Automatic speech/music discrimination in audio files

Lars Ericsson

Master's thesis in Music Acoustics (30 ECTS credits)
at the School of Media Technology
Royal Institute of Technology year 2009
Supervisor at SR was Björn Carlsson
Supervisor at CSC was Anders Friberg
Examiner was Sten Ternström

Automatic speech/music discrimination in audio files

Abstract

This master's thesis presents an algorithm for discrimination of speech and music in audio files. The algorithm is made for Swedish Radio and is therefore suited for their needs and optimized for their material. A series of tests on acoustic features extracted from Swedish Radio broadcasts was performed to distinguish the best method for the discrimination. These tests were evaluated to find the most suited method for the task. Methods based on RMS amplitude of the signal were chosen for both classification and segmentation. A feature for the proportion of low energy that uses the small pauses that can tell speech from music was used for the classification and a similarity measure based on mean and variance of the RMS was used to find the transition points for the segments. This resulted in an algorithm that with an accuracy of 97,3% can discriminate speech from music in Swedish Radio broadcasts.

Automatisk diskriminering av tal och musik i ljudfiler

Sammanfattning

Detta examensarbete presenterar en algoritm för diskriminering av tal och musik i ljudfiler. Algoritmen är gjord för Sveriges Radio och är därför utformad utefter deras behov och optimerad för deras material. För att komma fram till den bästa metoden för diskrimineringen gjordes tester på akustiska parametrar som extraherats ur material från SR:s digitala arkiv. Dessa tester utvärderades sedan för att hitta den metod som var bäst lämpad för ändamålet. För både klassificering och segmentering valdes metoder som baseras på RMS för amplituden av signalen. En parameter för lågnivåproportioner som utnyttjar de korta pauserna som skiljer tal från musik valdes för att klassificera segment och ett likhetsmått baserad på medelvärde och varians av RMS valdes för att hitta gränserna till segmenten. Detta resulterade i en algoritm som med 97,3% träffsäkerhet kan diskriminera tal från musik i SR:s programutbud.

Contents

List of Abbreviations

DCS	Discrete Cosine Function
ERB	Equivalent Rectangular Bandwidth
FFT	Fast Fourier Transform
LFCC	Linear Frequency scaled Cepstrum Coefficients
LPC	Linear Predictive Coding
MFCC	Mel Frequency Cepstrum Coefficients
MLER	Modified Low Energy Ratio
RMS	Root Mean Square
SC	Spectral Centroid
SLL	Standard Low Level (features)
SMD	Speech/Music Discrimination
SR	Swedish Radio
ZCR	Zero Crossing Rate

1 Introduction

This chapter includes an overview of the task, the purpose, method and limitations of the work. It also gives a brief view of the rest of the report.

1.1 Speech/Music discrimination

The purpose of speech/music discrimination (SMD) systems is to divide audio to music and speech segments and classify them, whether the discrimination is done in real time or on recorded audio files.

The Speech/Music Discrimination task is an important part of Automatic Speech Recognition (ASR) systems, where it is used to disable the ASR when music or other classes of audio are present in automatic transcription of speech.

SMD systems are also useful for bit-rate coders. Speech coders achieve better results for speech coding than music coders do, and vice versa, therefore it is important to discriminate between the two audio classes, to select the right type of bit-rate coding.

When indexing sound and even video a SMD system can be of great importance. Apart from using the detection of speech and music in audio files, the same algorithm can be applied to the audio of TV shows or movies. This can later be used for indexing the video material to be able to jump straight to a desired part in e.g. on-demand video solutions for the web.

1.2 Swedish Radio

SR is a non-commercial, public service radio broadcaster with over 40 radio channels, including four national FM channels (P1, P2, P3 and P4) and 28 local channels. P4 is the biggest radio channel in Sweden. SR is also offering more than 10 channels online on their website, together with an archive of all broadcasted programs available on demand. Broadcasts are also available via shortwave, medium wave and satellite. [12]

1.3 Goal

A system that effectively discriminated between speech and music will be useful for Swedish Radio in many of the above mentioned applications.
The goal is therefore to create an algorithm that can do the task accurately for the material produced by Swedish Radio. To meet the requirements of a cost efficient, accurate algorithm, an offline system structure was chosen.

1.4 Method

The project started by reviewing necessary literature and articles of previous work. This made it possible to enter the test phase with knowledge of the area. In this phase many features and discrimination methods were tested and evaluated to be able to find the method best suited for this specific task. The chosen method was then coded in a way that permitted an integration into existing systems at Swedish Radio.

1.5 Limitations

This work focused on the discrimination between speech and music, and did not a finer discrimination within the classes. Thus, the algorithm will not be able to discriminate between different speakers such as women, men or children and will not be able to tell different voices apart. Music will only consist of one class, and will not be further divided into different classes for different genres.

1.6 Overview of the paper

The paper starts with a background chapter including an introduction of the area of Music Information Retrieval and an overview of earlier work done in the speech/music discrimination area. The chapter also presents features that are commonly used for these kinds of systems. Chapter 3 contains all the tests done within this work. It also presents the test results and ends with an evaluation of the results. Chapter 4 gives a detailed description of the finished algorithm and the paper then ends with a chapter including final conclusions drawn from the project.

2 Background

This chapter will give an introduction to the part of the Music Information Retrieval research area, which has been used during this project. It will also give a brief look at earlier work.

2.1 System structure

Earlier systems developed for SMD tasks have different structures. They can be divided into two main groups, online systems where the discrimination is made in real-time and offline systems where the discrimination is made on audio files. Both groups have their advantages; online systems are better suited for live purposes while offline systems can be made more accurate and faster. Different tasks require different systems, suited for specific purposes.

2.1.1 Online systems

In online systems, both the segmentation and the classification tasks need to be done at the same time. They can even be regarded as one task, where the output of the classifier is used for the segmentation. This is all done in real-time. ”Real-time” means that the system outputs results continuously as the input audio stream comes in, but with a delay of at least one analysis frame.

Online systems often concentrate on finding large changes in the audio to be able to find borders of speech or music segments. This is done by Saunders in [9] by dividing the audio into non-overlapping 16 ms long frames (256 samples at 16 KHz) from which simple features are extracted and using a longer 2,4 second (150 x 16 ms frames) analyze frame for statistical features used in the classification (see next chapter for more on features). Here a multivariate-Gaussian classifier does the classification.

Another real-time system is described in [7]. A typical frame-size of earlier online systems is around 20 ms and the analysis frame varies from half a second up to three seconds. The use of statistical features demands longer analysis frames to get good results. The short frame is used to extract features and the longer analysis frame is used to extract statistics of these features, such as variance and mean values.

2.1.2 Offline systems

The structure of offline systems varies considerably. The most common system has three steps plus pre- and post-processing, as seen in Figure 1. First the audio is divided into frames. From each of the frames a set of features is extracted and stored in a feature vector. In the second step the system segments the sound and in the third step the segments are classified by using some kind of classification method to decide whether a segment consists of speech or music. The SMD task is often done in more steps in an offline system than in an online system. The segmentation can be refined as in [8] by using the fact that neighboring segments have high probabilities of containing the same class.

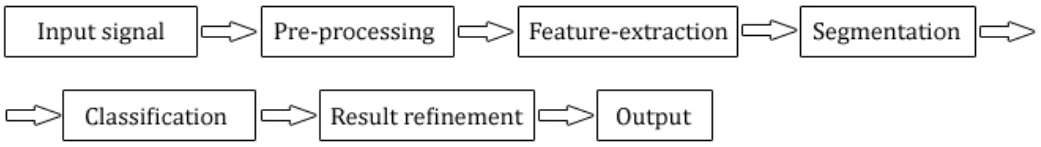

Figure 1. Block diagram of an offline system structure

In [8] some classification is done during the segmentation, and the more difficult segments are left for another more complex classification method. This saves computation time, since a simpler classifier makes the first classification. Other systems classifies large segments, and then divide the segments where a border is found into smaller segments, which also is a way to save computation time.

Offline systems often use a frame length of around 20 ms and an analysis window length around 1 second. Shorter frames make the features more sensitive to noise, while longer frames might include too many phonemes.

2.2 Features and feature extraction

Sound has many features, some that our ears can pick up and some that we cannot even hear. Speech has been closely studied and is relatively well defined, whilst music is a much wider class of sound. The French composer Edgar Varèse defined it as “Music is organized sound”. This might seem abstract, but can be used even in these kinds of technical solutions. It is a common approach to look for repetition in sound to classify it as music.

