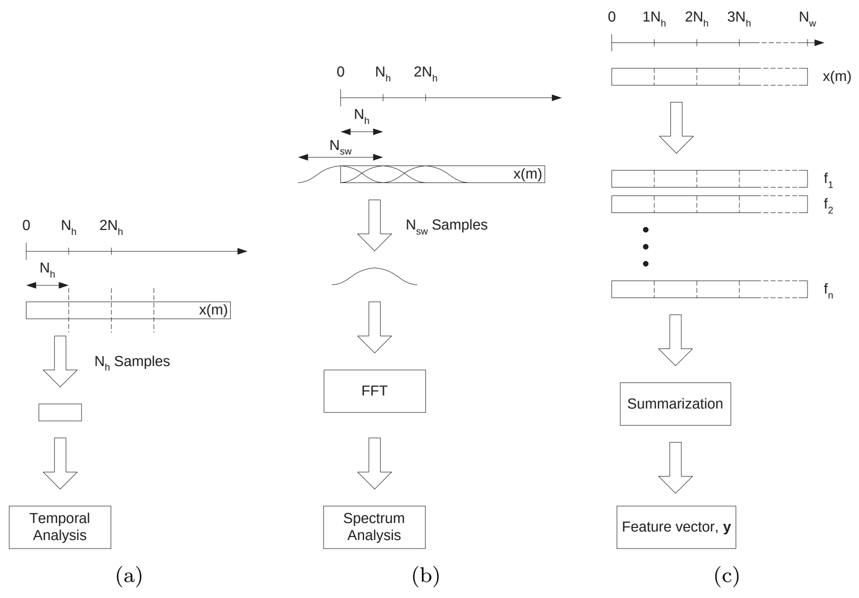


Figure 6.1: Feature extraction scheme from an audio sequence. In scheme (a), the methodology for temporal-based feature extraction is shown. At N_h samples long intervals, the audio samples are extracted and different features are calculated by temporal analysis. In scheme (b), the methodology for frequency-based feature extraction is shown. A sliding window is applied, centered at N_h samples long intervals. To follow the MPEG-7 Audio standard, the window is a Hamming-window of length N_{sw}. A FFT transform the time domain of the audio signal to the frequency domain, which can be analyzed to produce frequency-based features. In scheme (c), the summarization of feature values based on short time analysis is shown. Summarization is done to get informative feature values, as the mean and the variance of specific features, and is performed on different analysis window lengths N_w.

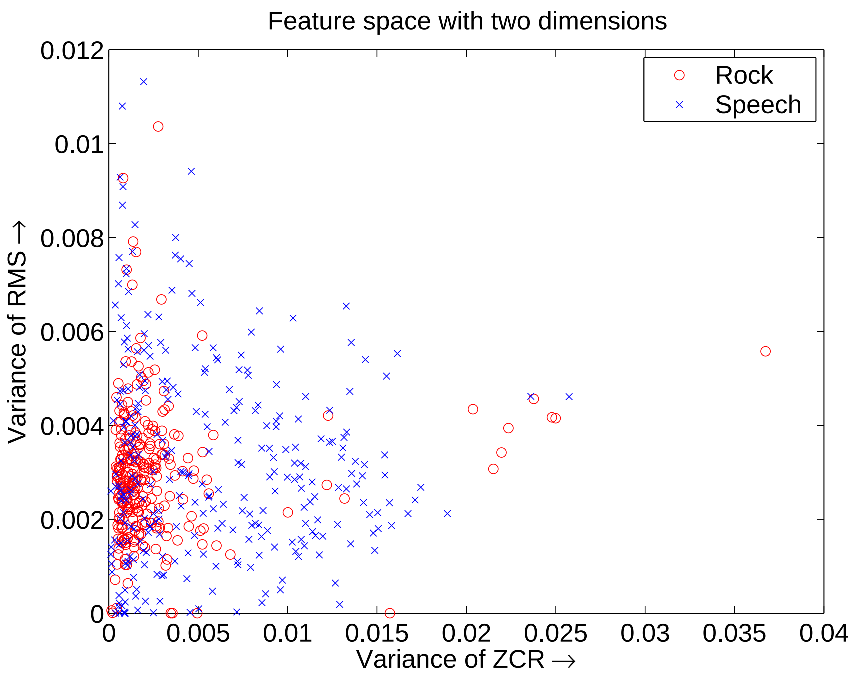


Figure 6.2: 2-dimensional feature space with variance of Root-Mean-Square on the y-axis and variance of Zero-Crossing-Rate on the x-axis. The points are extracted feature values from U2's "Sunday Bloody Sunday" and a speech recording from an audiotape. Although rock is slightly more concentrated to the left in the feature space, no decision boundary can be chosen to separate the two classes.

6.2.2 Feature Selection

To design an efficient speech/music classifier it is critical to compose the feature vector to be descriptive between the two classes. In Chapter 5, a number of features are proposed, which are summarized by calculation of mean values, variances and temporal characteristics to produce a total number of 24 different features. The key to make robust classification possible is to select a few of these features to compose an effective feature vector.

To exemplify this, consider two audio clips. One audio clip is an ordinary pop/rock song, U2's "Sunday Bloody Sunday", and the other is a speech recording from an audiobook. As the variation of the ZCR and the RMS can be discriminative between speech and music, the values for the two clips are shown in Figure 6.2. The figure shows the 2-dimensional feature space with variance of RMS on the y-axis and variance of the ZCR on the x-axis. It can be seen that points of the music clip is slightly more concentrated to the left than points of the speech clip. However, points from both categories are mixed and no decision boundary can be drawn to efficiently separate the two classes.

