CONTENT-BASED AUDIO MANAGEMENT AND RETRIEVAL SYSTEM FOR NEWS BROADCASTS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EBRU DOĞAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

SEPTEMBER 2009

Approval of the thesis:

CONTENT-BASED AUDIO MANAGEMENT AND RETRIEVAL SYSTEM FOR NEWS BROADCASTS

submitted by **EBRU DOĞAN** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _______________

Prof. Dr. Müslim Bozyiğit
Head of Department, **Computer Engineering** _______________

Prof. Dr. Adnan Yazıcı
Supervisor, **Computer Engineering Department, METU** _______________

Asst. Prof. Dr. Mustafa Sert
Co-supervisor, **Computer Engineering Department, Başkent Univ.** _______________

Examining Committee Members:

Assoc. Prof. Dr. Ahmet Coşar
Computer Engineering, METU _______________

Prof. Dr. Adnan Yazıcı
Computer Engineering, METU _______________

Asst. Prof. Dr. Murat Koyuncu
Information Systems Engineering, Atılım Univ. _______________

Asst. Prof. Dr. Mustafa Sert
Computer Engineering, Başkent Univ. _______________

Ozan Küsmen, M.S.
Senior Engineer, ASELSAN _______________

Date: _______________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: EBRU DOĞAN

Signature :

ABSTRACT

CONTENT-BASED AUDIO MANAGEMENT AND RETRIEVAL SYSTEM FOR NEWS BROADCASTS

Doğan, Ebru
M.S., Department of Computer Engineering
Supervisor : Prof. Dr. Adnan Yazıcı
Co-Supervisor : Asst. Prof. Dr. Mustafa Sert

September 2009, 68 pages

The audio signals can provide rich semantic cues for analyzing multimedia content, so audio information has been recently used for content-based multimedia indexing and retrieval. Due to growing amount of audio data, demand for efficient retrieval techniques is increasing. In this thesis work, we propose a complete, scalable and extensible audio based content management and retrieval system for news broadcasts. The proposed system considers classification, segmentation, analysis and retrieval of an audio stream. In the sound classification and segmentation stage, a sound stream is segmented by classifying each sub-segment into silence, pure speech, music, environmental sound, speech over music, and speech over environmental sound in multiple steps. Support Vector Machines and Hidden Markov Models are employed for classification and these models are trained by using different sets of MPEG-7 features. In the analysis and retrieval stage, two alternatives exist for users to query audio data. The first of these isolates user from main acoustic classes by providing semantic domain based fuzzy classes. The latter offers users to query audio by giving an audio sample in order to find out the similar segments or by requesting expressive summary of the content directly. Additionally, a series of tests was conducted on audio tracks of TRECVID news broadcasts to evaluate the performance of the proposed solution.

Keywords: Content-Based Retrieval, News Broadcasts, Audio Classification and Segmentation, Audio Retrieval, Fuzzy

ÖZ

HABER YAYINLARI İÇİN İÇERİK TABANLI SES YÖNETİM VE ERİŞİM SİSTEMİ

Doğan, Ebru
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Adnan Yazıcı
Ortak Tez Yöneticisi : Y. Doç. Dr. Mustafa Sert

Eylül 2009, 68 sayfa

Ses sinyalleri çoklu ortam içerik analizi için zengin anlamsal ipuçları sağlayabilirler, bu yüzden ses bilgisi içerik tabanlı çoklu ortam dizinleme ve erişimi için kullanılmaya başlanmıştır. Miktarı artan ses verileri nedeniyle verimli erişim sistemlerine olan talep artmaktadır. Bu tez çalışmasında haber yayınları için tam, ölçeklenebilir ve genişletilebilir ses tabanlı bir içerik yönetim ve erişim sistemi sunulmaktadır. Önerilen sınıflandırma sistemi ses sinyallerinin sınıflandırılmasını, bölütlendirilmesini, analizini ve erişimini dikkate alır. Sesin sınıflandırılması ve bölütlendirilmesi aşamasında, ses sinyalinin her bir alt bölümü sessizlik, saf konuşma, müzik, çevresel ses, konuşma üzerine müzik ve konuşma üzerine çevresel ses sınıflarına birden çok adımda ayrılır. Sınıflandırma için Destek Vektör Makineleri ve Saklı Markov Modelleri kullanılmış ve bu modeller MPEG-7 standardında tanımlı olan farklı öznitelikler kullanılarak eğitilmişlerdir. Analiz ve erişim aşamasında, kullanıcıların ses verisini sorgulaması için iki alternatif yol mevcuttur. Bunlardan ilki kullanıcıya anlamsal, alan tabanlı bulanık sınıflar (sunucu, reklam, muhabir, spor, geçiş, hava durumu ve olay yeri sesi) sağlayarak, ana ses sınıflarından soyutlanmasını sağlar. Diğer yol ise kullanıcıların örnek ses verileriyle sorgu yapmasına ya da doğrudan bir ses verisi içinde anahtar kavramları bulmasına olanak sağlar. Ayrıca önerilen çözüm yönteminin performansını değerlendirmek için TRECVID haber yayınları üzerinde bir dizi test yapılmıştır.

Anahtar Kelimeler: İçerik Tabanlı Erişim, Haber Yayınları, Sesin Sınıflandırılması ve Bölütlendirilmesi, Ses Erişimi, Bulanık

To My Husband

ACKNOWLEDGMENTS

I express my sincere appreciation to my supervisor, Prof. Dr. Adnan Yazıcı for his guidance, criticism, wisdom, encouragement and insight throughout the research.

I am very grateful to my cosupervisor, Asst. Prof. Dr. Mustafa Sert for his advice, motivation, guidance and support through his previous works and experience.

I am thankful to my company ASELSAN Inc. for letting and supporting of my thesis.

I would like to thank to TÜBİTAK (The Scientific and Technological Research Council of Turkey) for providing financial support during my graduate study.

I am indebted to my husband for his invaluable patience, encouragement and help on my study.

TABLE OF CONTENTS

APPENDICES

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

In recent years, available news videos in digital archives are increasing rapidly and demand for efficient management and retrieval techniques is increasing correspondingly. For instance, news information providers would like to label a huge amount of news audio data in an easy way, and thereafter retrieve the data in a reliable way. Development of a complete content management and retrieval system requires a wide range of operations on audio streams, including temporal segmentation, analysis and retrieval of audio.

Audio classification and segmentation technologies are the first crucial step towards building such systems. This initial step is performed by low level perceptual features, though user tends to make queries on high level semantic content of the audio. Therefore, low level features needed to be processed to provide some higher level description of audio content. In this thesis work, we propose a complete content-based management and retrieval system for news broadcasts moving from low level features to a high level semantic description of the audio being examined. The proposed system is basically composed of a general sound classification and segmentation unit introducing new mixed type classes (e.g., speech over music, speech over environmental sound) which are suitable for multimedia indexing and retrieval, and a retrieval unit providing flexible querying of audio data.

Audio segmentation methods can be divided into two groups: change detection and audio classification. Change detection methods identify homogeneous audio segments by performing event detection algorithms. Speaker change detection is the most prominent application area based on these methods. The latter group, audio classification which is the task of classifying audio data into different classes has been implemented by using a number of different schemes. Feature and classification method selection are two challenging issues that must be crucially taken into consideration for this group. In terms of the former issue, MPEG-7 (formally called the Multimedia Content Description Interface) international standard provides simple, low complexity features that can be used to characterize any type of the sound [1]. For the latter issue, there are a number of classification methods that can be divided into rule-based and model-based schemes.

The rule-based approaches use simple rules deducted from the properties of the features. Since these rules depend on thresholds, they may not be able to classify the given test data set. To delve into more, rule based approaches sometimes are not managed to find any rule or rule set conforming with the given incomplete test data set. Hence they cannot output any class for that data set.

Model-based approaches fundamentally base on statistical models, such as Gaussian Mixture Model (GMM), Hidden Markov Model (HMM), Neural Network (NN), Support Vector Machine (SVM), and so forth. In this study, we make use of two classification methods, namely HMM and SVM. The reason why we chose HMM refers to the preference of it in Mpeg-7 audio classification tool [2]. SVM technique is chosen as another employed pattern classifier due to better performance it showed in sound classification over other classification methods [3, 4, 5, 6].

The notion of “content” in an audio stream is often abstract. In many cases, it is difficult for the system to explicitly represent this content [7]. User often finds it difficult to express the content. In such situations, Query-By-Example (QBE) is preferred. The user gives a sample query sound and the system retrieves the most similar sounds. However, when the user does not have any sample audio, the best way is to describe the intended audio by semantic key words.

In this thesis work, we present a complete content-based audio management and retrieval system for news broadcasts. The proposed system considers classification, segmentation, analysis and retrieval of an audio stream. In the sound classification and segmentation stage [8], an audio stream is segmented into six different classes: silence, pure speech, music, environmental sound, speech over music, and speech over environmental sound. In this study, both SVM and HMM are employed to classify audio segments into those audio classes. Mpeg-7 features are used to train SVM and HMM models. In addition, various audio classification experiments are presented to exploit the ability of Mpeg-7 features and the selected classification methods. In the analysis and retrieval stage [9], the segmented audio stream is described semantically by using domain based fuzzy classes (i.e., anchor, commercial, reporter, sports, transition, weatherforecast, and venuesound) and various alternative ways are provided for users to query the audio stream of news broadcasts.

Compared with the existing content-based audio management and retrieval systems, the major contributions and advantages of the proposed approach are as follows:

1. In this study, the high level semantic description of the audio is gradually generated allowing each stage to be replaced by alternative methods without affecting other stages. For instance, employing different classification and segmentation approaches does not affect the analysis and retrieval stage.

2. We propose a general solution in audio classification and segmentation of news broadcasts by introducing new mixed type classes (e.g., speech over music, speech over environmental sound) which are suitable for multimedia indexing and retrieval.

3. We present a comprehensive evaluation and analysis of the recognition accuracy of different Mpeg-7 audio features on news audio classification and segmentation. Additionally, we present a comparative study of SVM and HMM as the classifiers for the Mpeg-7 audio feature family.

4. In this study, inclusion of domain based fuzzy classes into the retrieval stage provides extracting information from experts rapidly and responding to user queries from this information. We assess the relationships between semantic classes (i.e., anchor, commercial) and acoustic classes (i.e., pure speech, music) in news domain and we present a method for retrieving domain dependent information from the segmented audio using these relationships.

5. In this study, users can query the audio data in various ways via query interfaces supplied by the proposed system. These query types can be listed as follows: temporal queries, temporal relationship queries, similarity search queries (QBE) and expressive summary search queries (ESS).