2.2.1 Speech and music

To be able to distinguish between speech and music, features that differ between the two classes need to be used. A simple look at the waveform of 1 minute excerpts of speech, pop music, classical music and opera (all examples taken from Swedish radio SR broadcasts) already indicates large differences between the classes. The speech waveform in Figure 2 shows rapid changes in energy and amplitude that none of the music waveforms does. The heavily compressed waveform of The Killers’ song in Figure 3 seems to totally lack dynamics, while the classical piece in Figure 4 and the opera piece in Figure 5 has some short amplitude peaks and shows large dynamic variations.

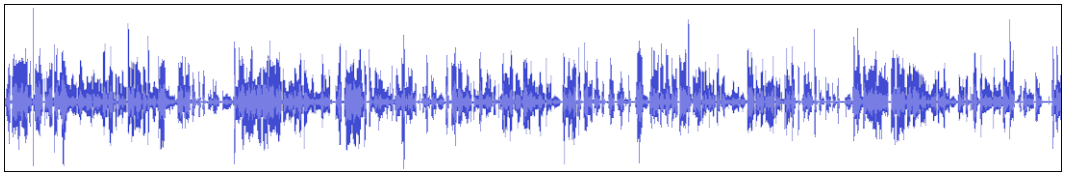

Figure 2. 1 minute of speech.

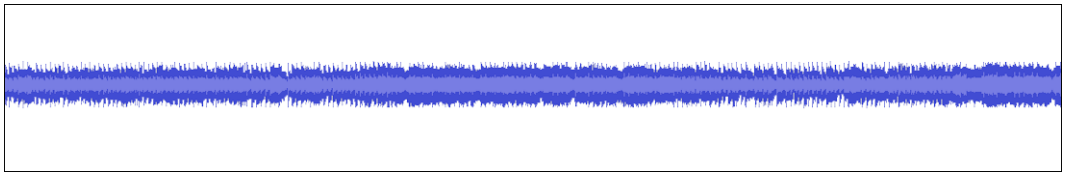

Figure 3. 1 minute excerpt of The Killers – All These Things.

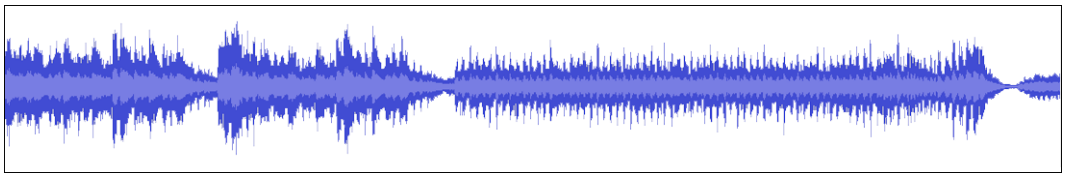

Figure 4. 1 minute of classical music.

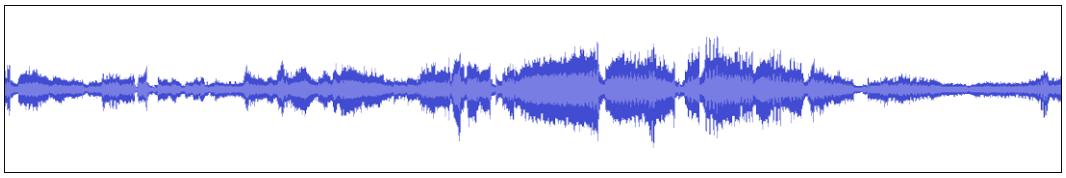

Figure 5. 1 minute of opera.

Even if the classes are easy to identify in the waveform, the exact position of the transitions can be difficult to detect. Figure 6 shows an excerpt from P3 Pop where a pop song abruptly stops and the host of the show starts speaking. The transition is marked with a vertical line.

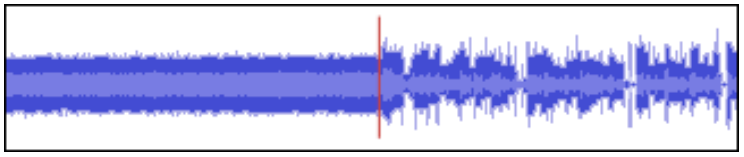

Figure 6. Waveform representation of excerpt of P3 Pop with transition from music to speech marked.

The changes in energy are even clearer when the sound wave is plotted in Figure 6. The scales aren't the same in the four examples, but the most interesting thing is the changes. In all the examples containing music the RMS never goes down to zero and doesn't diverge largely from the mean RMS. However, in the speech example there are relatively many frames containing zero or close to zero RMS, and the changes are rapid. What looks like large variations in RMS values of the pop music in Figure 7 can be explained by the scale on the Y-axis, ranging from 0 to 0.1, while speech ranges from 0 to 0.25.

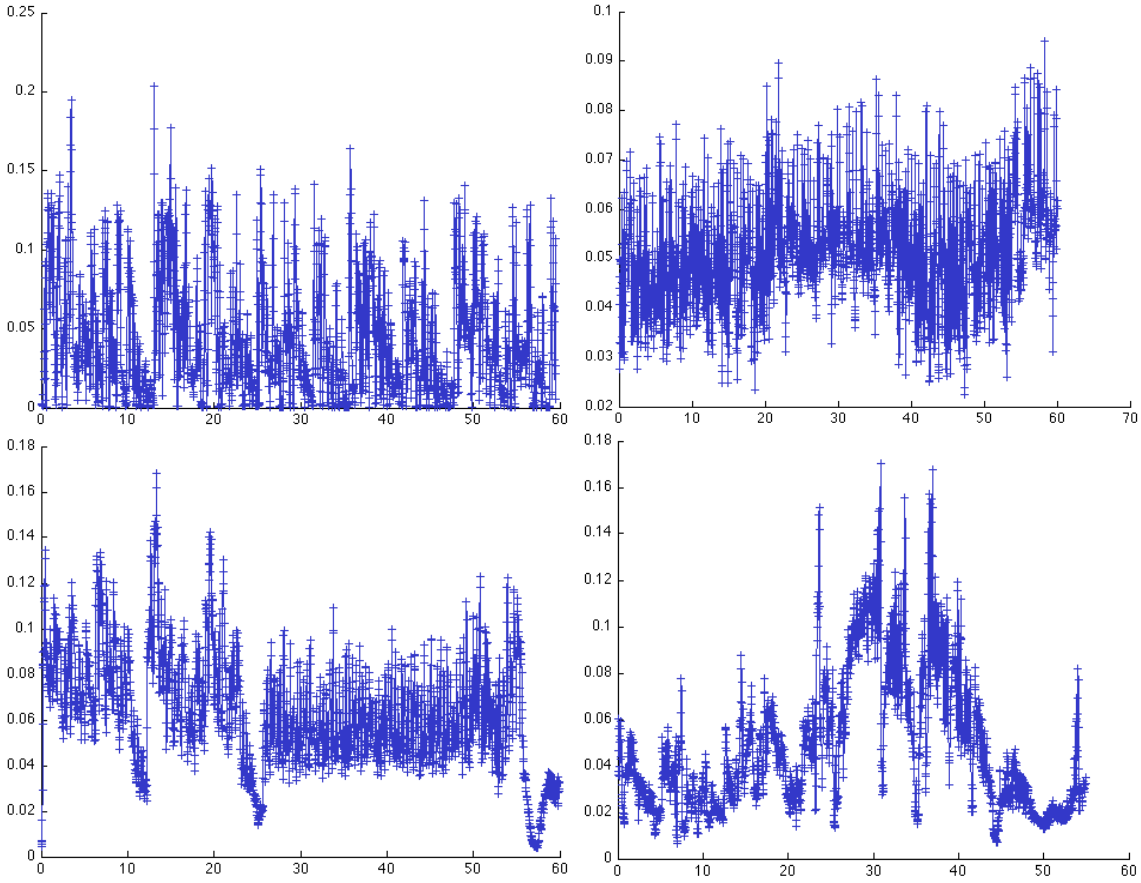

Figure 7. RMS graphs. Y-axis shows the RMS value calculated from 20 ms frames and X-axis shows temporal location in s. Top left: Speech. Top right: Pop music. Bottom left: Classical piece. Bottom right: Opera piece.