Another measure of feature values variance is the low feature-value ratio, which is especially designed for discrimination between speech and music. Figure

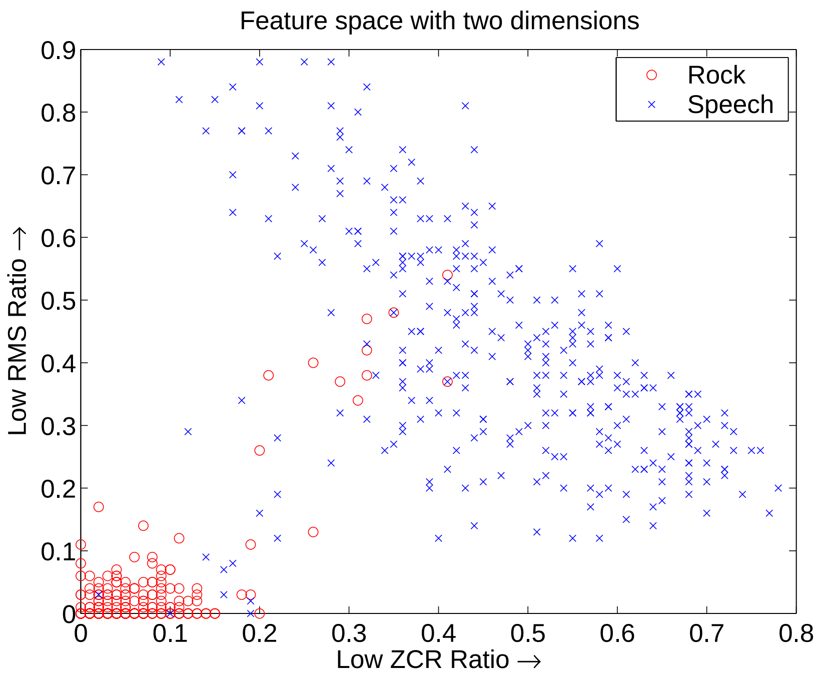


Figure 6.3: 2-dimensional feature space with low Root-Mean-Square ratio on the y-axis and low Zero-Crossing-Rate ratio on the x-axis. The points are extracted feature values from U2's "Sunday Bloody Sunday" and a speech recording from an audiotape. Most of the points from both classes are grouped and a decision boundary may be drawn to separate the two classes. Some points, both from speech and music, are however not clearly separable.

6.3 show the 2-dimensional feature space with LRMSR on the y-axis and LZCRR on the x-axis. These features result in a feature space where most of the points from both classes is grouped by themselves and a decision boundary may be drawn to separate the two classes. However, there are points from the speech clip down to the left in the feature space, where most of the music clip's points are. And there are points from the music clip in the middle of the feature space, where most of the speech clip's points are.

As Figure 6.3 show, most of extracted feature values may be separable by a decision boundary. There are however a few points from the music clip that is mixed with the points from the speech clip. The points correspond to the intro of the music clip, which is a steady drum beat with repetitive hi-hats. Hence, both the LRMSR and LZCRR features are inefficient to discriminate between drum patterns and speech. To be able to discriminate between pure drum patterns and speech, other feature combinations have to be considered. As the hi-hats in the drum intro have high frequencies, a feature as the spectral rolloff frequency may be efficient to discriminate between speech and music. Figure 6.4 show the 2-dimensional feature space with spectral rolloff frequency on the y-axis and LZCRR on the x-axis. Here, a decision boundary may be chosen to separate all points but two. For this case, a good classifier can be

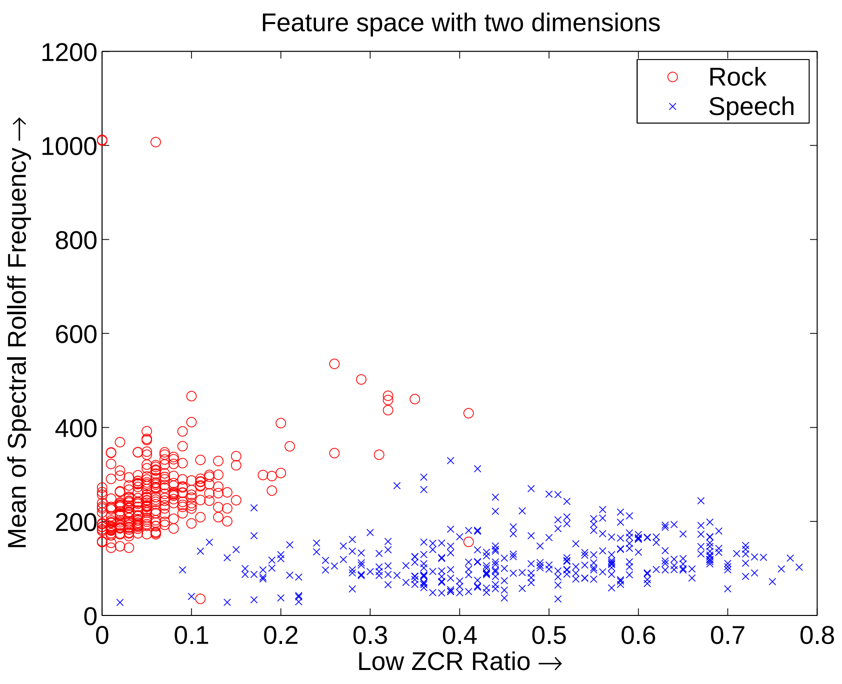


Figure 6.4: 2-dimensional feature space with spectral rolloff frequency on the y-axis and low Zero-Crossing-Rate ratio on the x-axis. The points are extracted feature values from U2's "Sunday Bloody Sunday" and a speech recording from an audiotape. In comparison to Figure 6.3, only two points from music are not separable from speech with a decision boundary.

designed with features as SRF and LZCRR.

Music is however a complex class of audio, which contain many different genres and, therefore, have many different characteristics. Good classification between speech and rock can be achieved with features as above. Robust classification between speech and music can however not be achieved with these features. In Figure 6.5 feature values from a music clip of jazz, Miles Davis' "Blue in Green", and the same speech clip as before are shown. The features values of SRF and LZCRR, which where efficient in discriminating between rock and speech, is shown to overlap each other. In this case, the two features are inefficient for discrimination between speech and music.