The rest of the thesis is organized as follows: Chapter 2 presents the researches done on content-based information management and retrieval. In Chapter 3, a set of Mpeg-7 audio low level descriptors are briefly explained. In Chapter 4, some of the main directions in the proposed management and retrieval system are outlined. Every technology we used is investigated and why we choose the corresponding technology is discussed. Experiments and evaluations on TRECVID audio data are given. Implementation of the proposed system is discussed in Chapter 5. The usage of the technology is explained by giving screenshots from the proposed system. The parts are related with the system modules. Finally, Chapter 6 provides conclusion with some future extensions of our proposed system.

CHAPTER 2

RELATED STUDIES

Great amount of researches have been done on content-based information management and retrieval on audio data. In order to develop a complete audio management and retrieval system, studies on audio classification, segmentation and efficient querying are inspected. The related studies are classified according to their major subjects and briefly explained in the following sections.

2.1 Classification and Segmentation of Audio

Content-based audio classification and segmentation is a basis for further audio/video analysis. If an audio clip can be automatically classified, management of it can be improved dramatically. Such a method is helpful in audio retrieval and video structure extraction.

2.1.1 General Audio Classification and Segmentation

The classification of audio data into single type classes (e.g., music, speech) is well studied, however there have been few studies focusing on general sound classification. Studies on general sound classification enhance their audio classification algorithms by using more audio classes and considering mixed type classes (e.g., speech over music).

Zhang et al. [10] build a hierarchical system in which the audio recordings are first classified and segmented into single type classes (i.e., speech, music, environmental sound, and silence) and then environmental sounds are classified into finer classes such as applause, rain, etc. A rule-based heuristic procedure is applied in the first stage and Hidden Markov Model (HMM) is used as the classifier in the second stage. They state that misclassification usually occurs in the hybrid sound which contains more than one basic type of audio.

Lu et al. propose a two-step scheme to classify audio clips into one of the single type audio classes [11]. First, the input stream is classified into speech and non-speech segments by a K-Nearest-Neighbor (KNN) classifier and Linear Spectral Pairs - Vector Quantization (LSP-VQ) analysis. Second, non-speech segments are further classified into music, environmental sound and silence, by a rule-based scheme. According to their experimental evaluation the total accuracy rate is over 96%.

In one of another studies of Lu et al., they compare performance of Support Vector Machine (SVM), KNN, and Gaussian Mixture Model (GMM) by including mixed type audio classes [5]. They show that accuracy of SVM-based method is much better than the others. When using testing unit above 0.3 s, the accuracy would be above 90%.

Chen et al. also conduct a study on classification of mixed type audio based on SVMs [3]. They compare their SVM results with the results obtained from KNN, Neural Network (NN), and Naïve Bayes (NB) methods, and showed that SVM performs better than the others. The reported accuracy of classification is 78%.

2.1.2 Mpeg-7 Based Audio Classification and Segmentation

A variety of signal features are proposed for audio classification and segmentation [12]. Most of these features consist of low-level features including parameters such as zero-crossing rate, bandwidth, spectrum flux, and sub-band energy [3, 4, 5, 6, 11]. The remaining features are Mel-Frequency Cepstral Coefficients (MFCCs) which have been widely used in speech recognition [13, 14, 15] and spectral features adopted by Mpeg-7 international standard [1].

Kim et al. propose an Mpeg-7 based audio classification technique for analysis of film material [16]. They apply Hidden Markov Models (HMMs) as classifiers using Audio Spectrum Projection (ASP) feature based on Audio Spectrum Basis (ASB). Two recognition systems are offered in the study, speaker recognition and sound effect recognition (e.g., laughter, telephone ringing).

Wang et al. present a new environmental sound classification architecture which fuses SVM and KNN [17]. Three Mpeg-7 low-level descriptors, namely, Audio Spectrum Centroid (ASC), Audio Spectrum Spread (ASS), and Audio Spectrum Flatness (ASF) are used to exploit their ability in environmental sound classification. An accuracy of 85% is reported.

Simmermacher et al. studied on classification of classical musical instruments in solo musical passages [18]. They group the instruments into four major classes: piano, brass, string, and woodwind. Nineteen features selected from the MFCC and the Mpeg-7 audio descriptors achieved a recognition rate of around 94% by the best classifier assessed by cross validation.

Xiong et al. present a comparative study for classification of sports audio [19]. Classification accuracies of Mpeg-7 audio features and MFCC are compared with three variations of HMM classifier. Around 90% accuracy is reported for all the combinations with the best being classification with Mpeg-7 features.

Different from the abovementioned studies, Wellhausen et al. apply an unsupervised learning algorithm to segment an audio data by searching similar sections of a song [20]. The search is performed on Mpeg-7 audio features. Unfortunately, proposed algorithm does not perform well for live recorded audio material due to tempo variations within a song.

It should be noted that these studies report satisfactory results for problems that are restricted to few types of audio classes. Our study differs from these above methodologies in that we deal with various types of audio classes as general as possible for the selected domain.

2.1.3 News Broadcasts Audio Classification and Segmentation

There exist many studies on news broadcasts audio classification and segmentation. Most of these studies present speaker detection, identification, or speech recognition systems. For instance, Zibert et al. propose a system for speaker-based audio indexing and an application for speaker-tracking in news broadcasts audio [21]. Audio segmentation, speech detection, speaker clustering, and speaker identification are four main building blocks of the proposed system.

Meinedo et al. develope a system of audio segmentation, classification, speaker clustering and anchor identification applied to a News Broadcasts task for the European Portuguese language [22]. Initially, changes in the acoustic conditions are detected and marked as segment boundaries. Then, speech and non-speech discrimination is performed for each segment respectively. Non-speech segments are discarded for further speaker gender and speaker identification processes. Background categorization for speech segments is also implemented, but successful results could not be achieved.

Nwe et al. propose a new approach towards news broadcasts audio segmentation [23]. Without considering commonly used classes, they segment audio streams into speech, commercials, environmental sound, physical violence and silence in multiple steps. As a classifier, multi-model HMM method is used and nearly 85% of accuracy is reported. This system is error prone due to deficient analysis of news domain. For instance, although speech with background music segments are assumed to present only in commercial audio, they also present within news story, such that, weathercasts or sport news. Indeed, all defined classes in non-commercial audio may also exist in commercial audio.

2.1.4 Audio Classification and Retrieval using Fuzzy Logic

Audio classification and retrieval applications using fuzzy logic are generally based on a rule-based system consisting of a set of IF-THEN rules, a set of facts, and an interpreter controlling the application of the rules, given the facts. The rules and facts are used to convert the high-level query given by the user to a low-level query that can directly use the features.

Exposito et al. present a fuzzy, rule-based speech/music discrimination approach for audio coding [24]. The classification task is performed by applying a Support Vector Machine (SVM) to the selected features. The final decision is made by a fuzzy expert system, which is designed to improve the accuracy rate provided by the SVM classifier. Inclusion of fuzzy expert system into the classification stage is claimed to improve the classification performance from 92% to 98%.

Nitanda et al. propose an audio-based shot classification method for audiovisual indexing [25]. The proposed method mainly consists of two parts, an audio analysis and a shot classification part. In the audio signal analysis, the probability that an audio signal belongs to the four audio classes, which are silence, speech, music and noise, is measured. In the shot classification part, fuzzy inference (based on fuzzy rules) is applied to roughly measure mixing rate of multiple audio sources and classify the shots accordingly.

Tao et al. present a fuzzy logic based system to label audio clips as speech or not [26]. Predefined rules are applied to the extracted features of audio segments to provide the final classification. Furthermore, a fuzzy variable speech-likelihood is introduced to express the degree of a clip belonging to speech. Proposed methodology is claimed to improve the accuracy of the speech detection.

Liu et al. focus on classification and retrieval of speech and sounds of music instruments using fuzzy logic [27]. Simulating the feature distribution, membership functions and rules are designed to construct a fuzzy classifier for each classification step. Initially audio is classified as speech and music with an accuracy of 92%, then speech clips are classified as female and male with an accuracy of 89%, and finally music clips are classified as percussion and others with an accuracy of 81%.

2.2 Search and Retrieval of Audio

In recent years, numerous audio recordings are generated. However, it is usually difficult to find and retrieve an audio clip relevant to the user's interest, since few audio clips are manually annotated. Currently, there are several papers addressing the problem of audio retrieval.

2.2.1 Content-Based Audio Retrieval

Spevak and Favreau present a prototype system, namely Soundspotter, for content-based audio section retrieval within an audio file [28]. In their work, the user first selects a reference passage and asks the system to retrieve similar occurrences of it in the same audio file. Then, the system carries out frame-based feature extraction using Mel-Frequency Cepstral Coefficients (MFCCs) and performs a pattern matching of the feature vectors afterwards. Finally, a prescribed number of best matches are retrieved and ranked by their similarity to the reference passage.

Wan and Liu introduce two relevance feedback techniques to carry out retrieval in audio domain [29]. The user first inputs a query example as common in Query-By-Example (QBE) paradigm, then a list of files according to the decreasing similarity is displayed to the user for listening and browsing. If the user marks some of the retrieved files as non-relevant, then the system updates the result in order to find more relevant files.

Chechik et al. propose content-based audio retrieval approach for retrieving sounds from free-form text queries rather than sound based queries [30]. Retrieval of sounds is done by acoustic features of the audio content itself rather than textual metadata. Using MFCC standard features, three learning approaches are employed to match sounds to text tags.

Virtanen and Helen propose different measures for estimating the similarity of two audio signals [31]. Similarity between audio signals is calculated by modeling the distributions of acoustical features using GMMs and HMMs, then measuring difference between the distributions by Kullback-Leibler divergence and Euclidean distance. Performance of the proposed methods in QBE is tested using a database consisting of speech, music, and environmental sounds. Although none of the tested measures is superior in comparison to the others, their retrieval accuracy is claimed to be better than the reference methods in their work.

Sert et al. propose a content-based retrieval system for querying and browsing of auditory data [32]. In their study they introduce a new method to extract semantically important portions (namely audio excerpts) of an audio and describe the content by a (dis)similarity matrix. Then they perform both structural and semantic queries on the generated descriptions. Structural queries are provided based on the generated audio excerpts via QBE paradigm whereas semantic queries are enabled by the provided sound effect database. In their system clients express their queries based on predefined keywords, such as gun-shot, explosion, animal sounds, and so forth. Despite the fact that the system is generic, the semantic queries are limited by the provided sound effect database.

2.2.2 News Broadcasts Audio Retrieval

Automatic information retrieval from news broadcasts is a challenging problem due to frequent and unpredictable changes that occur in speaker, speaking style, topic, background conditions, etc. Major studies on this recognition problem rely on visual information only, neglecting the rich supplementary source of the accompanying audio signal. However audio information can be even more important in certain applications where mostly unique and stable information is needed within the entire duration of the content.