By looking at the spectrum of the four examples in Figure 8 it is clear that all the music examples have a higher peak in the low frequencies, although the peak occurs at different frequencies. This peak most likely corresponds to the fundamental frequency of the vocal components. The classical piece, which lacks vocals, does not have the same sharp peak as the other three examples. The speech example has more energy in the frequency range around 10 000 Hz than the music examples.

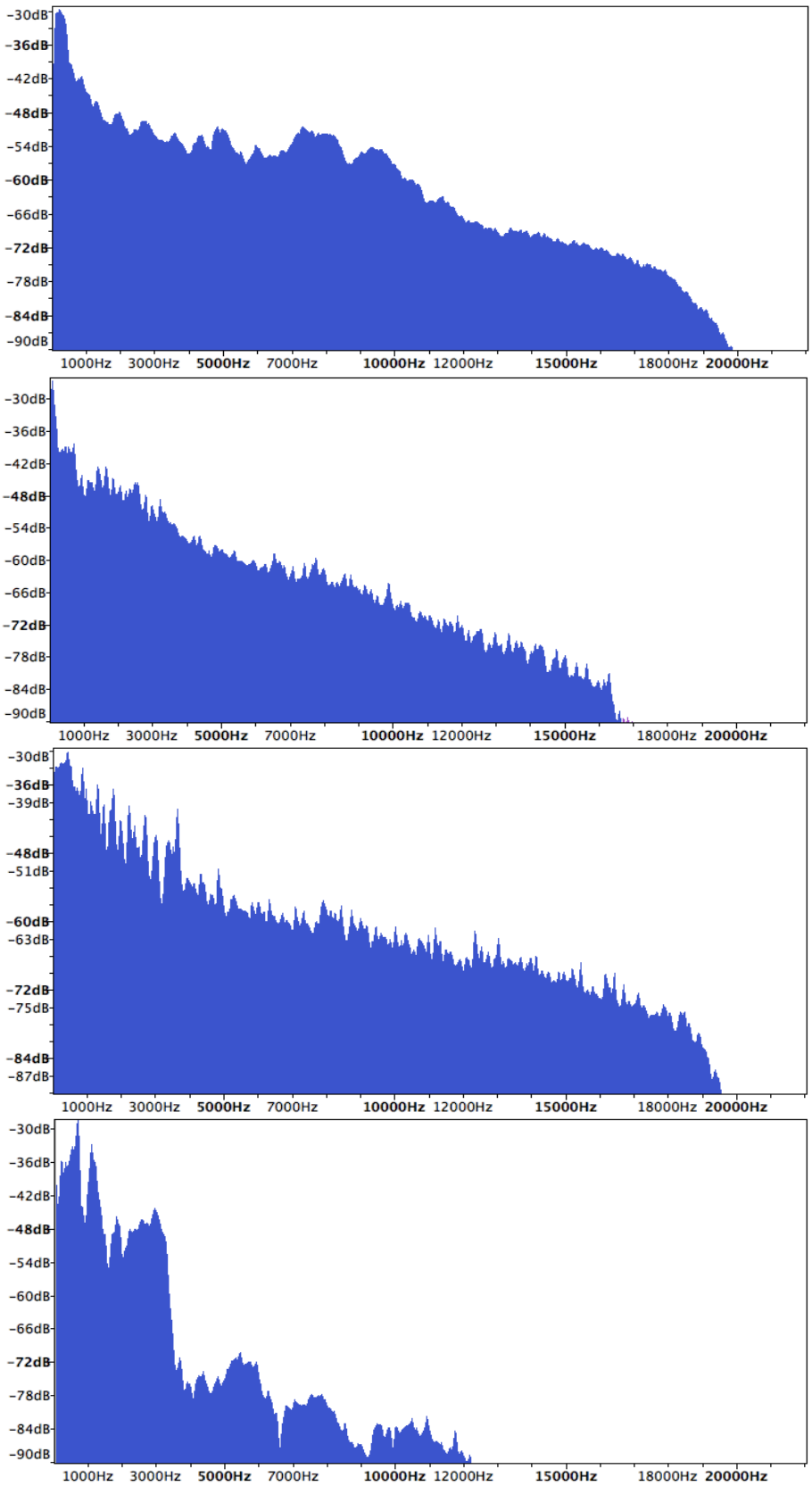

Figure 8. Spectrum plots analysed using a Hanning windows with a window size of 1024 samples. Y-axis shows sound level (dB) and X-axis shows frequency (Hz). From the top: speech, pop music, classical piece, opera piece.

The features described below are divided into three groups according to [5]: The simpler Standard low-level (SLL) features, the Frequency Cepstrum Coefficients and the more advanced Psychoacoustic features.

Online systems often use simpler features to avoid having to compute the FFT transform, which is a relatively costly computation compared to the SLL features. The most commonly used features in these systems are the zero crossing rate and RMS.

2.2.2 Standard Low-Level (SLL) features

These include RMS, Zero Crossing Rate, Spectral Centroid, Spectral Rolloff, Band Energy Ratio, Flux (also called Delta Spectrum Magnitude), Bandwidth, pitch and pitch strength.

Some features like bandwidth and pitch are self-explanatory. The other tested features are explained in the following sections.

2.2.2.1 RMS

RMS or Root Mean Square is a measure of amplitude of a sound wave in one analysis window. This is defined as

$$RMS = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n}} \quad (1)$$

where n is the number of samples within an analysis window and x is the value of the sample.

2.2.2.2 Zero Crossing Rate

This is a measure that counts the number of times the amplitude of the signal changes sign, i.e. crossing the x-axis, within one analysis window. The feature is defined as

$$ZCR = \frac{1}{T-1}\sum_{t=1}^{T-1} func\{s_t s_{t-1} < 0\} \quad (2)$$

where s is the sound signal of length T measured in time and $func\ \{ A \}$ equals 1 if A is true and 0 otherwise.

The Zero Crossing Rate feature is sometimes used as a primitive pitch detection for mono signals. It also is a rough estimate of the spectral content.

2.2.2.3 Spectral Centroid

This feature is effective in describing the spectral shape of the audio. The feature is correlated with the psychoacoustic features sharpness and brightness. There are several definitions of the Spectral Centroid feature in previous work. In this study it is calculated as a weighted mean of the frequencies in the FFT transform of the signal as

$$SC = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n=0}^{N-1} x(n)} \quad (3)$$

where *x(n)* represents the magnitude of bin number *n*, and *f(n)* represents the center frequency of that bin.

2.2.2.4 Spectral Rolloff

As the Spectral Centroid, the Spectral Rolloff is also a representation of the spectral shape of a sound, and they are strongly correlated. It's defined as the frequency where 85% of the energy in the spectrum is below that frequency. If K is the bin that fulfils

$$\sum_{n=0}^{K} x(n) = 0.85 \sum_{n=0}^{N-1} x(n) \quad (4)$$

then the Spectral Rolloff frequency is *f(*K*)*, where *x(n)* represents the magnitude of bin number *n*, and *f(n)* represents the center frequency of that bin.

2.2.2.5 Flux (Delta Spectrum Magnitude)

The Flux, or Delta Spectrum Magnitude, feature is a measure of the rate at which the spectral shape changes, or fluctuates. It is calculated by summing the squared differences of magnitude spectra of two neighboring frames. This feature has shown good results in the SMD task in [14].

$$F = \sum_{k=1}^{N/2} (|X_r[k]| - |X_{r-1}[k]|)^2 \quad (5)$$

where *N* is the number of FFT points and $X_r[k]$ is the STFT of frame *r* at bin *k*.

2.2.3 Frequency Cepstrum Coefficients

The second group is the Frequency Cepstrum Coefficients (FCC), which includes the Mel Frequency Cepstrum Coefficients (MFCC) and the Logarithmic Frequency Cepstrum Coefficients (LFCC). These are all power spectrum representation features calculated with different frequency scales.

The most frequently used coefficients for these systems are the MFCC. These are computed by taking the FFT of every analysis window, mapping the spectrum to the Mel scale, taking the base 10 logarithms of the powers and then applying a Discrete Cosine Function (DCT) to decorrelate the coefficients. [14]

The overall performance of MFCC features was shown in [5] to be slightly better than the SLL-features. This relates to the fact that MFCC performs better at pop and rock music, but somewhat worse at classical music that contains very little vocal information.

2.2.4 Psychoacoustic features

These features are more closely based on our perception of sound, and are therefore called psychoacoustic.