To design a robust speech/music classifier, different types of speech and music have to be considered when selecting features. The main characteristics that discriminate speech from music have to be used in the final composition of the feature vector. That is, a set of features have to be selected to compose the feature vector that best discriminate between speech and music. As a total of 24 different features are produced from the features in Chapter 5, efficient selection of features by looking at plots for a variety of music genres and speakers is not practical. It would be time consuming and, basically, unrealistic to accomplish any good results.

To select a set of features to compose a feature vector some guidance is

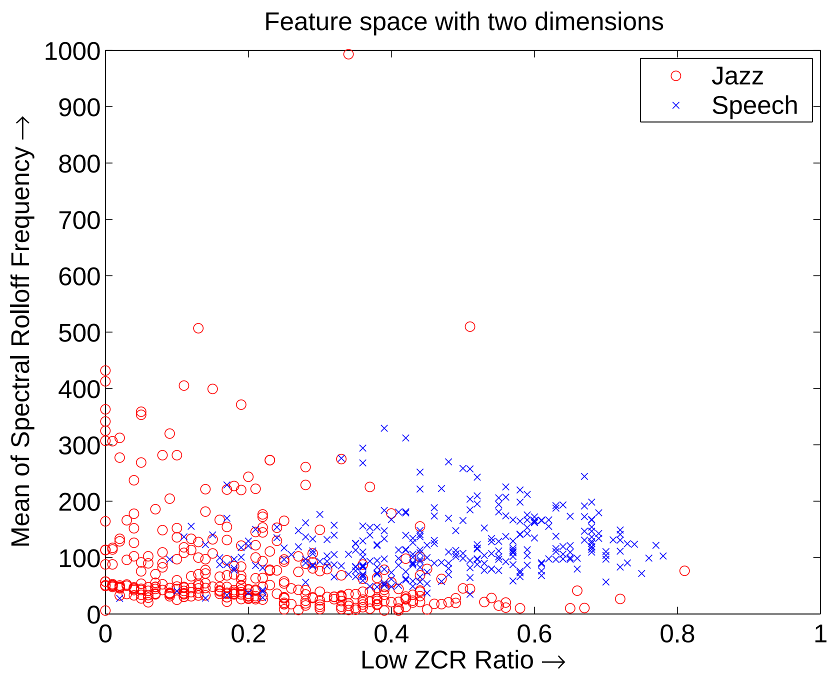


Figure 6.5: 2-dimensional feature space with spectral rolloff frequency on the y-axis and low Zero-Crossing-Rate ratio on the x-axis. The points are extracted feature values from Miles Davis' "Blue in Green" and a speech recording from an audiotape. In comparison to Figure 6.4, the feature values are mixed and no decision boundary can separate the two classes.

	$w = 0.5\,s$	$w = 1\,s$	$w = 5\,s$
Feature	Error	Error	Error
HZCRR	38.6 ± 2.5	36.3 ± 2.2	29.6 ± 3.9
HSTER	25.3 ± 0.7	33.7 ± 2.6	28.4 ± 2.4
HRMSR	18.6 ± 2.1	21.0 ± 2.7	5.5 ± 2.4
LZCRR	27.4 ± 1.2	23.3 ± 1.6	10.1 ± 2.7
LSTER	21.1 ± 2.2	20.3 ± 1.5	13.3 ± 3.6
LRMSR	15.4 ± 2.6	14.7 ± 2.1	6.1 ± 2.1
meanZCR	49.2 ± 1.1	48.6 ± 1.8	41.4 ± 3.0
meanSTE	40.0 ± 1.7	40.6 ± 1.8	30.5 ± 2.7
meanRMS	36.8 ± 1.1	35.7 ± 1.5	25.1 ± 3.5
meanSRF	48.5 ± 1.0	48.7 ± 2.1	42.1 ± 3.0
meanSS	48.3 ± 1.6	49.1 ± 1.5	41.5 ± 4.1
meanSC	43.9 ± 1.2	43.3 ± 1.6	33.8 ± 1.9
meanSF	33.4 ± 2.3	30.4 ± 0.7	19.1 ± 4.3
meanSS MPEG7	38.1 ± 1.6	40.2 ± 1.8	36.0 ± 4.7
meanSC MPEG7	48.3 ± 1.0	46.1 ± 2.2	39.0 ± 2.1
meanSF MPEG7	18.2 ± 0.9	12.6 ± 1.2	5.8 ± 2.0
varZCR	37.4 ± 1.1	30.2 ± 1.6	16.6 ± 2.0
varSTE	45.7 ± 1.2	45.9 ± 1.5	45.7 ± 3.9
varRMS	42.7 ± 0.9	40.0 ± 2.1	30.6 ± 2.3
varSRF	36.2 ± 1.0	30.9 ± 1.6	20.3 ± 3.0
varSS	42.4 ± 1.1	37.0 ± 2.6	25.5 ± 4.5
varSC	37.1 ± 1.8	29.4 ± 2.3	10.4 ± 2.1
varSF	40.5 ± 1.2	34.5 ± 1.7	22.4 ± 2.4
varSS MPEG7	31.8 ± 1.5	23.9 ± 1.6	12.2 ± 1.9
varSC MPEG7	30.8 ± 1.1	24.1 ± 1.8	12.5 ± 2.5
varSF MPEG7	24.1 ± 0.8	13.9 ± 1.0	5.6 ± 1.5

Table 6.1: Evaluation of feature's efficiency for speech/music classification. The table show 10-fold cross validation error rates for individual features using a Nearest Neighbor classifier. Classification is done for analysis length of 0.5, 1 and 5 seconds to see how feature's efficiency depends on different analysis length.

needed. The method proposed by Scheirer [4], which is described in Section 4.3, is used to indicate how effective each feature is. This gives guidance how to compose the feature vector. In this case a Nearest Neighbor algorithm is used, that is a k-Nearest Neighbor algorithm with k set to 1. Different analysis window lengths are also used to see how each feature's effectiveness depend on the analysis time. The results can be seen in table 6.1.