Zhu and Ming present a scene classification and segmentation scheme using combination of audio and visual features [33]. First, a video source is split into audio and video streams which are segmented into one-second clips respectively. Then, each clip is classified into one of the classes (i.e., news, commercial, weather forecast, cartoon, MTV, tennis game, basketball game, and football game) by employing SVM classifiers.

Nakamura and Kanade introduce the Spotting by Association method, which detects relevant video segments by associating image data and language data [34]. Image clues are detected by image analysis and language clues are detected by language analysis to identify each segment respectively. Finally, segments having same class label are detected.

Finally, Bertini et al. present a complete system for content-based retrieval and browsing of news reports based on visual features extracted from video shots and textual strings extracted from captions and audio tracks [35]. Videos are segmented into shots by using only visual features, and then each shot is classified into anchorman or reporter classes based on a statistical approach. Further analysis is performed on additional information extracted from text captions and anchorman speech.

CHAPTER 3

MPEG-7 PART 4: AUDIO

In the previous chapter, variety of signal features proposed for audio classification is discussed. In this study, Mpeg-7 standard is chosen as a basis for the feature extraction. The Mpeg-7 standard is described in ISO/IEC 15938:4 [2], where it is divided into eight parts. In part 4, a set of audio low-level descriptors are described for the use in higher-level audio applications.

The audio framework consists of seventeen temporal and spectral descriptors which can be divided into six different groups as seen in Figure 3.1. Each group is briefly explained below depending on the explanations in ISO/IEC 15938:4 [2] and the book of *"MPEG-7 Audio and Beyond"* [1].

3.1 Basic

The two basic audio descriptors are temporally sampled scalar values to provide a simple and economical description of the temporal properties of an audio signal.

The AudioWaveform Descriptor provides an estimate of the signal envelope in the time domain by storing its minimum and maximum samples. It also allows economical and straightforward storage, display or comparison techniques of waveforms.

The AudioPower Descriptor describes the temporally-smoothed instantaneous power of the audio signal. In conjunction with other basic spectral descriptors (described below), it provides a quick representation of the spectrogram of a signal.

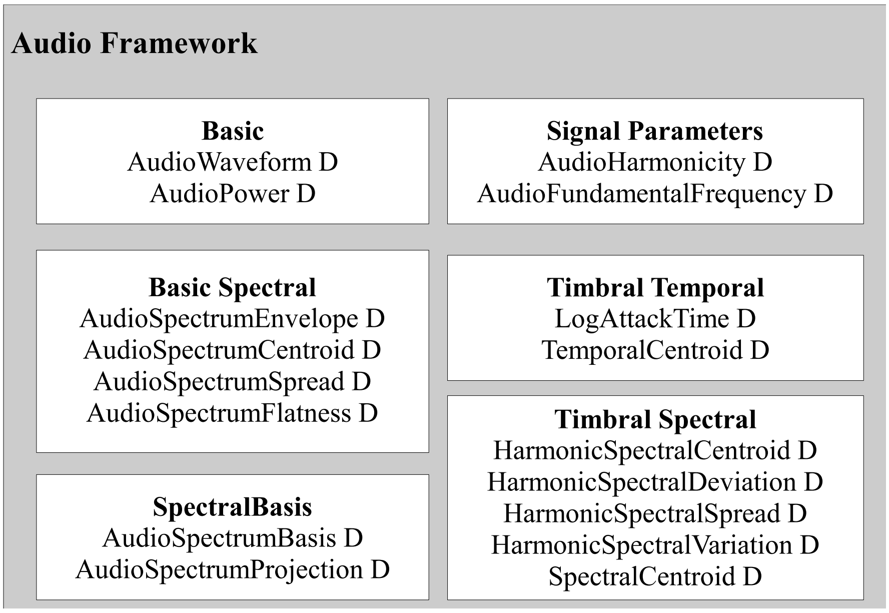


Figure 3.1: Overview of the Mpeg-7 Audio Framework

3.2 Basic Spectral

The four basic spectral audio descriptors all derive from a single time-frequency analysis of an audio signal. These descriptors are supposed to approximate the response of the human ear, so very important for content-based audio applications.

The first descriptor, the AudioSpectrumEnvelope (ASE) Descriptor, is a vector that describes the short-term power spectrum of an audio signal. It is obtained by summing the energy of the original power spectrum within a series of frequency bands. It may be used to display a spectrogram or as a general-purpose descriptor for search and comparison.

The AudioSpectrumCentroid Descriptor (ASC) describes the center of gravity of the log-frequency power spectrum. This descriptor gives information on the shape of the power spectrum. It indicates whether a power spectrum is dominated by low or high frequencies and can be regarded as an approximation of the perceptual sharpness of the signal.

The AudioSpectrumSpread Descriptor (ASS) is another spectrum shape descriptor that gives indications about how the spectrum is distributed around its centroid. Low ASS value means that the spectrum may be concentrated around the centroid, while high value reflects a distribution of power across a wider range of frequencies. This descriptor is designed to help differentiation of noise-like and tonal sounds.

The AudioSpectrumFlatness Descriptor (ASF) is a measure of how flat a particular portion of the signal is. More precisely, for a given signal frame, it consists of a series of values, each one expressing the deviation of the signal's power spectrum from a flat shape inside a predefined frequency band. A large deviation from a flat shape generally depicts tonal components. This feature can differentiate noise-like and impulse-like signals effectively. The spectral flatness coefficients may also be used as a feature vector for robust matching between pairs of audio signals.

3.3 Signal Parameters

The above-mentioned basic spectral descriptors can not reflect the detailed harmonic structure of periodic sounds because of a lack of frequency resolution. The two signal parameter descriptors provide some complementary information, by describing the degree of harmonicity of audio signals.

The AudioHarmonicity Descriptor provides a compact description of the harmonic properties of sounds. This descriptor can be used for distinguishing between harmonic sounds (e.g., musical sounds and voiced speech segments) and non-harmonic sounds (e.g., noisy sounds and unvoiced speech segments).

The AudioFundamentalFrequency Descriptor provides estimations of the fundamental frequency in segments where the signal is assumed to be periodic. The standard does not specify any normative extraction method, but provides overview of several widely used fundamental frequency estimation techniques.

3.4 Timbral Temporal

The two timbral temporal descriptors describe temporal characteristics of segments of sounds, and are especially useful for the description of musical timbre (characteristic tone quality independent of pitch and loudness). These descriptors are extracted from the signal envelope in the time domain. The signal describes the energy change of the signal and is generally equivalent to the so-called ADSR (Attack, Decay, Sustain, Release) of a musical sound.

The typical ADSR phases of a sound are:

- Attack is the length of time required for the sound to reach its initial maximum volume.
- Decay is the time taken for the volume to reach a second volume level known as the sustain level.
- Sustain is the volume level at which the sound sustains after the decay phase.
- Release is the time it takes the volume to reduce to zero.

The LogAttackTime Descriptor characterizes the ”attack” of a sound. This feature signifies the difference between a sudden and a smooth sound.

The TemporalCentroid Descriptor also characterizes the signal envelope, representing where in time the energy of a signal is focused. This descriptor may, for example, distinguish between a decaying piano note and a sustained organ note, when the lengths and the attacks of the two notes are identical.

3.5 Timbral Spectral

The five timbral spectral descriptors aim at describing the structure of harmonic spectra in a linear-frequency space. They are designed to be computed using signal frames if instantaneous values are required or larger analysis windows if global values are required.

The HarmonicSpectralCentroid Descriptor is the amplitude-weighted mean of the harmonic peaks of the spectrum. It has a similar semantic to the other centroid descriptors, but applies only to the harmonic (non-noise) parts of the musical tone.

The HarmonicSpectralDeviation Descriptor measures the deviation of the harmonic peaks from the envelopes of the local spectra.

The HarmonicSpectralSpread Descriptor is a measure of the average spectrum spread in relation to the HarmonicSpectralCentroid Descriptor.

The HarmonicSpectralVariation Descriptor is the normalized correlation between the amplitude of the harmonic peaks between two subsequent time-slices of the signal.

The SpectralCentroid Descriptor is not related to the harmonic structure of the signal. It gives the power weighted average of the discrete frequencies of the estimated spectrum over the sound segment. This descriptor is very similar to the AudioSpectrumCentroid Descriptor, but specialized for use in distinguishing musical instrument timbres.

3.6 Spectral Basis

The two spectral basis descriptors represent low-dimensional projections of a high-dimensional spectral space to aid compactness and recognition. These descriptors are used primarily with the Sound Classification and Indexing Description Tools, but may be of use with other types of applications as well. The extraction of AudioSpectrumBasis Descriptor and AudioSpectrumProjection Descriptor is based on normalized techniques which are part of the standard: the Singular Value Decomposition (SVD) and the Independent Component Analysis (ISA). The AudioSpectrumBasis Descriptor is a series of (potentially time-varying and/or statistically independent) basis functions that are derived from the singular value decomposition of a normalized power spectrum. These functions are used to project high-dimensional spectrum into a low-dimensional representation contained by the AudioSpectrumProjection Descriptor.

The feature extraction of the Mpeg-7 Sound Recognition Classifier [2] is based on AudioSpectrumBasis Descriptor to attain a good performance by performing a balanced trade-off between reducing the dimensionality of data and retaining maximum information content.

AudioSpectrumProjection Descriptor is used together with the AudioSpectrumBasis Descriptor, and represents low-dimensional features of a spectrum after projection upon a reduced rank basis. Extraction of this feature mainly consists of a Normalized Audio Spectrum Envelope (NASE), a basis decomposition algorithm, and a spectrum basis projection, obtained by multiplying the NASE with a set of extracted basis functions.

CHAPTER 4

PROPOSED SYSTEM

As mentioned earlier, a complete audio management and retrieval system considers classification, segmentation, and efficient querying techniques. The focus in this thesis is on general and complete audio management and retrieval system, therefore this chapter will outline some of the main directions in the proposed management and retrieval system.

4.1 General Classification Approach

There are many different ways to implement automatic classification of audio data (see section 2.1). The available data to be classified may be processed in more or less efficient ways to give informative features of the data by different classifiers and different features. In a basic scheme of automatic classification, initially audio stream is processed to isolate specific characters of it. These characters, which contain several descriptive measures, are stored in a feature vector. Then, this feature vector is used to classify the sequence into defined classes, which is done by the classifier.

The block diagram of the proposed classification approach is illustrated in Figure 4.1. First, various Mpeg-7 audio low-level descriptors are extracted from the audio source to obtain description of its content. Audio Power (AP), Audio Spectrum Centroid (ASC), Audio Spectrum Spread (ASS), Audio Spectrum Flatness (ASF) and Audio Spectrum Projection (ASP) are the Mpeg-7 features chosen due to their effectiveness in capturing structures of different audio classes [1].