Loudness is the sensation of signal strength, and is primarily a subjective measure for us to rank sounds from weak to strong. Loudness can be calculated (Calculated Loudness) and is then measured in Sone. One Sone is defined as the loudness of a pure 1000 Hz tone at 40 dB re 20 µPa [21].

Roughness is described in [5] as "*the perception of temporal envelope modulations in the range of about 20-150 Hz, maximal at 70 Hz*" and is also said to be a primary component of musical dissonance.

Sharpness is a measure of the high-frequency energy related to the low-frequency energy strength. Sounds with lots of energy in the higher frequencies, and low energy levels in the lower frequencies are considered sharp.

2.2.5 Special features

In [1] the authors use a feature called Chromatic Entropy which is a version of Spectral Entropy. The spectrum is first mapped to the Mel scale and then divided into twelve sub-bands with center frequencies that coincide with the frequencies of the chromatic scale. The energy in each sub-band is then normalized by the total energy of all the sub-bands. Lastly the entropy of the normalized spectral energy is calculated as

$$E = -\sum_{i=0}^{L-1} n_i \times \log_2(n_i) \qquad (6)$$

where n_i is the normalized energy of sub-band i and L is the number of sub-bands.

The feature Modified Low Energy Ratio (MLER) is introduced in [8]. The feature exploits the fact that music shows little variation in energy contour of the waveform, whilst speech shows large variations between voicing and frication. MLER is defined as the proportion of frames with RMS power less than a variable threshold within one second. It is suggested by the authors that the threshold should be in the interval [0.05%, 0.12%] for best performance.

In [4] a feature called Warped LPC-based Spectral Centroid (WLPC-SC) is introduced. The frequency analysis is mapped to the Bark scale and then the centroid frequency is computed by a one-pole lpc-filter. This feature exploits the fact that speech has a low centroid frequency that varies with voiced and unvoiced speech, whilst music has a changing behavior.

2.2.6 Psychoacoustic pitch scales

Psychoacoustic scales are commonly used in Music Information Retrieval (MIR) systems. Speech and music is often well adjusted to our ears and therefore have most information in the frequencies where our ears have the best resolution. The most used scale is Mel, but even Bark and Equivalent

Rectangular Bandwidth (ERB) are sometimes used. The one used in this paper, Mel frequency, is defined as

$$f = 1127.01048 \times \log\left(\frac{f_1}{700} + 1\right) \quad (7)$$

where *f1* is the original frequency [1].

2.2.7 Extracting features

Usually a chosen set of features is extracted from each frame of the audio. The features are then often normalized by the computed mean value and the standard variation over a larger time unit and then stored in a feature vector.

Features are used in two ways, either by using the extracted value or by using changes over time. When using the changes over time it is possible to calculate statistical features like variance and standard deviation. In [5] it is shown how using changes over time are more accurate than only using the absolute values of the features.

Only one feature is used in [1], [4] and [7], although they all use advanced special features described earlier. Others have chosen to have a set of standard features. In [3] five different features are used, energy, ZCR, Spectral Entropy and the two first MFCCs. RMS and ZCR are used in [2].

2.3 Segmentation

In [1] and [4] a region growing technique is used for the segmentation step. This technique is widely used in image segmentation, but can also be used for audio. A number of frames are selected as seeds. The feature vectors of the seeds are then compared to the frames next to it. The segment then grows with the neighboring frames as long as the difference in the features does not exceed a predefined threshold.

Other systems like in [2] look for big changes between two neighboring 1 second frames. The two neighboring frames feature vectors are compared, and if they are sufficiently different, a segment border is detected. When a border is detected in a frame the transaction is marked within the frame with an accuracy of 20 ms.

2.4 Classification methods

When the segmentation process is done each segment should be classified either as speech or music. In a more complex system as in [3] more classes can be defined, such as silence or speech over music. The latter is often classed as speech in systems with only the two basic classes.

The extracted feature vector is used to classify each segment. A mean vector is calculated for the whole segment and is then compared either to results from training data or to predefined thresholds.

A method where the classification is based on the output of many frames together is proposed in [10]. Each second consists of 50 frames, and each frame is assigned a class by a quadratic Gaussian classifier. Then, a global decision is made based on the most frequently appearing class within that second.

2.4.1 Hidden Markov Models

HMMs are commonly used for classification. In [3] they are used together with a Bayesian Network Classifier. The feature vector sequence is used as input to the model. The model has a state for each class. Only two classes are used in [3], one for speech and one for music. Probability of transfers between states are computed on learning data and stored in the model.

In [6] a HMM with 24 states is used for speech, and another model is used for music. The use of three or four states could correspond to some phoneme classes, instead of having only one class for speech.

Different methods are used to train classification models. The Viterbi algorithm is used for HMM training in [11] and the Baum-Welch algorithm is another alternative for the system's learning process.

2.4.2 Refined classification

A method to refine the results of the classification method is described in [8]. Four states are used, one for speech, one for music, one for transitions to music and the last for transitions to speech. If the classifier outputs a music segment whilst the system is in the speech state, the segment will be stored in a stack. If the classifier keeps outputting music segments for a set time the state will change to music, and all the segments on the stack will be classed as music. But if the classifier outputs a speech segment within that time, the system will go back to the speech state and all the segments on the stack will be classified as speech. The accuracy of the classification is reported to increase by 6.5% percent when this method is used.

Refinement can be done in both online- and offline systems, but can be made more efficient in the latter where no demands on real-time output are present. The refinement technique described above needs a set number of segments to perform well. When these segments are fewer the performance will decrease drastically.

2.5 Evaluation methods

The results are often evaluated as in [3] with the measures recall, precision and the overall accuracy. In [3] recall is defined as the proportion of the frames with a specific class that were correctly classified and precision is defined as the proportion of the frames classified as a specific class, that actually belonged to that class. A total accuracy is then calculated as the total percentage of correctly classified data.

Systems can be optimized for either music or speech to raise the precision of that specific class in the system, although this often decreases the accuracy.

2.6 Earlier results

Comparing earlier systems for SMD tasks is not easy. There is no standard database for evaluating them, like there is for speaker and speech recognition systems. This makes it hard to actually rank the existing systems. Most articles report systems with accuracies above 90% and in [9] an accuracy as high as 98% is reported.

3 Evaluation and tests

This chapter presents the tests done within this work. The results of the tests are evaluated in order to be able to find the best method for the specific task.

The test phase was done in three steps: Feature and classification tests, segmentation tests and tests of the complete algorithm where both classification and segmentation were evaluated. The algorithm tests are presented at the end of chapter 4.

3.1 Test database

A test database was needed to run all the tests. Unfortunately, there is no standardized test database for SMD tasks, which makes the test results harder to compare. However, in this specific case the test database was constructed of material from Swedish Radio broadcasts to match the actual material that will be used as input when the algorithm is implemented in a production environment.

Material was selected from Swedish Radio's digital archive Digas to cover all kinds of genres. The material is made up of whole programs that were aired on radio and was selected in order to get a wide spread of included material.

Three sets of test audio were selected and extracted form the material. The first set consisted of 30 seconds long audio files containing only speech or only music. The speech examples included female and male voices, interviews, phone interviews, sport commentary and more. The music examples ranged over many genres from different programs. These were then used for the feature tests to be able to value each feature and calculate the correlation between the features. The second set consisted of 1 minute long audio files containing both speech and music and with at least one transition between classes. The files were selected so that the transition can be anywhere within the file. These were then used for the segmentation tests. The third and last set consisted of whole programs containing both music and speech segments. The length of these files varied from a couple of minutes up to 90 minutes. These were then used to test the complete algorithm.

3.2 Tools

Audacity [15] was used to edit and convert audio files to construct the test database.

Sonic Visualiser [16], together with various Vamp plugins [17], has been used for early analysis of the test database. Sonic Visualiser is developed by the Centre for Digital Music, Queen Mary University of London [18] and is easy to use to get a quick look at how values of features changes in audio.

MATLAB has been used for testing and evaluating during the whole process. Many of the features have been extracted using the MIR Toolbox [19],

developed by the University of Jyväskylä [20] in Finland, and the rest of the features were extracted using custom written code.

The final algorithm is written in C code because of its effectiveness and speed, using the libsndfile library [13] for reading wave files.

3.3 Feature tests

The first test was to check if the selected features have any significance for the classification task. The tested features were chosen from earlier algorithms that performed the SMD task with good results and a few were added based on personal hypothesizes.

Some initial tests were also done with other features like flux, other rhythm features and different uses of MFCC, however since these features did not show any potential for the SMD task they were not included in these tests.