Based on the values in Table 6.1 the LRMSR, meanSF MPEG7 and varSF MPEG7 features are selected to compose a set of features, from now on called the *BEST3*. These give relatively good classification accuracy alone and should perform even better combined. The BEST3 feature set contains features based on the spectrum of an audio clip, which is relatively complex to calculate. It is interesting to compose a *FAST3* set that only contain features extracted in the time domain. Therefore, a FAST3 set is composed by the computationally efficient features LZCRR, LRMSR and meanRMS. Two additional feature sets are composed to see if more features than 3 give better results. These are called *FAST3+BEST3*, which contain the features in FAST3 and BEST3, and *BEST6*, which is composed by the LZCRR, LRMSR, meanSF MPEG7, varSF MPEG7, varSC MPEG7 and varSS MPEG7 features.

6.2.3 Classification Algorithm

Only one classification algorithm is studied due to time limitations. However, the chosen algorithm is a commonly used classification algorithm that can produce complex decision boundaries. The classifier is based on a k-Nearest Neighbor algorithm and this section describes how features are combined in the classification algorithm and what parameter setting that is chosen.

The k-Nearest Neighbor algorithm is sensitive to how the features are combined. As the k-Nearest Neighbor algorithm's predictions are based on Euclidean distance measurements, by Equation 4.16, the range in which feature values are is important. If, for instance, a distance measurement is performed on the feature values shown in Figure 6.4, the distance measure would only reflect distances along the y-axis. In the figure it clearly can be seen that the feature values along the y-axis is much bigger then the feature values on the x-axis, which make the feature along the x-axis irrelevant in the distance measurement.

To combine features, the feature values have to be weighted to suit the Euclidean Space, $\Re^n$. Normalization of each axis in the feature space may be used to produce a feature space with feature values in the range from 0 to 1. This can however result in that some features become less important in the distance measurement, which the case would be if normalization of feature values would be used on Figure 6.4. Better is to normalize the standard deviation for each axis, which make each feature important and a suitable feature space is achieved. In Figure 6.6, the feature space from Figure 6.4 with normalized standard deviation feature values are shown.

The k-Nearest Neighbor only has one parameter to adjust, which is the k-value. Scheirer conclude in his work [4] that varying k in the k-Nearest Neighbor algorithm make only very little difference in classification performance. To validate this, a cross-validation on the training set is performed using an analysis window of 1 second and the feature set BEST3 for different values of k. The classification accuracy for different k-values is shown in Table 6.7, where it can be seen that the variation in performance is small. Based on Scheier's work and the obtained results, k is chosen to be 3 in the final classifier.

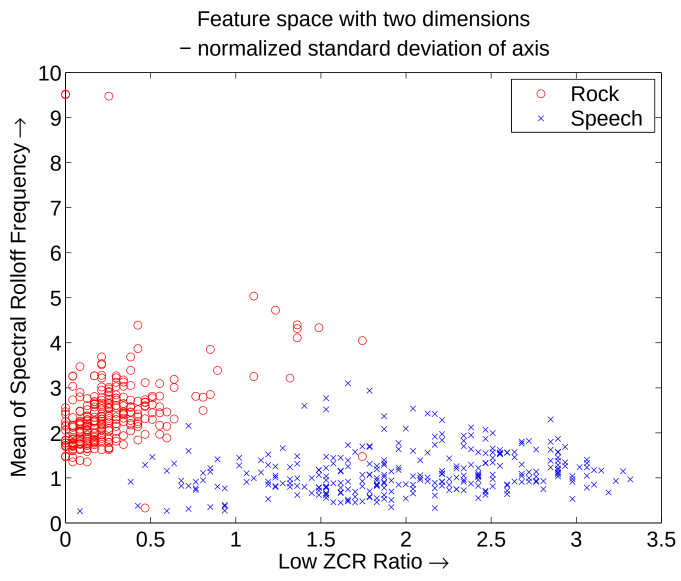


Figure 6.6: 2-dimensional feature space with spectral rolloff frequency on the y-axis and low Zero-Crossing-Rate ratio on the x-axis. The standard deviation of the feature values are normalized to get a well weighted feature space.

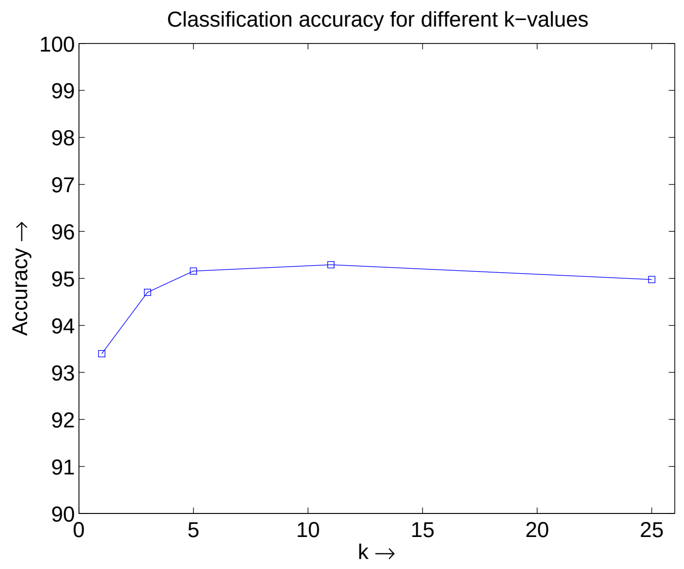


Figure 6.7: 10-fold cross validation classification accuracies for BEST3 feature set and an analysis window of 1 second. The accuracies are evaluated for different k-values of 1, 3, 5, 11 and 25. It can be seen that there is a slight difference in performance for different k.

6.3 Performance Evaluation

Four different feature vectors are composed and named the FAST3, BEST3, FAST3+BEST3 and BEST6 feature set. The performance of the speech/music classifier can be seen in Table 6.2. Many interesting facts about classification between speech and music can be read from this table. Overall classification accuracy for different feature sets and analysis window lengths are shown. Also, classification accuracy for specifically speech and music is shown.