Then, the input audio data is first classified into silence and non-silence by the silence detector. If it is classified as silence, no further step is required. Otherwise, audio data is classified into

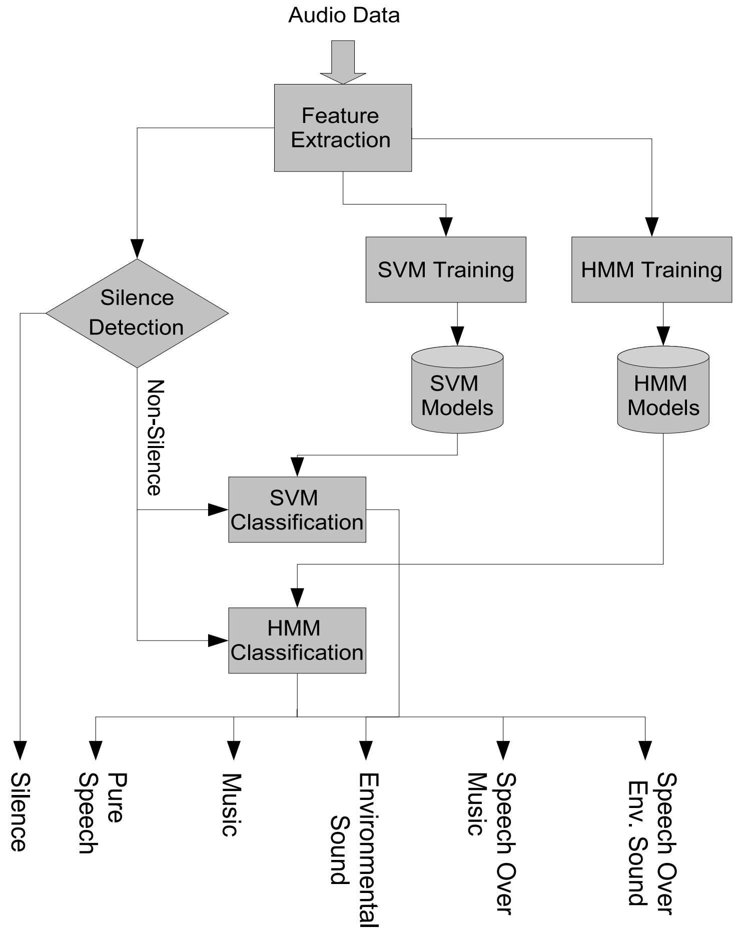


Figure 4.1: Block Diagram of the Proposed Classification Approach

one of the audio types: pure speech, music, environmental sound, speech over music, and speech over environmental sound. These sound classes represent a wide range of acoustic structure of news videos and this is why we chose them.

After the silence detection, the extracted features are used as the inputs in classification which is the last stage. This stage, of which result is one of the predefined non-silent sound classes, maintains the main classification performed by previously trained SVM and HMM models.

A comprehensive description of the proposed classification approach components are given in the following sections.

4.1.1 Feature Extraction

The Mpeg-7 Audio standard contains description tools for audio describing content as explained in the previous chapter. In this thesis, silence segments are detected by using Mpeg-7 Audio Power (AP) feature. Audio classification is performed by using two sets of Mpeg-7 features respectively. The first set contains Audio Spectrum Projection (ASP) coefficients, while the second set namely ASCSF contains Audio Spectrum Centroid (ASC), Audio Spectrum Spread (ASS), and Audio Spectrum Flatness (ASF) features. Features in the second set are combined as one feature vector.

Extraction of these features is normative and is based on the audio signal itself. Mpeg-7 Audio Annotator and Fact Extractor [36] is employed to extract the required Mpeg-7 features of audio data used in our experiments.

4.1.2 Audio Classification

Once feature extraction is completed for the data set, standard statistical classifiers are trained and used to predict the class label of previously unknown audio data. Support Vector Machine (SVM) and Hidden Markov Model (HMM) are two supervised learning algorithms that we use in our thesis work.

4.1.2.1 Support Vector Machine

Support Vector Machine (SVM) [37] is a binary classifier introduced by Vapnik in 1995. The principle of SVM classification can be described as separating the two classes by a linear hyper-plane which is induced from positive and negative examples in the training set. Consider the example [38] in Figure 4.2. Here there may be so many hyper-planes that might classify the data, but the one that maximizes the margin (maximizes the distance between it and the nearest data point of each class) is the optimal one, due to the larger the margin the lower the generalization error of the classifier. This optimal hyperplane is expected to work well on unseen examples, i.e., it generalizes well.

Consider a set of training data with l training vectors belonging to two separate classes. Each training vector is denoted by (X_i, y_i), where $i = 1, ..., l$, $X_i = \{x_i, ..., x_n\}$, and $y_i \in \{+1, -1\}$. This

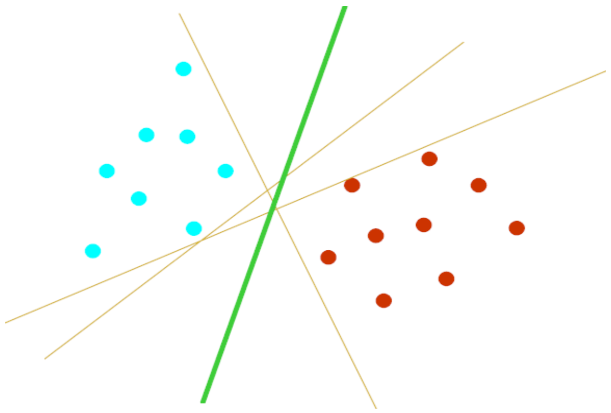


Figure 4.2: Optimal Separating Hyperplane

set of vectors is expected to be optimally separated by a hyperplane of equation $W \cdot X + b = 0$, $W \in R^N$ and $b \in R$ where X is the input vector, W is the vector perpendicular to the hyperplane, and b is a constant.

The decision function of classifying an unknown point x is defined as (4.1):

$$f(x) = \operatorname{sgn}(w \cdot x + b) = \operatorname{sgn}\left(\sum_{i=1}^{N_s} \alpha_i m_i x_i \cdot x + b\right) \tag{4.1}$$

where N_s is the support vector number, x_i is the support vector, α_i is the Lagrange multiplier and $m_i \in \{-1, +1\}$ describes which class x belongs to. More details on the mathematical background of SVM learning can be found in [37, 38, 39].

The feature distribution of the extracted audio data for different classes has overlapping and interwoven areas, so we could not linearly separate them in the given input space. For non-linearly separable data, the SVM uses kernel method to map the feature vectors into a higher dimensional space in which linear separation of the training set is possible. On using kernel functions to construct an optimal hyperplane instead of standard inner product $x \cdot y$, the decision function becomes (4.2):

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{N_s} \alpha_i m_i \,\mathrm{K}(x_i, x) + b\right) \tag{4.2}$$

Typical kernel functions are depicted in formulas in (4.3), (4.4), (4.5):

Linear Kernel:

$$\mathrm{K}(x, y) = x \cdot y \tag{4.3}$$

Polynomial:

$$\mathrm{K}(x, y) = (\gamma \cdot x \cdot y + c)^{d} \tag{4.4}$$

Radial Basis Kernel:

$$\mathrm{K}(x, y) = \exp\left(-\gamma \cdot \|x - y\|^{2}\right) \tag{4.5}$$

In our thesis work, we tried all abovementioned kernel functions to separate audio data into separate classes and achieved fulfilling results by using the non-linear kernel function Radial Basis Function (RBF).

4.1.2.2 Hidden Markov Model

Hidden Markov Model (HMM) is our second classification method which is a highly effective classifier for time series recognition applications. Although HMM was initially introduced in the late 1960s by Baum and his colleagues [40, 41, 42, 43, 44], it became popular in the late 1980's. A good tutorial on HMM is given by Rabiner [45].

HMM models process with time varying characteristics. An HMM model can be described as [1]:

- A set of N_s states $\{S_i\}$.
- A set of state transition probabilities $\left\{a_{ij}\right\}$, where a_{ij} is the probability of transition from state S_i to S_j.
- A set of d-dimensional probability density functions $\left\{b_j(x)\right\}$, where b_j is the density function of state S_j.
- A set of initial state probabilities $\{\pi_i\}$, where π_i is the probability that S_i is the initial state.

An example of an HMM model is shown in Figure 4.3. The system starts at time 0 in a state S_i with a probability $\{\pi_i\}$. When in a state S_i at time l the system moves at time $l + 1$ to state S_j with a probability a_{ij} and so on, generating a sequence of L observation vectors x_l.

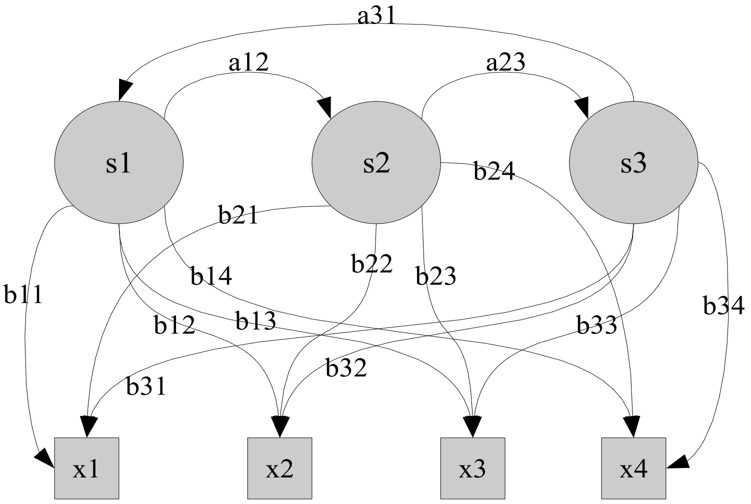


Figure 4.3: A Sample of an HMM Model

We estimated HMM parameters for each predefined audio classes by using the well-known Baum-Welch algorithm [45]. Taking the initial values for all the parameters, Baum-Welch finds the optimum values for the parameters by iterative reestimations. This problem is called "training problem" in HMM applications. The initial guess for the parameters is very important since this algorithm finds only locally optimum values. Therefore, instead of assigning initial values randomly, we use K-Means Clustering algorithm to estimate these values.

After specification of complete parameter set of HMM parameters, the classification problem turns to an "evaluation problem", namely given a model and a sequence of observations, how do we compute the probability that the observed sequence is produced by the model. We use Forward-Backward algorithm [45] to select the audio class for the given sequence.

Besides "training problem" and "evaluation problem", there exists one basic problem most applications reduced to solve, which is called "finding the correct state sequence". It can be defined as given the parameters of the model and a particular output sequence, find the state sequence that is most likely to have generated that output sequence. We have not dealt with this problem, because it is out the scope of our proposed classification approach.