In these tests the features were tested for the classification purpose and the correlation between features were measured. The chosen features were extracted from the test material in the first set, containing only one class to get a distribution of values for each class. Most features were tested in five different ways, the extracted value, the variance, the standard deviation, the derivative and the standard deviation of the derivative. Both the standard deviation and the variance derive from the same data since the variance is the square of the standard deviation. Because of this, only the standard deviation results are presented. Histograms of the results were then generated to visualize the distribution. Features that showed interesting results are further discussed later in this chapter.

The tested features were RMS amplitude, Zero Crossing-Rate (ZCR), Mel Frequency Cepstrum Coefficients (MFCC), Spectral Centroid (SC), Pulse Clarity (PC) and Modified Low Energy Ratio (MLER). Their abbreviations will be used during the rest of this chapter, together with an abbreviation for the way the feature is used using the following naming conventions.

SD	Standard Deviation
D	Derivative
SDD	Standard Deviation of Derivative

This means that the standard deviation of the Zero Crossing-Rate will be abbreviated ZCR|SD. A total of 29 different feature variations were tested in these tests.

3.3.1 Feature test results

Feature	Frame length	Accuracy
RMS	20 ms	0.639
RMS|SD	1 s	0.829
RMS|D	1 s	0.548
RMS|SDD	1 s	0.764
ZCR	20 ms	0.588
ZCR|SD	1 s	0.837
ZCR|D	1 s	0.550
ZCR|SDD	1 s	0.835
SC	1 s	0.581
SC|SD	30 s	0.792
SC|D	30 s	0.589
SC|SDD	30 s	0.958
MFCC1	20 ms	0.517
MFCC1|SD	1 s	0.548
MFCC1|D	1 s	0.525
MFCC1|SDD	1 s	0.553
MFCC2	20 ms	0.521
MFCC2|SD	1 s	0.548
MFCC2|D	1 s	0.525
MFCC2|SDD	1 s	0.553
MFCC3	20 ms	0.521
MFCC3|SD	1 s	0.548
MFCC3|D	1 s	0.525
MFCC3|SDD	1 s	0.553
MFCC4	20 ms	0.519
MFCC4|SD	1 s	0.548
MFCC4|D	1 s	0.525
MFCC4|SDD	1 s	0.553
MLER	1 s	0.969
PC	5 s	0.807

Table 1. Feature test results. Frame length is a time measure and accuracy is shown in percentage/100.

Results of the feature tests are shown in Table 1 above. The accuracy of the each feature is a measure of how far the speech and the music distribution are from each other. This was calculated by finding the threshold value with the least misclassifications by testing all threshold values in an interval specified for each feature. The numbers of misclassified frames for the best threshold were then counted and divided by the total number of frames and then the result was subtracted from 1.

$$A = 1 - \frac{\min(misclass)}{nFrames} \quad (8)$$

Features with accuracy results close to 50% can be considered as random, containing no useful information for the classification. The threshold

optimization described above is the reason that all features performed over 50%.

The five features with highest accuracies was the Modified Low Energy Ratio (97%), the standard deviation of the derivative of the Spectral Centroid (96%), the standard deviation of the Zero Crossing-Rate (84%), the standard deviation of the Root Mean Square (83%) and the Pulse Clarity (81%). Surprisingly, none of the MFCC features showed any useful results in these test and got the worst results of all features. The extracted values for each MFCC-parameters gave just above 50% accuracies with the optimal threshold. Since small variations can depend on the rather small test database, the MFCC-features themselves can be regarded as useless for these kinds of classification tasks. The reason that it got over 50% is also because the best threshold value is sought, and therefore gives a higher accuracy. However, MFCC features can still be used in other ways discussed later in this paper.

The correlation between the four top features cannot be calculated directly since they use different length of analysis frames and therefore generate different number of data points. The MLER feature uses the pauses in speech to discriminate between the classes and so does the ZCR, so they will show high correlation between them. Unfortunately all four features have problem with the same kind of audio, namely speech with background noise of some kind, which are often classified as music. The correlation between the top four features has been calculated by comparing the mean values for each feature for every file in the first set in the test database.

	MLER	**SC|SDD**	**ZCR|SD**	**RMS|SD**	**PC**
MLER	-				
SC|SDD	0.93	-			
ZCR|SD	0.97	0.97	-		
RMS|SD	0.97	0.93	0.97	-	
PC	0.87	0.77	0.77	0.83	-

Table 2. Cross-correlation of the top four features from the accuracy tests.

As seen in Table 2, all the top four features are strongly correlated. The Pulse Clarity feature shows the least correlation with the others while the standard deviation of the Zero Crossing Rate showed as high correlations as 97% with both the Modified Low Energy Ratio and the standard deviation of the derivative of the Spectral Centroid.

3.3.2 Analysis of Low-level features

The Low-level features are interesting because they incur relatively low computation costs. However, both RMS and ZCR showed weak results in the feature tests. RMS performed somewhat better which can be seen in the histograms below, where big differences between the red and blue gives good discrimination possibilities. Both RMS and ZCR where extracted for every frame. Speech typically contains many frames with RMS values close to zero, while music has almost no zero energy frames and a peak at about 0.4. The

close to zero energy frames in speech represents the pauses between syllables that always exists in speech recordings without background sound. The ZCR values are centered round the same mean value, but music has a higher peak, i.e. a lower standard deviation. This is also what gives the good results for ZCR|SD and ZCR|V.

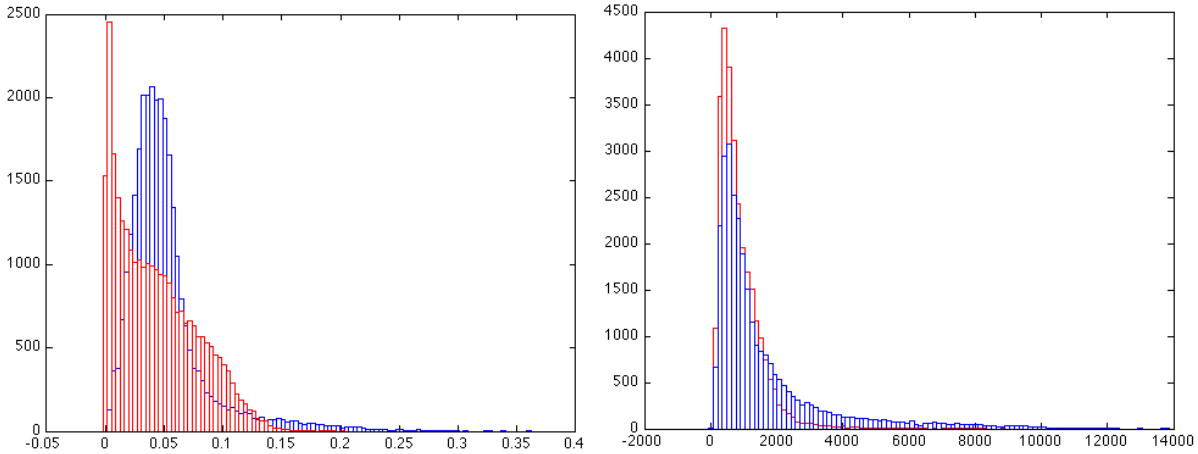

Figure 9. Histograms of feature values. Left: RMS. Right: ZC. Blue shows music and red shows speech. Y-axis is the number of occurrences of the bins centered around the X-values.

3.3.3 Analysis of MFCC

The MFCC features have been frequently used in MIR algorithms. However, as seen in the figure below all the MFCC values are centered round the same values, but with some differences in the standard deviations. As seen in the test results these variations of the features also scored higher than the extracted values, but still with very low top scores.