The conclusion is that the BEST3 set has overall best performance, with a classification accuracy of over 93 % for all analysis window lengths over 0.5 seconds. The FAST3 and the FAST3+BEST3 also perform well with accuracy over 92 % for all analysis window lengths over 0.5 seconds. A performance drop can however be seen for the BEST6 set, which only have an accuracy of 86.9 % for the analysis window lengths of 0.5 seconds. Classification accuracy for analysis window lengths of 0.1 seconds is poor for all feature sets. The best performance for analysis window lengths of 0.1 seconds is 82.7 %.

Another conclusion is that speech is easier than music to classify correctly. This is expected as music is, in general, a broader class that has more variations of it's composition than speech. Music can have a variety of instruments, moods, et cetera whereas speech is somewhat limited. Of course, speech can also be expressed in a variety of ways but is nonetheless limited by the nature of speech. It has to be mentioned that the results may come from a badly composed database, but considerations was made when composing the database.

Finally, satisfactory classification accuracy is achieved for longer analysis window lengths. The BEST3 stands out with a classification accuracy of 98.2 % for analysis window length of 5 seconds and 97.8 % for analysis window length of 2 seconds.

Classifier	$w = 0.1\,s$	$w = 0.5\,s$	$w = 1\,s$	$w = 2\,s$	$w = 5\,s$
BEST3					
Total accuracy	**79.0**	**93.1**	**95.7**	**97.8**	**98.2**
Music accuracy	78.0	89.7	93.4	96.8	97.1
Speech accuracy	80.0	96.6	98.1	98.8	99.4
FAST3					
Total accuracy	**79.0**	**92.5**	**92.9**	**95.6**	**93.3**
Music accuracy	75.6	88.1	87.2	92.2	87.9
Speech accuracy	82.2	97.0	98.7	99.0	98.8
FAST3+BEST3					
Total accuracy	**82,7**	**92.5**	**93.0**	**95.5**	**93.8**
Music accuracy	78.4	88.1	87.4	92.0	87.7
Speech accuracy	86.8	96.9	98.7	99.0	100
BEST6					
Total accuracy	**75.0**	**86.9**	**92.0**	**93.3**	**96.3**
Music accuracy	70.9	80.9	85.9	87.8	92.7
Speech accuracy	78.9	93.0	98.2	99.0	100

Table 6.2: Classification accuracy for different feature sets: BEST3, FAST3, FAST3+BEST3 and BEST6. Overall classification accuracy, classification accuracy for music and classification accuracy for speech is presented. The analysis window length, w, is set to 0.1, 0.5, 1, 2 and 5 seconds. Evaluation is performed on a testing set of about 1 hour of data divided into approximately a 50/50 ratio of speech and music. A k-Nearest Neighbor algorithm is used with k set to 3.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

Content description is a broad field that involve many techniques that aim to provide tools for content management. Classification, segmentation, recognition and summarization is examples on techniques being developed to aid users in media applications. A study of automatic audio content classification for was performed in this thesis and a test-bed in Matlab was implemented for testing content classification algorithms.

Audio classification is complex and many problems have to be considered when a classifier is designed. First, knowledge of what content the classifier should classify between is essential for successful classification. This leads to a hierarchical audio classification scheme to make limitations of possible content to be classified. The hierarchical structure allows efficient design of classifiers at each level. As speech/music commonly is on the top of the hierarchical classification structure, a speech/music classifier have been implemented and analyzed in this thesis.

Second, features have to be selected that maximizes the discrimination between different classes. This is a problem that can be found in all classification tasks. Audio classification is often performed by extracting such features as zero-crossing rate, short-term energy and different spectrum measurements. In this thesis the performance of commonly used features, and variations of them, have been studied. Interesting to see is the efficiency of features based on the MPEG-7 spectrum descriptor. The main difference between an ordinary frequency spectrum and MPEG-7 spectrum descriptor is that the frequency axis is mapped to a logarithmic frequency axis in MPEG-7. As human perception is logarithmic, with respect to frequencies, the MPEG-7 spectrum descriptor can be seen to represent a "crude auralization" of audio. This results in that the efficiency of all spectrum based features is outperformed by ones based on the MPEG-7 spectrum descriptor. Other efficient features are the LZCRR, LRMSR and meanRMS that all are variations of commonly used features. This indicates that no ultimate feature has been proposed and with more research better features may be designed.

Third, a classification algorithm has to be implemented that make predictions of new samples based on extracted feature values. This can be done in several ways but this thesis only studies the k-Nearest Neighbor algorithm. The k-Nearest Neighbor is sensitive to irrelevant features and feature values. If irrelevant features are considered when classification is performed, the classification performance degrades. This can be seen in table 6.2, where the classification accuracy of different feature sets is shown. Higher dimensional feature sets as FAST3+BEST3 and BEST6 perform, in general, worse than lower dimensional feature sets. If irrelevant feature values are used, the relationship measurement that is used for the k-Nearest Neighbor's predictions may be inefficient. For instance, if two features with values around 3 and 400 are to compose the feature vector and used in a k-Nearest Neighbor classifier the first feature will not contribute much to the final prediction. In this thesis, the k-Nearest Neighbor normalizes the variance of the feature values to make each feature equally important. This seems to work well and good performance is achieved.

Fourth, the amount of information that is gathered to make predictions of new samples highly determines the performance of the classification. In table 6.2 classification accuracy for different analysis window lengths are listed. It is shown that performance, in general, is better for longer analysis lengths. As more information about the audio is gathered it is logical to see an improvement as analysis lengths increase. Hence, a trade-off between resolution in the classification and classification accuracy has to be done. However, table 6.2 also show that as the analysis length reaches 5 seconds the performance drop. This drop in performance may come from badly designed features that only work for shorter analysis lengths. Another possibility is that the performance drop is related to the number of instances in the training set. When a longer analysis window is used, less training samples is generated for the same training set of audio. As the k-Nearest Neighbor is an instance-based classifier, performance drop as the number of instances decline.

And finally, as some audio classes are more complex than others, the overall classification accuracy of a classifier may not be indicative to how well the classifier performs for a given class. As the results show for the speech/music classifier, speech is easier to classify correctly compared to music. Music is a broader class that contain a variety of compositions and this has to be considered when training and testing a classifier.