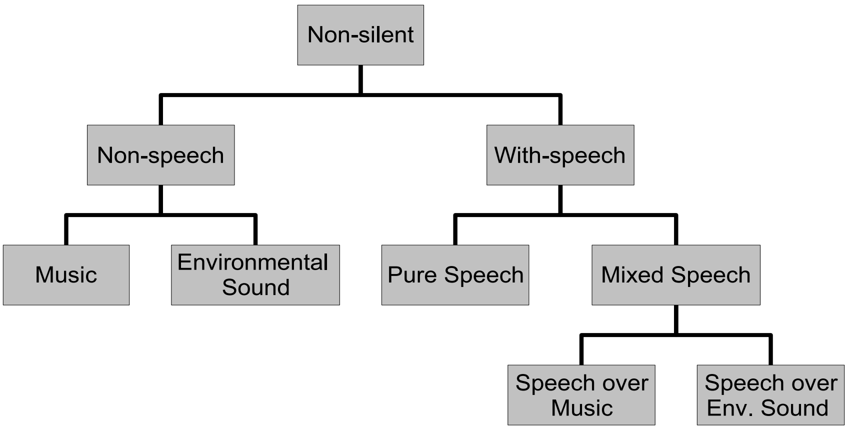


Figure 4.4: Audio Classification Tree

4.1.3 Multi-Class Classification

In our thesis work, an audio clip is classified into six different classes. Those are silence, pure speech, music, environmental sound, speech over music, and speech over environmental sound. Initially, input audio stream is segmented into silent and non-silent frames by a rule-based classifier instead of SVM or HMM.

We construct a binary tree for classification of non-silent frames, as shown in Figure 4.4. Starting from the root node, recursively each child pair is compared and "winner" of those becomes parent node for other comparison. Finally reached leaf node is selected as the current class.

Depending on the applied method, SVM and HMM models construct our classification decision tree. For SVM classification, each branch in the tree is associated with an SVM model. On the other hand, for HMM classification, each branching nodes and leaves is associated with an HMM model.

Selection of "winners" by HMM nodes is performed by calculating the probabilities of an observational sequence for each HMM model at the same level by using Forward-Backward algorithm [45], and selecting the one having the higher probability.

SVM classifier labels only one vector at a time, this is why we apply frame-based sound classification [17] to label the observational sequence. The process of frame based classification can be explained as follows: Each feature vector in the same observation sequence is tagged as +1 or −1 separately according to the decision function (4.2) of the current SVM classifier. Then, sum of the tags is calculated. If the sum of all tags in the observation sequence is greater than zero, this sequence is classified to +1 class. Otherwise, it is classified to −1 class.

4.1.4 Experiments and the Evaluation for the Classification Approach

The data used in our experiments, conducted for the proposed classification approach, are amassed from real news broadcasts from TRECVID 2003 ABC World News Tonight and CNN headline news, the Internet [46], and music CDs. The data obtained from these sources are segmented and labeled in terms of the mentioned six classes manually. Total duration of audio clips in the data set is 3 hours and 30 minutes. More specifically, 65 minutes of pure speech, 43 minutes of music, 31 minutes of environmental sound, 45 minutes of speech over music (S+M), and 26 minutes of speech over environmental sound (S+E) is included in the data set. Pure speech and with speech audio clips involve males and females at different ages at studio or outdoor. Music and with music clips involve songs and instrumental music covering as many styles as possible, such as rock, pop, jazz, classical, etc. Environmental sound and with environmental sound clips also involve noise, various kinds of environmental sound, or any sound not classified as (with) speech or (with) music. For each sound class, half of the sound files are utilized for training and the others are used for testing.

In these experiments, we compared the performance of both SVM-based and HMM-based methods on audio classification and tested the effectiveness of each feature set by employing both SVM and HMM classification methods.

Most speaker recognition systems build left-right models for HMM classification. However, we did not prefer that topology due to temporal structure of our data. We built a 5-state ergodic model, where each state can be reached from any other state and can be revisited after leaving.

RBF kernel is used in the SVM classifier. In order to find out the optimal SVM parameters (γ, C) , we used a small portion of the training set as validation data. C is cost of the penalty of the errors in SVM classification. Based on the 5-fold cross validation results practiced by trial and error, we set parameters of classifiers as $(\gamma = 6, C = 20)$ for the ASP feature set and

($\gamma = 2, C = 20$) for the ASCSF feature set.

Experimental results of the proposed classification approach are listed in Appendix A from Table A.1 to Table A.4. The classification accuracy, which is defined as the ratio of correctly classified samples over the actual time of the test data, is measured for each binary classification. The number of correctly classified samples is denoted by “hit” and the number of misclassified samples is denoted by “miss” in the tables.

Between the two sets of Mpeg-7 audio features, ASCSF feature set yields better performance than the ASP feature set in this study. We performed comparisons between two Mpeg-7 feature sets for both SVM and HMM classifiers. Accuracy rates using ASP are significantly lower than the accuracy rates of ASCSF feature set, that is, ASCSF feature set has the highest class separability for mixed type classes in news domain. To illustrate it further;

- The highest accuracy rate for discrimination of “with-speech” and “non-speech” is 91.9% and achieved by HMM using ASCSF feature set.
- The highest accuracy rate for discrimination of “pure speech” and “mixed speech” is 89.4% and achieved by SVM using ASCSF feature set.
- The highest accuracy rate for discrimination of “speech over environmental sound” and “speech over music” is 85.2% and achieved by HMM using ASCSF feature set.
- The highest accuracy rate for discrimination of “environmental sound” and “music” is 95.6% and achieved by SVM using ASCSF feature set.

Complexity wise, SVM takes extremely long time to train whereas training of HMM takes only minutes. For the case of feature sets, extraction of ASP feature is time and memory consuming compared to ASC, ASS, and ASF features.

4.2 Audio Segmentation

Audio segmentation is the task of segmenting an audio stream into homogenous regions. Homogeneity means arranging same type of data in one class. In our thesis, segmentation of an audio stream consists of three phases: silence detection, classification of non-silent frames, and smoothing.

4.2.1 Step 1: Silence Detection

Objective of this step is to extract silent and non-silent segments, and then obtain an initial segmentation for the next step. Kiranyaz et al. [47] propose an approach for silence detection. In the proposed approach, input audio stream is first divided into frames and then silence detection is performed per frame. On the other hand, we first detect silence segments and then divide non-silent segments into sub-segments.

The minimum (P_{min}), maximum (P_{max}), and average $\left(P_{\mu}\right)$ AP values are calculated from 10 ms sub-frames for the entire audio clip. To ensure the presence of audible content, two conditions are checked:

1. $P_{max} >$ Min. Audible Power Value
2. $P_{max} \geq P_{min}$

Once the presence of some non-silent is confirmed, then a threshold value (T) is calculated according to (4.6).

$$\mathrm{T} = P_{min} + \lambda_s \left(P_{\mu} - P_{min}\right), \quad where \quad 0 < \lambda_s \leq 1 \tag{4.6}$$

λ_s is the silence coefficient, which determines the silence threshold value between (P_{min}) and $\left(P_{\mu}\right)$.

If all samples of AP feature values for at least one second duration of an audio signal are less than the calculated threshold (T), then that segment is classified as silent, otherwise non-silent.

4.2.2 Step 2: Classification of Non-Silent Frames

The unit duration is set to one second in our work, thus non-silent segments are divided into one second sub-frames for further classification.

First of all, sub-frame is classified into non-speech and with-speech classes. This process goes on sequentially until the end of the audio stream reached and is also valid for the rest. Then, non-speech is further classified into music and environmental sound, and with-speech is further classified into pure speech and mixed speech classes. Finally, mixed speech is classified into speech over music and speech over environmental sound classes.

4.2.3 Step 3: Smoothing

Finally, two smoothing rules are applied to prevent some misclassification. Misclassification decision is taken under the assumption that audio stream is continuous in video programs, and it is nearly impossible to change the audio types suddenly and frequently. However, there are some special cases that should be kept in mind in the smoothing process. For example, music and speech over music classes are generally overlapped (e.g., commercial breaks), so successive interchanges of these classes should be ruled out. Similarly, environmental sound and speech over environmental classes are generally overlapped (e.g., news from the outdoor reporter), and hence interchanges of these classes should also be ruled out.

In the smoothing process 3-s sequence is considered at a time s_0, s_1, s_2 stands for the audio type of previous 2-s, previous second and current second, respectively.

Silence frames are considered as breakpoints, so we perform smoothing separately to each non-silent segment between two silence frames. Additionally, a parameter called "smoothing accuracy" is stored for each frame. The value of this parameter is set to 1 prior to the smoothing process, and may be changed after the smoothing process if class of the frame is changed.

Rule 1: $(s_1 \neq s_0 \wedge s_0 = s_2) \Rightarrow s_1 = s_0$

This rule implies that if class of the middle frame is different from the other two whereas the other two are the same, the middle one is considered as misclassification. For instance, if we detect a pattern of consecutive sequence like "pure speech-music-pure speech", it is most likely the sequence should belong to pure speeches. Besides, "smoothing accuracy" of the middle frame is set to 0.75.

Rule 2: $(s_1 \neq s_0 \wedge s_0 \neq s_2 \wedge s_2 \neq s_1 \wedge s_1 \neq Env.Sound \wedge s\_accuracy > 0.5) \Rightarrow s_1 = s_0$

This rule implies that if all classes of the frames are different from each other, class of the middle frame is not "environmental sound", and "smoothing accuracy" is greater than 0.5; then rectify class of the middle frame according to its previous audio type. For instance, if we detect a pattern of consecutive sequence like "pure speech-music-speech over environmental sound", then re-label the middle frame as "pure speech". Also, "smoothing accuracy" of the middle frame is set to 0.5. Since "environmental sound" is highly possible to appear in 1-sec window and can be changeover from one type of class to another, its class is not rectified.

Following the smoothing process, we group temporally adjoining frames together if they share the same audio type and average their "smoothing accuracy" values. As a result, the entire audio sequence is partitioned into homogeneous segments with each having a distinct audio class label and an approximate smoothing accuracy value.

After performing the segmentation process, extracted information is stored in a file to be queried and accessed thereafter. Format of this file is so simple that every line in the file corresponds to an audio segment and every entry in a line corresponds to a feature of this segment. Class label, starting time, end time and precision are four basic features of an audio segment and written to the file respectively.

4.3 Audio Content Retrieval

The proposed segmentation process mentioned in the previous section is also convenient for many other application domains since no domain-dependent knowledge about the structure of the audio data is used and the resulting classes are general and comprehensive enough. However users may tend to make more domain dependent queries according to their interests and preferences regarding audio content.

In this context, we provide two alternative ways for the user to query the audio data. One way is describing the requested audio semantically by using domain based fuzzy classes instead of main acoustic classes. The other way is providing an audio example and requesting similar audio segments or directly requesting summary of the content. In the following sections, diverse content retrieval capability of the proposed system is introduced.