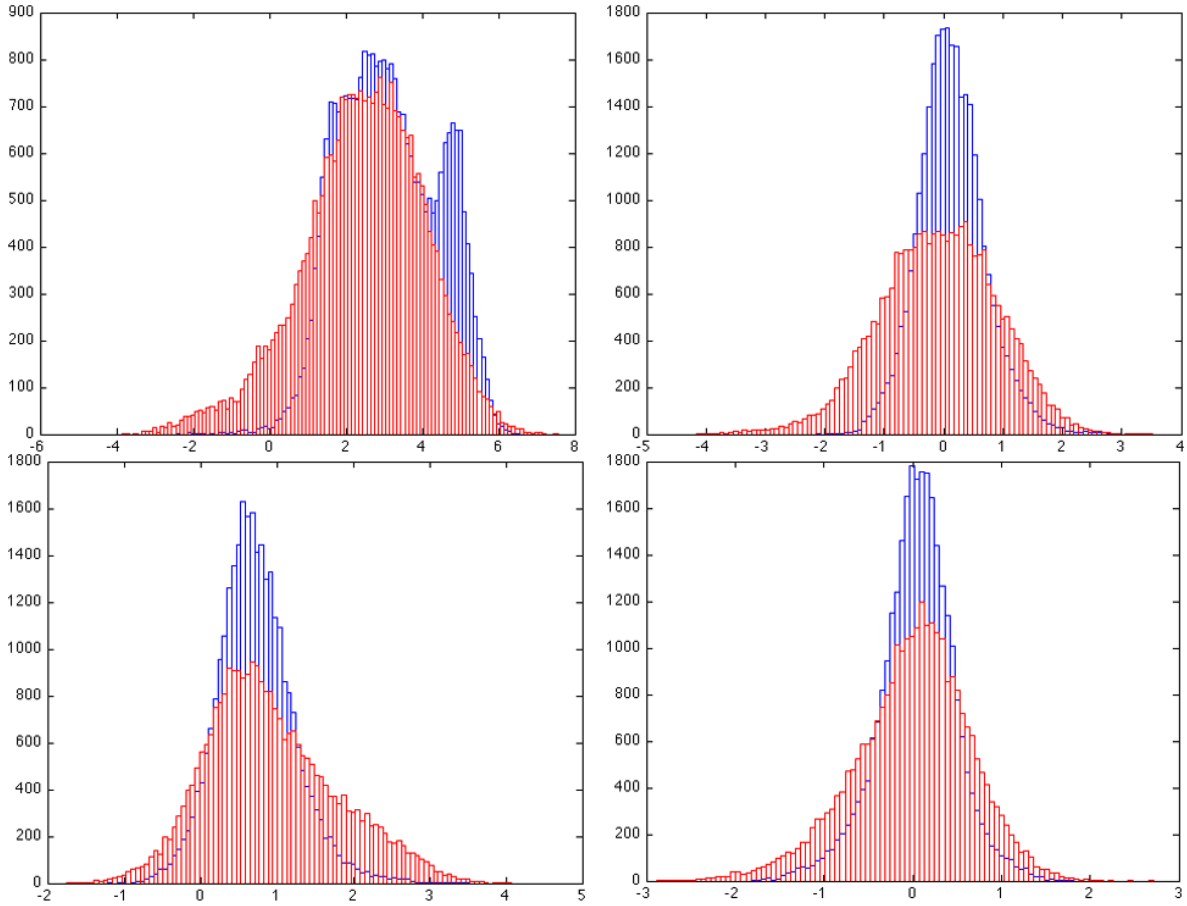

Figure 10. Histogram of feature values. Top left: MFCC1. Top right: MFCC2. Bottom left: MFCC3. Bottom right: MFCC4. Blue shows music and red shows speech. Y-axis is the number of occurrences of the bins centered round the X-values.

3.3.4 Analysis of Modified Low Energy Ratio

As seen in Table 1, the absolute values of the features are often not a good way to discriminate between the two classes. A better way might be to use the feature changes over time as done by the MLER feature.

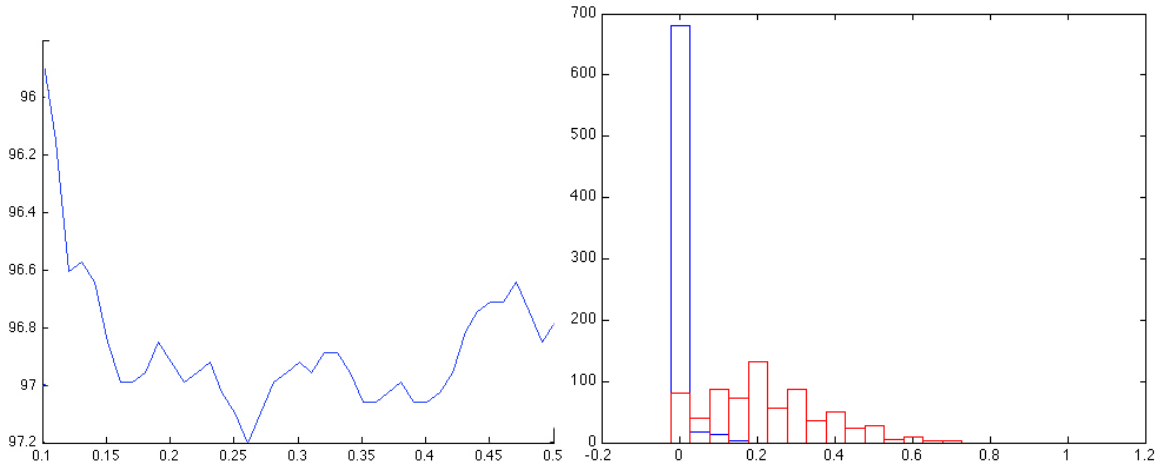

Figure 11. MLER. Left: Threshold optimization, Y-axis shows correctly classified frame percentage and X-axis shows the threshold. Right: With threshold 0.26. Blue shows music and red show speech. Y-axis is the number of occurrences of the bins centered round the X-values.

Figure 11 shows the Modified Low Energy Ratio as described earlier in the report. When using a threshold for low energy at 0.26 of the mean RMS value, the classification achieved 97% accuracy. If the MLER equals zero the frame is classified as music, otherwise it is classified as speech. Almost all the speech frames with zero MLER come from sport commentary with an audience in the background. The left graph shows the optimization of the total error-rate. In some applications it might be better to optimize it for either speech or music.

3.4 Segmentation tests

The segmentation tests were done on the audio files containing at least one transition between classes. The exact positions of the transitions within the files were noted as a ground truth for the tests. These positions are not absolute and a small test showed that different people would place these positions differently. Sometimes a transition can be up to half a second long segment of silence, and it would be acceptable for the segmentation to place the transition anywhere in this silence, since silence was not defined as either speech or music. For these two reasons, a deviation of 100 ms was allowed and counted as a clean hit. A deviance of 0.1 - 1 second counted as a hit, and the distance from the border of the clean hit segment was calculated. If the segmentation marked a transition more than 1 second away from the ground truth, this counted as a miss.

Three measures were then used to evaluate the segmentation techniques:

Hit efficiency: A measure of how many hits were found. The misses is subtracted from the hits, and the difference is then divided by the total number of transitions.

$$Hit_{eff} = \frac{Hits - Misses}{Transitions} \quad (9)$$

Hit accuracy: An accuracy measure where only the hits are considered. All the distances are added and a mean value was calculated by dividing by the total number of hits. A clean hit counts as zero distance.

$$Hit_{acc} = \frac{\sum_{n=1}^{N} \left| Hitpos_n - Transpos_n \right|}{N} \quad (10)$$

Where *N* is the numbers of hits.

Hit rate: A simple measure where only the hits are considered once again. The total number of hits is divided by the total number of transitions in the test files.

$$Hit_{rate} = \frac{Hits}{Transitions} \quad (11)$$

None of these measures considers the computation times: These are discussed in the test evaluations below.

In the first part of the test, the region growing technique was tested against the neighboring difference technique, both described in 2.3. Both techniques were tested with the same features.

Technique	Hit efficiency	Hit accuracy	Hit rate
Region growing	81%	79 ms	100%
Neighboring difference	94%	44 ms	99%

Table 3. Segmentation technique test results.

As seen in Table 3, the neighboring difference technique achieved better results at both the Hit efficiency and the Hit accuracy. The Hit rate measure is harder to evaluate, but results as close to 100% as possible are good. The neighboring difference technique did not detect 1% of the hits, but as can be seen in the Hit efficiency measure, did not detect as many misses either. A choice was made to pursue only the Neighboring difference technique because of its efficiency and accuracy.

Feature	Hit efficiency	Hit accuracy	Hit rate
MLER	95%	19 ms	99%
SC|SDD	93%	70 ms	99%
ZCR|SD	88%	24 ms	99%
RMS|SD	87%	27 ms	99%
PC	88%	72 ms	99%

Table 4. Segmentation feature test results, showing the top 5 features.