7.2 Future Work

Many techniques have not been studied closer in this thesis and more work can be done. Below is a list of possible directions to continue evaluating and designing new classifiers.

- **Implement more efficient features.** To achieve better classification accuracy, more efficient features may be designed. For example, extraction of possible tempo in an audio clips, stereo information and better use of perceptual models (as the MPEG-7 spectrum representation) can result in more efficient features. Better features can also be designed to work for music genre classification, speaker classification et cetera.

- **Implement more efficient classifiers.** Although many research groups conclude that the difference in performance between different classification algorithms is limited, much of the computational complexity can be improved. Efficient searches in the k-Nearest Neighbor algorithm may be implemented but to implement completely other classification algorithms, as the GMM or HMM, is also interesting.

- **Use other training sets.** A better selection of training material can improve classification accuracy. If more audio clips are used to train the classifier, better performance should be achieved. The audio content of the clips can also be improved to be represent a diverse set of genres of speech and music.

- **Implement automatic content segmentation.** To improve classification results at transitions between different content in audio clips, automatic content segmentation may be used. Good automatic segmentation would increase classification performance as analysis windows could be positioned to only contain one class.

- **Implement more classifiers.** As relatively good accuracy is achieved in the implemented speech/music classifier, a natural step is to implement more levels in the hierarchical classification structure. When audio is classified as speech, a classifier can be used to classify the speaker to be either female or male. And as the audio is classified as music, a classifier can be used to classify between musical genres.

These are some proposed steps that can be studied with help of the implemented test-bed and give more knowledge of automatic audio classification and content description.

Bibliography

[1] L. Lu, H.-J. Zhang, and H. Jiang. Content analysis for audio classification and segmentation. *IEEE Transaction on Speech and Audio Processing*, 10(7):504–516, October 2002.

[2] D. Li, I. K. Sethi, N. Dimitrova, and T. McGee. Classification of general audio data for content-based retrieval. *Pattern Recognition Letters*, 22(5):533–544, April 2001.

[3] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, July 2002.

[4] E. Scheirer and M. Slaney. Construction and evaluation of a robust multi-feature speech/music discriminator. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2:p 1331–1334, April 1997.

[5] K.-P. Han, Y.-S. Park, S.-G. Jeon, G.-C. Lee, and Y.-H Ha. Genre classification system of tv sound signals based on a spectrogram analysis. *IEEE Transactions on Consumer Electronics*, 44(1):33–42, February 1998.

[6] D. Pye. Content-based methods for the management of digital music. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 4:2437–2440, June 2000.

[7] D.-N. Jiang, L. Lu, H.-J. Zhang, J.-Hua. Tao, and L.-H. Cai. Music type classification by spectral contrast feature. *Proceedings 2002 IEEE International Conference on Multimedia and Expo (Cat. No.02TH8604)*, 1:113–116, August 2002. Conference date 26-29 Aug. 2002.

[8] D. Kimber and L. Wilcox. Acoustic segmentation for audio browsers. *Computing Science and Statistics. Vol.28. Graph-Image-Vision. Proceedings of the 28th Symposium on the Interface. Interface '96*, (295-304), 1997.

[9] A. Albiol, L. Torres, and E. J. Delp. The indexing of persons in news sequences using audio-visual data. *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '03*, 3:137–140, April 2003.

[10] M. Abe and M. Nishiguchi. Self-optimized spectral correlation method for background music identification. *Proceedings 2002 IEEE International Conference on Multimedia and Expo (Cat. No.02TH8604)*, 1(1):333–336, August 2002.

[11] L. R. Rabiner. Tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[12] J.-T. Chien. Linear regression based bayesian predictive classification for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 11(1):70–79, January 2003.

[13] J. Chen, K. K. Paliwal, and S. Nakamura. Cepstrum derived from differentiated power spectrum for robust speech recognition. *Speech Communication*, 41(2-3):469–484, October 2003.

[14] F. Gouyon, F. Pachet, and O. Delerue. On the use of zero-crossing rate for an application of clasification of percussive sounds. *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00)*, December 2000.

[15] D. Slezak, P. Synak, A. Wieczorkowska, and J. Wróblewski. Kdd-based approach to musical instrument sound recognition. *Foundations of Intelligent Systems. 13th International Symposium, ISMIS 2002. Proceedings (Lecture Notes in Computer Science Vol.2366)*, pages 28–36, June 2002.

[16] D. Slezak, P. Synak, A. Wieczorkowska, and J. Wróblewski. Application of temporal descriptors to musical instrument sound recognition. *Journal of Intelligent Information Systems*, 21(1):71–93, July 2003.

[17] E. Wold, T. Blum, D. Keislar, and J. Wheaton. Content-based classification, search, and retrieval of audio. *IEEE Multimedia*, 3(3):27–36, 1996.

[18] M. Casey. *Sound Classification and Similarity Tools.* Wiley, 2002. B. S. Manjunath (Editor), P. Salembier (Editor), T. Sikora (Editor), ISBN: 0-471-48678-7.

[19] B. Logan and S. Chu. Music summarization using key phrases. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2:749–752, June 2000.

[20] C. Xu, Y. Zhu, and Q. Tian. Automatic music summarization based on temporal, spectral and cepstral features. *Proceedings 2002 IEEE International Conference on Multimedia and Expo (Cat. No.02TH8604)*, 1(1):117–120, August 2002.

[21] J. Song, S.-Y Bae, and K. Yoon. Query by humming: matching humming query to polyphonic audio. *Proceedings 2002 IEEE International Conference on Multimedia and Expo (Cat. No.02TH8604)*, 1(1):329–332, August 2002.

[22] B. Liu, Y. Wu, and Y. Li. A linear hidden markov model for music information retrieval based on humming. *Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '03*, 5:533–536, 2003.

[23] Mpeg-7 overview (version 9). *ISO/IEC JTC1/SC29/WG11 N5525*, March 2003.