4.3.1 Fuzzy Classes

The retrieval process of the proposed system is domain specific; as a result user tends to make domain-dependent queries. In order to address this requirement, we defined human understandable concepts in terms of main acoustic classes computed in the segmentation process. As a result, our system is able to support higher level queries through provided GUIs.

We examined general structure of television news broadcasts and consulted domain experts to assess semantic classes and their relationships with acoustic classes (e.g., pure speech, music). In this way we have a method for rapidly extracting information from experts and responding to user queries from this information. We defined seven semantic classes for the news broadcasts domain, i.e., anchor, commercial, reporter, sports, transition, weatherforecast, venuesound. Domain knowledge-dependent definition of these classes can be explained as follows:

- Anchor: Anchor segments cover the news reported by the anchorperson. Since the person breaks news in the studio, recorded audio is mostly noise free and in high quality. However, it would be possible to hear music or any kind of environmental sound while anchor speaking.

- Commercial: Commercial segments cover the period of commercial breaks. These segments are composed of uninterrupted background music and speeches at times.

- Reporter: Reporter segments often provide information on the location and people presented. Since news of this kind is generally reported at the outside, speech of the reporter can not be recorded as pure speech but speech over a noisy environment.

- Sports: Sports segments cover many aspects of human athletic competition like competitive events, athletes, business of sports, etc. Sport announcing refers to any type of speech over a very noisy background or just the background noise itself.

- Transition: Transition encapsulates segments of intro, outro (opposite of intro), and transition parts which occur in the news videos. Transition segments are theme music played at the beginning, at the end, or at transition to a different headline.

- WeatherForecast: Weather forecast news cover prediction of weather for a future time at a local or national region usually with accompaniment of constant background music.

- VenueSound: Venue means the location for a significant event. In news domain we define “venue sound” as any type of sound effect or background noise significant for the news content. Cheering and applause of people, explosion sound, traffic noise, etc. are examples of venue sounds.

In our study, we consider defined semantic classes as fuzzy sets and main acoustic classes as the elements of these fuzzy sets. Classically, an element may or may not belong to a set. This concept of crisp sets may be extended to fuzzy sets with the introduction of the idea of partial truth. Thus, in fuzzy theory, an object (element) may be a member of a set to a certain degree which we call membership degree. Membership degrees of each acoustic class to the defined fuzzy sets are illustrated in Table 4.1. Let the class label of an audio segment as "Speech over Music". This class label has membership in four fuzzy sets that were defined. However, their truthness differs i.e., 0.8 on commercial, 0.6 on weather forecast, 0.1 on anchor, and 0.1 on transition.

Table 4.1: Membership Degrees of Acoustic Classes to the Defined Fuzzy Sets

	Env.Sound	Music	PureSpeech	S+E	S+M
Anchor			0.8	0.1	0.1
Commercial		0.2			0.8
Reporter			0.2	0.8	
Sports	0.4			0.6	
Transition		0.9			0.1
VenueSound	0.9	0.1			
WeatherForecast			0.4		0.6

Parameters in Table 4.1 have been experimentally determined. It is not possible to compute them mathematically because perceptual information from expert listeners must be taken into account. In order to design the system, numerous real news broadcasts from TRECVID 2003 ABC World News Tonight and CNN headline news are available. These files are used to build the knowledge base, which includes all membership values that allow users to make semantic queries in news domain. The proposed retrieval system can be applied to many other domains by only changing the defined semantic classes and their membership degrees without requiring any retraining.

4.3.2 Fuzzy Based Retrieval of Audio

In general, we can say that previously mentioned segmentation process will generate a set of Labeled Audio Segments $\{LAS\}$. A labeled audio segment is defined as the pair:

$$LAS = \langle S, C \rangle , \tag{4.7}$$

where S contains audio information (e.g. start time, end time, smoothing accuracy) in the segment of interest and C is a class label (i.e., pure speech, music, environmental sound, speech over music, and speech over environmental sound).

Now, we can define a Fuzzy Labeled Audio Segment $\{LASf\}$ as the following triple:

$$LASf = \langle S, FC, D \rangle \,, \tag{4.8}$$

where S contains audio information (i.e., start time, end time, acoustic class label, smoothing accuracy), FC is a fuzzy class label, and D is the membership degree of S in the fuzzy set FC. Thus, a set of fuzzy labeled audio segments $\{LASf\}$ which is the fuzzy version of the previously defined set of crisp labeled audio segments is retrieved. Key to the interpretation of above-mentioned fuzzy classes and consistency between the crisp labeled audio segments and the domain expert information is the value of membership degree D. This value is defined within a range of $[0, 1]$.

Essentially the proposed retrieval system does not include fuzzy based classification. In our system, we use fuzzy set theory for flexible retrieval of audio data. Segmentation results and the membership degrees of acoustic classes to the defined fuzzy sets are two decision criteria for the retrieval system. For instance, once a user inputs a query like “Find me all possible *commercial* segments”, all *music* and *speech over music* segments are retrieved. Then, the system combines these segments as $\langle S, commercial, D \rangle$ and assigns 0.2 (membership degree of *music* to the *commercial*) to D for the *music* segments and 0.8 (membership degree of *speech over music* to the *commercial*) to D for the *speech over music* segments. As a result, all possible commercial segments are ordered according to their membership degrees.

4.3.3 Temporal and Temporal Relationship Queries

Audio data of any video represents information related to a timeline of events. Therefore addressing temporal offsets and time intervals, or comparing timestamps of two audio segments are crucial for content description. Temporal and temporal relationship queries are two kinds of queries directly related to timestamps of audio data and these types of queries are supported in our system.

Temporal query indicates temporal segments of interest and based on fuzzy classes defined in the previous sections. Retrieval of audio instants, intervals, and periods are supported in our system.

Temporal relationship query is also based on fuzzy classes but unlike temporal query, it represents comparative temporal relationships of two audio segments. There exist many temporal relationships in video/audio object, such as before, after, meets and so on. Allen has concluded thirteen correlations [48] in two temporal intervals some of them are shown in Table 4.2. Considering B and C as two different audio intervals, third column shows the position of these in timeline that suits the relationship. Only *before*, *after*, *meets*, *starts with*, and *ends with* relationships are meaningful in our system.

Table 4.2: Temporal Relationships

Relation	Symbol	Inverse	Meaning
B *before* C	b	bi	BBBB CCCC
B *meets* C	M	mi	BBBBCCCC
B *overlaps* C	o	oi	BBBB CCCC
B *during* C	d	di	BBBB CCCCCCCC
B *equal* C	e	e	BBBB CCCC

4.3.4 Similarity Search

Unlike the traditional way of using keywords as input to search for the audio segments, query examples can be used as input to search for similar audio segments. This type of queries is called "query by example". When a user inputs a query audio file and requests for finding relevant files to the query, both the query and each audio segment in the segmented audio file are represented as feature vectors. Similarity measurements between the vectors of query and audio segments are adopted. Then, a list of audio segments according to the decreasing similarity is displayed to the user for listening or browsing.

In this study, ASF audio feature is selected as the feature vector for similarity measurements and similarity between two series of ASF feature vectors is measured by employing a correlation function which computes the correlation coefficient of A and B, where A and B are the feature vector representations of two audio segments of the same size. The correlation function is used as follows:

$$\frac{\sum_m \sum_n \left(A_{mn} - \bar{A}\right)\left(B_{mn} - \bar{B}\right)}{\sqrt{\left(\sum_m \sum_n \left(A_{mn} - \bar{A}\right)^2\right)\left(\sum_m \sum_n \left(B_{mn} - \bar{B}\right)^2\right)}} \tag{4.9}$$

Let $f(x, y)$ and $w(x, y)$ be two feature matrices of size $M \times N$ and $J \times K$ respectively where $J \leq M$ and $K = N$. $f(x, y)$ is the feature matrix of an audio segment in which $w(x, y)$ is searched. Maximum correlation between f and w matrices are calculated by sliding the matrix w over f and computing the correlation coefficient for every window position. This generates a correlation coefficient array, which is then processed to extract the window position which causes the correlation coefficient to peak.

Our system supports three types of queries for similarity search which are “point query”, “k-nearest neighbor query”, and “range query”. That is, “point query” retrieves the most similar signal, “k-nearest neighbor query” retrieves the k best matches and “range query” retrieves the signals having similarity between the predetermined ranges.

4.3.5 Key Concept Detection

In this thesis, we apply the method developed by Sert et al. to detect repetitive structures [32] within speech segments of news broadcasts. Repetitive structures are the patterns repeating themselves over the time. This method relies on detecting abrupt changes in feature values of audio and advancing it by realizing a structural similarity analysis technique.

Repetitive structures are suitable for content-based audio information retrieval systems, i.e., audio indexing, browsing, or thumb nailing. Additionally, extraction of these structures can be defined in terms of key concept for speech.

Key concept detection algorithm can be divided into two main stages: structural similarity analysis and extraction of expressive patterns. These stages are briefly described in the following sections. Detail information regarding this algorithm can be found in [32].

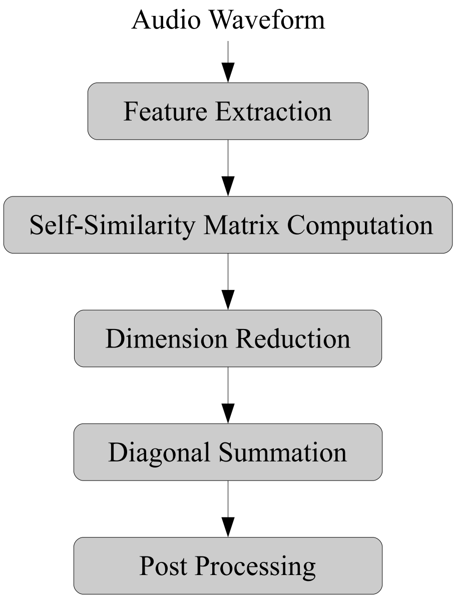


Figure 4.5: Block Diagram of the Structural Similarity Analysis Approach

4.3.5.1 Structural Similarity Analysis

The block diagram of the applied structural similarity analysis approach is depicted in Figure 4.5.

Feature extraction is the initial step for similarity analysis. MFCC is suggested for speech signals and Mpeg-7 ASF is suggested for music signals as the underlying feature set. However, we implemented the method by utilizing the Mpeg-7 ASF feature as one of our goals is to experience the abilities of Mpeg-7 features for different aspects of content-based analysis and retrieval. Extraction of Mpeg-7 ASF feature is depicted in section 3.2.