In the second test all features tested in the feature tests were tested for the neighboring difference technique. The same kind of features performed well also in these tests, although features with low analysis windows and short frames performed better and showed better Hit accuracy results. Table 4 shows that the top 4 features all detected 99% of the transitions, but the MLER feature achieved the best results both for the Hit efficiency and the Hit accuracy measures. Further tests were then done with the MLER feature, and both the efficiency and the accuracy improved when using shorter frame lengths and instead used a combination of the mean RMS and the variance of RMS. A Hit efficiency of 96% and a Hit accuracy of 17 ms were then achieved.

3.5 Design criteria

There are many aspects to consider when choosing features to use in the algorithm. The test results need to be thoroughly analyzed regarding these aspects:

- Accuracy of the feature to be able to discriminate between the two classes, needed for both classification and segmentation. Ideal features have similar values within one class and a clear difference from other classes.

- Computational costs. Costly operations should be avoided to make the discrimination task efficient and cheap. Even offline systems need to be fast to improve the cost efficiency.

- Correlation between chosen features. The use of two different features needs to be motivated by improved results. If two features are highly correlated, the improvements in accuracy will be relatively costly.

- Insensitivity to noise in the input signal.

The MLER feature was chosen for its excellent accuracy and because of its combination with the neighboring difference segmentation technique. Both the segmentation and the classification are based on only one single low level feature, the RMS amplitude. This makes the computational costs very low. Since the high performing features were all highly correlated it was not motivated to add another feature. The improvements in accuracy will be costly, and another classification method will be needed. When the RMS amplitude is normalized over the file maximum it is also insensitive to sound levels, but the background noise will still be a problem in the classification

The three calculated measures from the segmentation tests where considered when choosing the segmentation method. The Hit rate was considered the most important, since misses (transition detections where no transition is within one second) can be discarded in an algorithm using segmentation refinement methods. The Hit efficiency was only considered when two tests showed common results in the Hit rate. The computation times too, were considered when choosing the final method for the algorithm.

The neighboring difference technique showed better results for all features and was therefore the strongest candidate. It is also a good match when the MLER is used for the classification task since it achieved good results in Hit accuracy when using RMS. Computation times are largely reduced since both the classification and the segmentation will use the same extracted features.

4 Algorithm

This chapter contains a detailed description of the final algorithm that was coded and delivered to Swedish Radio.

The algorithm is written in C code and uses the libsndfile library [13] to read and write to wave files. The discrimination is done on audio files and hence this is an offline procedure.

4.1 Signal preprocessing

Before anything is done with the audio, a few preprocesses are performed on the audio signal.

There is no added information in the difference of two channels that can be used for the classification or the segmentation. Therefore it is desirable to have a mono signal to simplify later processes. The algorithm checks the number of channels of the audio. If the signal has more than one channel, it is mixed down to mono.

The amplitude of the signal is then normalized to the maximum amplitude of the whole file to remove any effects the overall amplitude level might have on the feature extraction.

4.2 Feature extraction

After the audio signal has gone through the preprocessing part, it is split into 21 ms non-overlapping frames. The RMS amplitude is then calculated for each frame using equation 1.

Once the RMS amplitude has been calculated, the frames are grouped together to form 1 second (48 short frames) analysis frames. These are also non-overlapping. Four features based on the RMS amplitude values are then extracted from each 1 second frame, mean RMS, variance of RMS, a locally normalized variance of the RMS and a Modified Low Energy Ratio (MLER). The normalized variance is the variance of RMS divided by the mean RMS.

The Modified Low Energy Ratio is the proportion of low energy short frames within the 1 second frame. The threshold for low energy depends on the mean RMS amplitude. The mean RMS amplitude in the analysis frame is multiplied with a predefined valued defined by the test results. Speech contains many small pauses between syllables and words and therefore has a higher MLER than music. This is used later to classify each segment.

4.3 Segmentation

The task for the segmentation part of the algorithm is to find the exact position of transitions between two classes. The segmentation is based, just as the classification, on the features extracted earlier. Every 1 second frame is

examined to look for candidates for transition frames, and then an exact position of the transition is found.

This is a modified version of the segmentation done in [2]. The advantage of this method is that it uses the same base feature, RMS amplitude, as the classification. This means that no further feature extraction is needed and this saves computation time and minimizes the reading of the audio files. The RMS amplitude is used in another way, since the MLER feature used for classification requires longer analysis frames, while the mean and variance of the RMS changes as quickly as do the audio classes.

The first step is done by looking at the frame before and the frame after the examined frame. If the two neighboring frames are different the examined frame is likely to have a transition. The transition can be anywhere within that second and the exact position is determined in the next step. A problem will occur if the class changes two times within the examined frame. Then the two neighboring frames will not differ enough to be chosen as a candidate for transition frames. Although, these kinds of errors are not corrected in the segmentation since segments smaller than 2 seconds will be removed later.

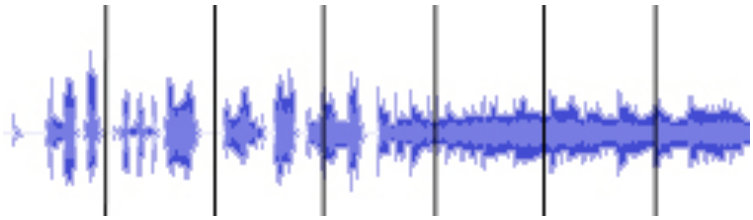

Figure 12. Waveform with transition from speech to music. Seconds are marked with vertical lines.

The comparison between neighboring frames are based on the mean and the variance of the RMS values. Since the distribution of the amplitude of the audio signals is a Laplacian distribution as shown in [2], a probability density function of the χ^2 distribution is used, defined as

$$p(x) = \frac{x^a e^{-bx}}{b^{a+1}\Gamma(a+1)} \qquad (12)$$

where $x \geq 0$, Γ is the gamma function and the two parameters, a and b, are defined as

$$a = \frac{\mu^2}{\sigma^2} - 1 \quad \text{and} \quad b = \frac{\sigma^2}{\mu} \qquad (13)$$

where μ is the mean RMS and σ^2 is the RMS variance. The similarity measure is based on the probability density function

$$p(p_1, p_2) = \int \sqrt{p_1(x) p_2(x)}\, dx \qquad (14)$$

where p_1 and p_2 refers to the probability functions of each frame. When the χ^2 distribution in equation (12) is inserted in equation (14) this gives a similarity measure calculated with

$$p(p_1,p_2)=\frac{\Gamma\left(\frac{a_1+a_2}{2}+1\right)}{\sqrt{\Gamma(a_1+1)\Gamma(a_2+1)}}\frac{2^{\frac{a_1+a_2}{2}+1}b_1^{\frac{a_2+1}{2}}b_2^{\frac{a_1+1}{2}}}{(b_1+b_2)^{\frac{a_1+a_2}{2}+1}} \quad (15)$$

Since the measure is calculated with two frames with one frame between, the examined frame *i*, the dissimilarity measure is defined as

$$Dissim(i)=1-p(p_{i-1},p_{i+1}) \quad (16)$$

This will give high probabilities of change even for the surrounding frames. As seen in Figure 12, out of the five seconds marked by the vertical lines second 2 and 4 will differ most in mean and variance of RMS. However, second 3 and 5 will also give high values in the dissimilarity measure. To dampen the effect of this error, a filter needs to be applied. The similarity value is therefore locally normalized over 5 seconds with the examined frame in the centre. The normalizing is calculated by

$$Dissim_{norm}(i)=\frac{Dissim(i)\times\left(D(i)-\frac{D(i-2)+\ldots+D(i+2)}{5}\right)}{\max(D(i-2),\ldots,D(i+2))} \quad (17)$$

The dissimilarity measure is multiplied by a positive difference of the mean of the neighborhood. If the difference is negative, it is set to zero. This is then divided by the maximum value of the neighborhood.

A threshold for the normalized dissimilarity value is then set according the results of the test material, to determine which frames are selected as candidates for transition. The threshold is variable and depends on the variance of the RMS in the neighboring frames.

When the candidate transition frames has been chosen, an exact position for the transition needs to be found. This is done in a way similar to the last step. For every 20 ms frame the previous and the next one seconds are compared. A value is given for every 20 ms frame for the probability of change, and the frame with the highest probability is marked as the exact position of the transition.

4.4 Classification

Only the MLER feature is used for the classification part. A predefined threshold is set and all segments that have a higher average MLER than the threshold are classed as speech, and everything below the threshold is classed as music. The use of only one feature reduces the computation time. The algorithm needs no training material to work, since the threshold is set according to the results of the test material.