[24] Text of international standard iso/iec 15938-4 information technology - multimedia content description interface - part 4: Audio. *ISO/IEC 15938-4*, 2002.

[25] M. A. Casey. Reduced-rank spectra and minimum-entropy priors as consistent and reliable cues for generalized sound recognition. *Proceedings of the Workshop on Consistent and Reliable Cues for Sound Analysis, EUROSPEECH 2001*, September 2001.

[26] T. M. Mitchell. *Machine learning*. McGraw-Hill, New York, 1997. ISBN: 0-07-042807-7.

[27] R. O. Duda, P. E. Hart, and David G. Stork. *Pattern classification*. Wiley, New York, 2 ed. edition, 2001. ISBN: 0-471-05669-3.

[28] L. Lu, H.-J. Zhang, and S. Z. Li. Content-based audio classification and segmentation by using support vector machines. *Multimedia Systems*, 8(6):482–492, April 2003.

[29] D. A. Reynolds and R. C. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing*, 3(1):72 – 83, January 1995.

Appendix A

Test-bed Environment

This appendix will describe the test-bed implemented as part of this thesis. The test-bed is implemented in Matlab and aim to be a general tool for evaluation of classification schemes. A description of the test-bed's structure and how to use it is outlined below.

A.1 Structure

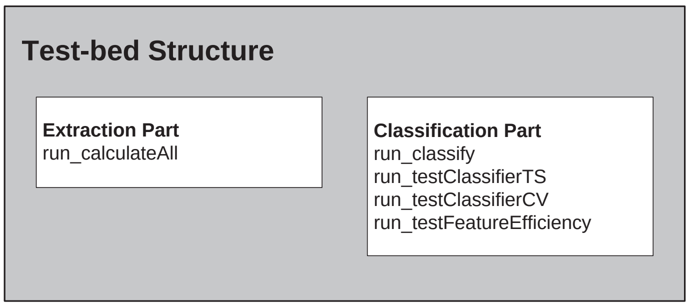


Figure A.1: Overview of the test-bed's structure.

The test-bed is initialized with a file called startup.m. It sets all needed paths and make the test-bed ready to use. The test-bed mainly consist of two parts named *extraction part* and *classification part*. Figure A.1 show the two parts and their m-files, which make testing of different classification schemes possible. The extraction part is a general framework for extraction of different features. It allows new features to be implemented and used. The classification part use different classification algorithms to classify test data. Different evaluation methods may be used to estimate classifiers performance.

Extraction Part

In the extraction part there exists a file called *calculateAll.m* that extracts feature values for all files in a sorted database. Different parameters are passed to calculateAll.m, which determines what features and how they will be extracted.

Below are the input variables for calculateAll.m shown:

- **directory** Directory in which all files are located. For instance, the root of a sorted database.
- **extension** Optional formats: wav
- **windowSizeGrp** A struct with desired hop length of analysis and optional window size. hopSize is used to set desired hop length of analysis and windowSize can be used to determine summarization lengths. Entries: hopSize, windowSize
- **featureGrp** A struct with desired features for given clip. Entries: my.LLD1, LLD2 ... LLDN and mpeg7.LLD1, LLD2 ... LLDN
- **attributeGrp** For MPEG-7 descriptors. Struct with attributes for spectral descriptors. Entries: octaveResolution, loEdge and hiEdge
- **writeResults** Flag that determine if storage of feature values shall be done.
- **rootPathsGrp** Contain paths to the root of the sample/result databases. Entries: databaseRootPath and storeRootPath

Classification Part

Common for classifiers is that they need a training set and a testing set. The classification part contain a *buildTrain* and a *buildTest* function that given a set of parameters build a training set and a test set respectively. The inputs and outputs of these two functions are described below.

Below are the input variables for builtTrain.m shown:

- **directories** A struct with sorted directories in which all files to be part of the training set are located. Directories.class should be of size nClasses x 1 Entries: directories.path and directories.class
- **windowSizeGrp** As described above.
- **featureGrp** As described above.
- **attributeGrp** As described above.

And the output variables for builtTrain.m:

- **trainData** Data that forms the training set.
- **trainLabels** Labels to the training samples.

The classification part also consist of four m-files that allow classification with different classifiers. Descriptions of the four m-files *run_classify.m*, *run_testClassifierTS*, *run_testClassifierCV* and *run_testFeatureEfficiency.m* are found in section A.5.

A.2 How to build a database

To simplify the evaluation procedure, the audio database should be sorted in a training set and a testing set. This is not necessary but it is advisable to do so. If all audio clips are stored in a sorted training and testing set all calculated feature values will be stored in the same sorted way. That is, a sorted feature value database is achieved automatically.

Another practical issue about the test-bed is that the audio clips should be relatively short. The implemented feature extraction solution does not handle memory efficiently, which causes Matlab to run slow. The analysis in this thesis was performed with clips of 30 seconds of audio. Hence, audio clips of 30 seconds are proposed to be used. A m-file to called *splitFiles* can be used to split all files in a sorted database.

Below is a numbered list that to guide users how to build a database:

1. Sample audio and store the audio clips in a sorted database.
2. If the audio files are long, use *run_splitFiles.m* to create a new database with shorter clip lengths. This limits inefficient memory use in the feature extraction process.
3. Use *calculateAll.m* to calculate desired features. Feature values is stored in a sorted database at a desired place.

A.3 How to add new features

Implementation of new features has to follow the framework's passing of parameters. That is, a new feature has to have the following input fields:

- **audioSampleGrp** A struct with information about audio clip to analyze. Entries: fileName, audio, size, sr and nbits
- **windowSizeGrp** As described above.
- **attributeGrp** As described above.

and return the result as follows:

- **Result** A vector of size NxM, where N is the dimensionality of the feature and M is the time index for feature values.

Below is a numbered list to guide users how to add new features:

1. Write a m-file with input and output arguments as shown above.
2. Store the m-file in directory "testbed\extraction\Feature_Analysis... ...\NEWFEATURENAME\"
3. As the new feature is stored in Feature_Analysis. Startup.m can be executed to set all paths for the test-bed, which allow the new feature to be used.
4. Finally, run_calculateAll.m is modified to work as wanted.