The next step following the feature extraction is to form a similarity matrix which will be used to distinguish similar (or dissimilar) regions within the audio signal. Let V_i and V_j be the feature vectors of frames i and j in the feature matrix. Then the similarity between the frames is defined by the *Euclidean* norm as follows:

$$s\left(V_i, V_j\right) = \sqrt{\sum_{k=1}^{m}\left(V_{i_k} - V_{j_k}\right)^2} \quad (i, j, m \in Z^{+})\,. \tag{4.10}$$

where $s\left(V_i, V_j\right)$ denotes the similarity between frame i and frame j, whereas m represents the vector length. This results in a similarity matrix S with the dimension of $n \times n$, where n is the number of feature vectors.

$$S = \begin{pmatrix} S_{1,1} & \dots & S_{n,1} \\ \vdots & \ddots & \vdots \\ S_{1,n} & \cdots & S_{n,n} \end{pmatrix} \tag{4.11}$$

This form of similarity matrix does not reveal the repetitive patterns, since discrete values of feature vectors having significant differences between them may be seen very close to each other in this initial form. Therefore further steps are needed to exploit repetitive patterns.

ASF feature vector is calculated in every 30 ms of an audio signal; as a result floating numbers for every 30 ms are held in the similarity matrix. Considering for example audio signal length of one minute, $2{*}10^3{\times}2{*}10^3 = 4{*}10^6$ floating numbers are stored. However such a resolution does not represent significant information for the extraction of repetitive patterns. Therefore the similarity matrix is compressed by factor b and they obtain a compressed similarity matrix with the dimension of $[n/b] \times [n/b]$. The compressed matrix is obtained by moving a kernel of size $b \times b$ over the similarity matrix, and then determining the minimum value from the corresponding area. The resulting matrix is as follows:

$$S = \begin{pmatrix} S_{1,1} & \dots & S_{p,1} \\ \vdots & \ddots & \vdots \\ S_{1,p} & \cdots & S_{p,p} \end{pmatrix} \quad p = \frac{n}{b} \tag{4.12}$$

Repetitive patterns are expected to be seen as diagonal stripes within the similarity matrix. To this end, the differences should be strengthened and horizontal and vertical lines disturbing the diagonal stripes should be removed. Considering the former issue, we implemented the diagonal summation process discussed in [32] to reveal the differences in the similarity matrix. The resulting similarity matrix is calculated as follows:

$$\hat{s}_{x,y} = \sum_{i=1}^{k} s_{x+i,y+i}, \tag{4.13}$$

where $\hat{s}$ represents the resulting similarity matrix and k represents the strengthen order. We decided the value of k as 8 on the strength of previously performed experiments. For the latter issue, we implemented the postprocessing algorithm explained in the study of Wellhausen and Crysandt [49]. Taking a block of the matrix with the dimension $d \times d$, the mean value of the main diagonal is divided by the mean of the whole block. The results of this operation are stored in the similarity matrix for all coordinates.

$$m_{diag,x,y} = \frac{\sum_{i=0}^{d} \hat{s}_{x+i,y+i}}{d} \quad m_{block,x,y} = \frac{\sum_{i=0}^{d} \sum_{j=0}^{d} \hat{s}_{x+i,y+j}}{d^2} \tag{4.14}$$

$$\tilde{s}_{x,y} = \frac{m_{diag,x,y}}{m_{block,x,y}} \tag{4.15}$$

4.3.5.2 Extraction of Expressive Patterns

In the previous section, we explained extracting of diagonal stripes from the similarity matrix which is calculated by using Mpeg-7 ASF feature vectors. The next step is to interpret the visible patterns.

An ideal similarity matrix [49] for a song with three refrains is shown in Figure 4.6 a). The main diagonal of this matrix is zero, because each feature vector is equal to itself. Diagonal stripes enumerated as (1) and (2) represent the first and second repetition of expressive patterns respectively. The first occurrence of the expressive patterns is visible as a diagonal pattern; it is hidden by the main diagonal. The diagonal stripe enumerated as (3) shows the repetition of the pattern (2), accordingly not used in the further processing. The projection of the patterns to the timeline is shown in Figure 4.6 b). The begin time and end time of the first occurrence of the expressive patterns are determined by projecting the stripes enumerated (1) and (2) to the main diagonal, and then to the time line.

In many cases the diagonal stripes representing the expressive summaries can not be extracted as easy as described above. Diagonal stripes may be seen in different sizes representing distinct repetitive pattern groups and stop-words. In this respect Sert et al. designed an algorithm to designate the most salient objects (expressive summaries) in the similarity matrix [32].

The proposed algorithm is composed of three steps for salient object extraction from a given similarity matrix. Firstly, two threshold values, namely $lThr$ (lower threshold) and $uThr$ (upper threshold), are computed to obtain the direction of maximum rate of change in the

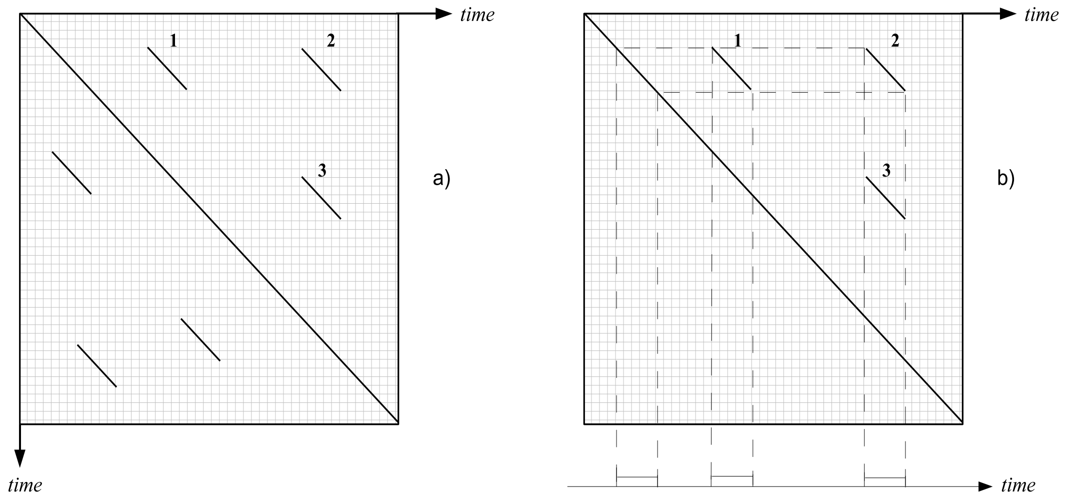


Figure 4.6: a) An ideal similarity matrix for a song with three refrains b) Refrain projection to the time line

pixel values of the similarity matrix. To this end, the upper triangular is sorted in ascending order and denoted by U, and then the first order of U is computed and denoted by ∇U. Since the similarity matrix is diagonally symmetric, lower triangular part of it is redundant. Afterwards, $lThr$ and $uThr$ are assigned to the first and second minimum values of ∇U, respectively. Consequently, a thresholding is performed over the similarity matrix as follows:

$$\acute{s}(x,y) = \begin{cases} 0 & \text{if } uThr \geq \tilde{s}(x,y) \geq lThr \\ 255 & \text{otherwise} \end{cases} \tag{4.16}$$

where pixel (x, y) denotes a gray-level and $\acute{s}$ represents the similarity matrix after thresholding. The pixel values 0 and 255 represent the darkest and the brightest pixel values for an $8 - bit$ gray level color palette, respectively. Thus, the resulting similarity matrix contains black diagonal stripes over the white background.

The next step after thresholding is to compute attributes (length, starting and ending locations) of each candidate pattern from the similarity matrix $\acute{s}$ as described in Figure 4.6. These attributes are stored in a structure called *element* for each candidate pattern and put into an array denoted by A.

As previously mentioned, a similarity matrix may contain many diagonal stripes in different sizes representing distinct repetitive pattern groups. The longest repetitive pattern is perceived as the expressive summary; therefore there is a need to group patterns in similar length into the same cluster and then find the pattern group containing patterns in longer length. For this purpose, *k-means clustering* algorithm [50] which is one of the simplest unsupervised learning algorithms is used.

Given the number of clusters (k), candidate elements (N) are clustered into k clusters with respect to their lengths. Initially, the first k elements are taken as single-element clusters, and then each of the remaining elements ($N - k$) is assigned to the cluster with nearest centroid. When all elements have been assigned, the centroids are recalculated. Afterwards, for each element in A, the distance from the centroid of each of the clusters is computed. The assignment of the element is changed if it is not currently in the cluster with the closest centroid. This process is continued until the centroids no longer move. We tried $k = 3 - 6$ in our experiments and take $k = 5$ in the clustering stage.

4.3.6 Experimental Results for the Retrieval Approach

In this section, we discuss the results of two kinds of QBE tests that have been performed on real news broadcasts from TRECVID 2003 ABC World News Tonight and CNN headline news and domain related sound effects from the Internet [46]. In the first test, 35 different news broadcasts, each with duration of nearly 30 minutes are used as the search space. Besides, 457 audio clips gathered from the mentioned news broadcasts are used as the sample set. More specifically, 255 speech, 190 music, and 144 environmental sound audio clips in variable length are included in the sample set. For the second test, 100 different domain related sound effects (e.g., alarm, car, gun shot, machines, wind, storm, applause, laughter) in different categories are collected.

In these experiments, we tried to measure retrieval performance of QBE technique which is directly affected by both effectiveness of ASF feature and accuracy of segmentation results. Each sample audio clip is submitted to the system one by one and five most similar audio clips in the search space are requested from the system. If the requested audio clip is retrieved at the first position in the result list, then retrieval accuracy for that file is measured as 100%. If the requested audio clip is retrieved at the second position in the result list, then retrieval accuracy for that file is measured as 80%, and so forth. Boundary accuracy (BA) for each correctly retrieved audio clip is also measured by dividing duration of overlapping part of the sample audio and the retrieved audio to the duration of the sample audio. Let sample audio interval be denoted by I_1 and retrieved audio interval is denoted by I_2, then boundary accuracy is calculated as follows:

$$BA = \frac{Length\,(I_1 \cap I_2)}{Length\,(I_1)} \tag{4.17}$$

The aim of the first test is measuring performance of retrieval of an audio clip exactly same as the sample audio. Therefore, we used audio data of both search space and the sample set gathered from the search space as they are. Dissimilarly, in the second test, sound effects in the sample set were embedded as a background into audio data of selected news broadcasts in separate positions. Experimental results are displayed in Table 4.3. Average retrieval accuracy is denoted by *RA* and average boundary accuracy is denoted by *BA* in this table.

From Table 4.3, it is obviously seen that average retrieval accuracy and boundary accuracy rates are the highest ones in music samples. Actually these results could be anticipated towards study of Sert et al., [32]. In that study, it is reported that the Mpeg-7 ASF descriptor captures the characteristics of musical audio better than speech signals due to the utilized logarithmic frequency scale in calculations. That is, most of musical audio make use of higher frequencies (e.g., 44.1 kHz CD quality) whereas speech signals generally consist of lower frequencies (e.g., below 1 kHz) relative to musical audio. A descriptor that uses a linear scale for lower frequencies (e.g., MFCC feature set) would perform slightly better performance in recognition of speech signals.