4.5 Refinement

Simple refinements of the segments are done after the classification. If two consecutive segments are given the same class they are merged together and the transition between them is erased.

4.6 Output

When all the processes, segmentation, classification and refinements are done, the results are ready to be output. The algorithm then creates a simple text file with one line for each segment. The line contains the exact position of the start of the segment and a binary number showing the class of the segment. A line would look like this

```
1 - 0.000000
0 - 9.770833
1 - 27.098166
```

where the 0 stands for speech (1 for music) and 9.770833 is the exact position of the transition measured in seconds. The position of the transition is measured in seconds because of easier handling for other applications. If the position were marked with an exact frame, the third party application would also have to know the sample frequency of the audio. The transition frame is easily calculated by

$$F = t * sr \quad (18)$$

where t is the position of the transition measured in seconds and sr is the sample rate of the audio.

A Flash player that uses the output data to mark the segments can then read the text file and mark the segments in the navigation bar, as seen in Figure 13. This could be used as a guide when editing the audio, or simply to look for errors in the output.

Figure 13. Flash player with marked segments in the navigation bar. Green shows speech segments and white shows music segments.

The Broadcast Wave Format contains a marker chunk, which could be used to mark the transition points of the segments. This is not yet integrated in the application, but could be done for specific implementations.

4.7 Results

The algorithm tests were performed on the finished algorithm. Audio files containing full lengths programs from Swedish Radio were used as input and both the segmentation and classification were evaluated at the same time. The resulting accuracy is the percentage where the right class is found at the right time. The length of the correctly classed audio is divided by the total length of the program.

	Speech	Music
Speech	95,4%	4,6%
Music	1,9%	98,1%

Table 5. Algorithm results. Left column shows input and top row shows the output of the algorithm.

The left column of the table shows the class of the inputted audio, and the top row shows the class outputted by the algorithm. Speech reaches a lower accuracy because of the sport commentary segments that are often classed as music, while music is correctly classed as music as often as 98,1% of the time.

	Right	Wrong
Total	97,3%	2,7%

Table 6. Summarized results for all inputs to the algorithm.

This gives a total accuracy of 97,3% since the test material contained more music than speech.

The computation time of the algorithm varies depending on the format of the input and the number of input channels. The computation time did not exceed 1% of the length of the audio in any of the test files.

5 Conclusions

The final algorithm gives an accuracy of over 97% in the tests performed with material from Swedish Radio. This matches the results reported in earlier work, yet without any advanced features that require long computation times. 97% is also enough for most applications. In some applications the misclassified audio still needs to be considered. Since we know from the tests what kind of audio that gives low accuracies, this can be done by manually discriminating these files.

A graphical interface where the segments are marked could be of good use for manual work with the audio. The algorithm does the discrimination job, but the results might still need refinement. This is true for an application for automatic editing for pod material, where the results of the discrimination can be used as a guide, but some editing like cross-fades is still needed to get a good sounding result.

Offline systems benefit most from faster computation times. Real-time, online systems still have to use the same length for the analysis window to gather the needed statistics and can benefit only by being able to use more and more advanced features. Offline systems, on the other hand, can both use more advanced features to achieve higher accuracy and at the same time compute faster.

6 Future work

There are still endless features and feature combinations to be tested for this task. The features tested during this work are still quite simple. Even the modified special features like the one used in the algorithm are based only on one low level feature. Further tests could also be done with MFCC features, even though they showed poor results in the present tests. Combinations of different MFCC features have been tested in earlier work. Further exploring relations between different MFCC, such as distances between the second and third coefficient, could give good results.

More complex features like pitch curves extraction have been discussed and tentatively tested, but there is still much to explore in this area. Pitch features should be a good complement for the MLER feature since they work on different aspects of the sound. Also features based on rhythm could be a good alternative to complement the MLER feature.

Adding subclasses to both the music and the speech classes would be useful for many applications. Swedish Radio has already requested the possibility to be able to discriminate between male and female speakers. Music could be split into genre sub-classes for further discrimination. There is already some work done with genre classification, but more research is needed to create a useful application.

Constructing a low cost real-time discriminator that could be inserted in car radio receivers could be another kind of project. Listening in cars often involves noisy environments where speech needs to be amplified more than music to increase the audibility. Different listening environments demand different amplifications of speech. In such an application, the processing cannot be done in the transmitter and needs be processed in real-time.

7 Acknowledgements

This work has been done with much help from both Swedish Radio and the Royal Institute of Technology. Special thanks to:

My supervisor at Swedish Radio, Björn Carlsson, for all the help with technical questions about radio and also for all the encouragement during the work.

My supervisor at the Royal Institute of Technology, Anders Friberg, for all the feedback and ideas. Also for all the special knowledge in feature extraction and classification.

Hasse Wessman and Lars Jonsson at Swedish Radio, Technical Development for giving me the opportunity to do this project and giving me an inspiring working environment.

The rest of the staff at Swedish Radio, Technical Development for feedback and motivation.

My friend John Häggkvist, who has helped me out during the coding of the algorithm.

8 References

[1] Pikrakis, A., Giannakopoulos, T. & Theodoridis, S. "*A computationally efficient speech/music discriminator for radio recordings*", in University of Victoria, ISBN: 1-55058-349-2, pp. 107-110, 2006.

[2] Panagiotakis, C. & Tziritas, G. "*A Speech/Music Discriminator Based on RMS and Zero-Crossings*", IEEE Transactions on Multimedia, vol. 7(1), pp. 155-166, Feb. 2005.

[3] Pikrakis, A., Giannakopoulos, T. & Theodoridis, S. "*Speech/Music Discrimination for radio broadcasts using a hybrid HMM-Bayesian Network architecture*", in Proc. of the 14th European SignalProcessing Conference (EUSIPCO-06), September 4-8, 2006, Florence, Italy.

[4] Muñoz-Expósito, J. E., Garcia-Galán, S., Ruiz-Reyes, N., Vera-Candeas P. & Rivas-Peña, F. "*Speech/Music discrimination using a single warped LPC-based feature*", in Proceedings of the International Symposium on Music Information Retrieval, 2005.

[5] McKinney, M.F. & Breebart, J. "*Features for Audio and Music Classification*", in Proceedings of the International Symposium on Music Information Retrieval, 2003.

[6] Karnebäck, S. "*Speech/Music Discrimination Using Discrete Hidden Markov Models*", TMH-QPSR, KTH, Vol. 46, 41-59, 2004.

[7] Alnabadi, M. S. "*Real Time Speech Music Discrimination Using A Single Feature*", Durham University School of Engineering, 2007.

[8] Wang, W. Q., Gao, W., Ying, D. W. "*A Fast and Robust Speech/Music Discrimination Approach*", In Proceedings of the International Conference on Information, Communications and Signal Processing, 2003.

[9] Saunders, J. "*Real-time discrimination of broadcast speech/music"*, in Proc. IEEE Intern. Conf. on Acoustics, Speech and Signal Processing, 1996.

[10] El-Maleh, K., Klein, M., Petrucci, G. & Kabal, P. "*Speech/Music discrimination for multimedia applications*", ICASSPOO, 2000.

[11] Ajmera, J., McCowan, I. & Bourlard, H. "*Speech/Music segmentation using entropy and dynamism features in a HMM classification framework*", in Speech Communication 40, pp. 351-363, 2003.

[12] Swedish Radio webpage, http://www.sr.se/sida/default.aspx?ProgramId=2438. Retrieved 6/1 2010.

[13] Libsndfile webpage, http://www.mega-nerd.com/libsndfile/. Retrieved 6/1 2010.

[14] Burred, J. J., "*An Objective Approach to Content-Based Audio Signal Classification*", Technische Universität Berlin, 2003.

[15] Audacity webpage, http://audacity.sourceforge.net/. Retrieved 24/2 2010.

[16] Sonic Visualiser webpage, http://www.sonicvisualiser.org/. Retrieved 24/2 2010.

[17] Vamp plugins webpage, http://vamp-plugins.org/. Retrieved 24/2 2010.

[18] Centre for Digital Music webpage, http://www.elec.qmul.ac.uk/digitalmusic/. Retrieved 24/2 2010.
[19] MIRtoolbox webpage, https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox. Retrieved 24/2 2010.
[20] University of Jyväskylä webpage, https://www.jyu.fi/en/. Retrieved 24/2 2010.
[21] Leijon, A. "*Sound Perception: Introduction and Exercise Problems*". Royal Institute of Technology, Stockholm, 2007.