A.4 How to add new classifiers

Implementation of new classifiers has to follow the framework's passing of parameters. That is, a new classifiers must have the following input fields:

- **testData** Data to be classified.
- **trainData** Data that forms the training set.
- **trainLabels** Labels to training samples.
- **paramaters** Classifier specific parameters.

and return the result as follows:

- **predictions** Predicted class labels for testData.

Below is a numbered list to guide users how to add new features:

1. Write a m-file with input and output arguments as shown above.
2. Store the m-file in directory "testbed\classification...
   ...\CLASSIFIERNAME\"
3. As the new feature is stored in "classification". Startup.m can be executed to set all paths for the test-bed, which allow the new classifier to be used.

A.5 How to test new classification schemes

When features and classification algorithms are implemented they can be combined to form complete classification schemes. Evaluation of classification schemes can be done in several ways and the test-bed consists of four functions to evaluate classifiers performance.

Below is a list of available functions and a short description of their functionality:

- **run_classify.m** Defines all settings used in *classify.m* and runs it. This function makes predications on a single audio clip for a defined classifier. Possibility to change audio clip, training set and classifier.
- **run_testClassifierTS** Defines all settings used in *testclassifierTS.m* and runs it. This function calculates the classification accuracy of a classifier for a given training and testing set. Possibility to change training set, testing set and classifier.
- **run_testClassifierCV** Defines all settings used in *testclassifierCV.m* and runs it. This function approximates the classification accuracy of a classifier with n-fold cross-validation. Possibility to change training set, n in "n-fold" and classifier.
- **run_testFeatureEfficiency.m** Defines all settings used in *testFeatureEfficiency.m* and runs it. It approximates individual features efficiency by n-fold cross-validation. Possibility to change training set, n in "n-fold" and classifier.

Appendix B

Abbreviations

BEST3	Feature set composed by Low Root-Mean-Square Ratio, mean and variance of Delta Spectrum based on MPEG-7 spectrum Descriptor
BEST6	Feature set composed by Low Root-Mean-Square Ratio, Low Zero-Crossing-Rate Ratio, mean and variance of Delta Spectrum, Spectrum Centroid and Spectrum Spread. All spectrum based features are based on the MPEG-7 spectrum Descriptor.
CD	Compact Disc
D	MPEG-7 Descriptor
DFT	Discrete Fourier Transform
DS	MPEG-7 Description scheme
FAST3	Feature set composed by Low Root-Mean-Square Ratio, Low Zero-Crossing-Rate and mean of Root-Mean-Square.
FAST3+BEST3	Feature set composed by features in FAST3 and BEST3. The set is Low Root-Mean-Square Ratio, Low Zero-Crossing-Rate Ratio, mean of Root-Mean-Square and mean and variance of Delta Spectrum based on MPEG-7 spectrum Descriptor
FFT	Fast Fourier Transform
GMM	Gaussian Mixture Model
HFVR	High Feature-Value Ratio
HMM	Hidden Markov Models
HRMSR	High Root-Mean-Square Ratio
HSTER	High Short-Time-Energy Ratio
HZCRR	High Zero-Crossing-Rate Ratio
ISO/IEC	The International Organization for Standardization and the International Electrotechnical Commission
kNN	k-Nearest Neighbor
LFVR	Low Feature-Value Ratio
LLD	Low Level Descriptor
LRMSR	Low Root-Mean-Square Ratio
LSTER	Low Short-Time-Energy Ratio
LZCRR	Low Zero-Crossing-Rate Ratio
meanRMS	Mean of Root-Mean-Square in an analysis window

meanSC	Mean of Spectrum Centroid in an analysis window
meanSC MPEG-7	Mean of the MPEG-7 Descriptor SpectrumCentroidD in an analysis window
meanSF	Mean of Delta Spectrum (or Spectrum Flux) in an analysis window
meanSF MPEG-7	Mean of the Delta Spectrum (or Spectrum Flux) in an analysis window. Based on the MPEG-7 spectrum Descriptor
meanSRF	Mean of Spectrum Rolloff Frequency in an analysis window
meanSS	Mean of Spectrum Spread in an analysis window
meanSS MPEG-7	Mean of the MPEG-7 Descriptor SpectrumSpreadD in an analysis window
meanSTE	Mean of Short-Time-Energy in an analysis window
meanZCR	Mean of Zero-Crossing-Rate in an analysis window
MIDI	Musical Instrument Digital Interface
MPEG	Moving Picture Experts Group
MPEG-7	The international standard named "Multimedia Content Description Interface"
QBH	Query-by-Humming
PCA	Principal Component Analysis
RMS	Root-Mean-Square
SC	Spectrum Centroid
SF	Delta Spectrum or Spectrum Flux
SRF	Spectral Rolloff Frequency
SS	Spectrum Spread
SNR	Signal-to-Noise Ratio
STE	Short-Time-Energy
varRMS	Variance of Root-Mean-Square in an analysis window
varSC	Variance of Spectrum Centroid in an analysis window
varSC MPEG-7	Variance of the MPEG-7 Descriptor SpectrumCentroidD in an analysis window. Based on the MPEG-7 spectrum Descriptor.
varSF	Variance of Delta Spectrum (or Spectrum Flux) in an analysis window
varSF MPEG-7	Variance of the Delta Spectrum (or Spectrum Flux) in an analysis window. Based on the MPEG-7 spectrum Descriptor.
varSRF	Variance of Spectrum Rolloff Frequency in an analysis window
varSS	Variance of Spectrum Spread in an analysis window
varSS MPEG-7	Variance of the MPEG-7 Descriptor SpectrumSpreadD in an analysis window. Based on the MPEG-7 spectrum Descriptor.
varSTE	Variance of Short-Time-Energy in an analysis window
varZCR	Variance of Zero-Crossing-Rate in an analysis window
ZCR	Zero-Crossing-Rate