Table 4.3: Retrieval Results of Sample Sets

Audio Type	Number of Samples	RA	BA
First Test			
Speech	255	62%	94%
Music	190	84%	95%
Environmental Sound	144	73%	90%
Second Test			
Sound Effect	100	81%	97%

CHAPTER 5

IMPLEMENTATION OF THE PROPOSED SYSTEM

We have implemented a complete audio management and retrieval system for news broadcasts audio by using Java programming language. In this system there is no database, all required data is stored in files. The general overview of the developed system is shown in Figure 5.1. The proposed system is composed of mainly six modules which are Feature Extraction, Train&Test, Classification&Segmentation, Indexing, Query, and User Interface.

The Feature Extraction module extracts various Mpeg-7 audio low-level descriptors from the audio source to obtain description of its content. The extracted features are Audio Power (AP), Audio Spectrum Centroid (ASC), Audio Spectrum Spread (ASS), Audio Spectrum Flatness (ASF) and Audio Spectrum Projection (ASP). Train&Test module prepares classifiers for the classification and segmentation processes. In the proposed system, the employed classifiers are Support Vector Machine (SVM) and Hidden Markov Model (HMM). Classification&Segmentation module handles audio classification and segmentation operations by the use of required Mpeg-7 XML (eXtended Mark-up Language) files and previously trained SVM or (and) HMM models. Query module satisfies user queries to retrieve content of the audio. The supported query types are temporal queries, temporal relationship queries, similarity search queries (QBE) and expressive summary search queries (ESS). Indexing module indexes the segmentation result data of an audio and provides input to the Query module for temporal and temporal relationship queries. Lastly, the User Interface module provides users a means for data entrance and requests. A comprehensive description of the proposed system modules are given in the following sections.

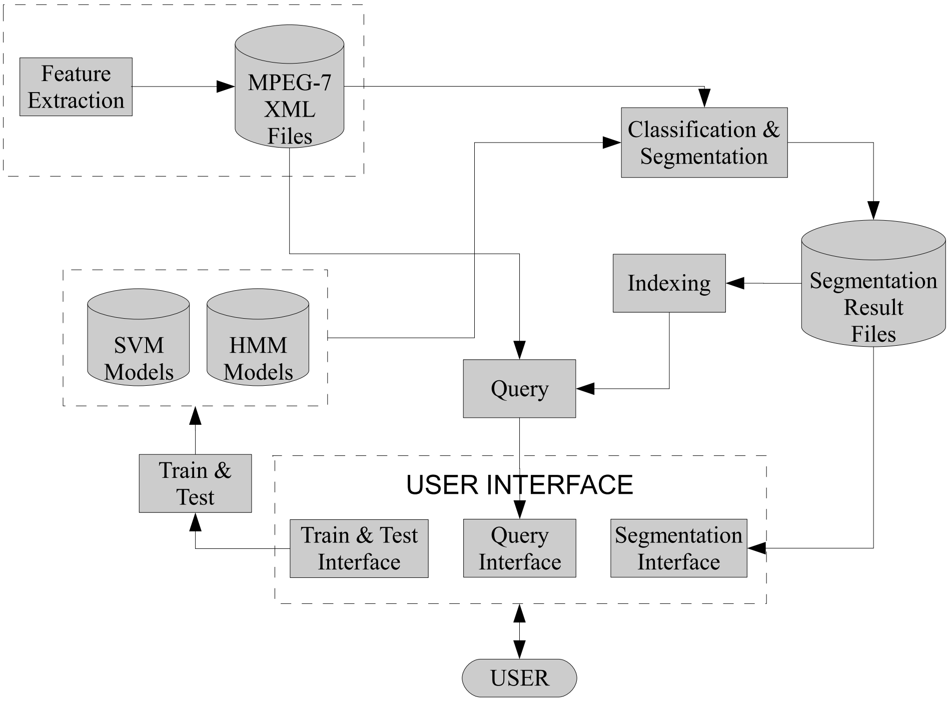
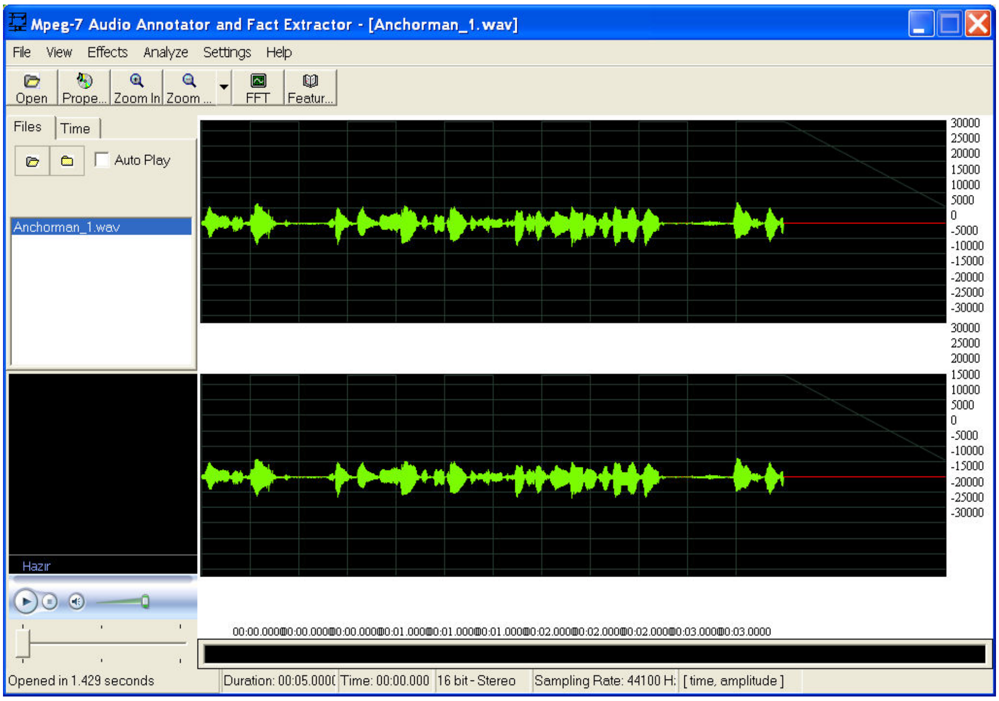


Figure 5.1: System Architecture of the Proposed System

5.1 Feature Extraction Module

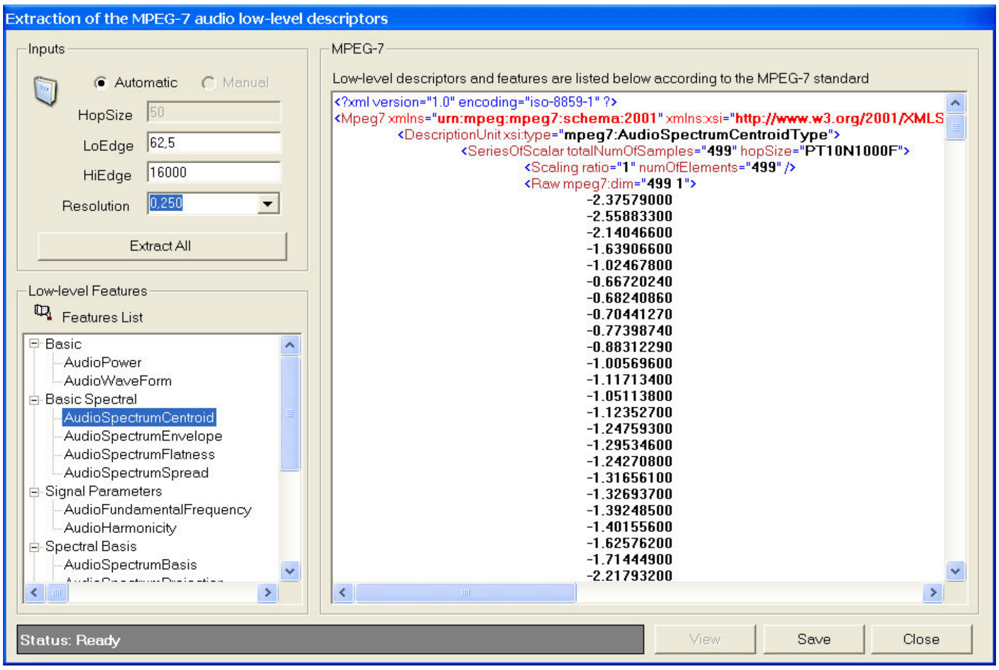
As previously mentioned, we employed Mpeg-7 Audio Annotator and Fact Extractor tool [36] to extract the required Mpeg-7 features of audio data. User extracts Mpeg-7 features of the input audio via this tool and stores in XML documents [51]. The usage of this tool is demonstrated by the following screenshots and a sample for an Mpeg-7 XML document is shown in Appendix B as Figure B.1.

User chooses the input audio file from the main menu as shown in Figure 5.2, and presses the *Features* button to extract the required Mpeg-7 features. Extracted feature values are shown as in Figure 5.3 and now user has option to store this data in an XML file.



Opened in 1.429 seconds | Duration: 00:05.000(| Time: 00:00.000 | 16 bit - Stereo | Sampling Rate: 44100 H: | [time, amplitude]

Figure 5.2: Mpeg-7 Audio Annotator and Fact Extractor Tool:File Selection



● Automatic ○ Manual

HopSize 50

LoEdge 62,5

HiEdge 16000

Resolution 0,250

- Basic
  - AudioPower
  - AudioWaveForm
- Basic Spectral
  - AudioSpectrumCentroid
  - AudioSpectrumEnvelope
  - AudioSpectrumFlatness
  - AudioSpectrumSpread
- Signal Parameters
  - AudioFundamentalFrequency
  - AudioHarmonicity
- Spectral Basis
  - AudioSpectrumBasis

Low-level descriptors and features are listed below according to the MPEG-7 standard

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001" xmlns:xsi="http://www.w3.org/2001/XMLS
        <DescriptionUnit xsi:type="mpeg7:AudioSpectrumCentroidType">
                <SeriesOfScalar totalNumOfSamples="499" hopSize="PT10N1000F">
                        <Scaling ratio="1" numOfElements="499" />
                        <Raw mpeg7:dim="499 1">
                                -2.37579000
                                -2.55883300
                                -2.14046600
                                -1.63906600
                                -1.02467800
                                -0.66720240
                                -0.68240860
                                -0.70441270
                                -0.77398740
                                -0.88312290
                                -1.00569600
                                -1.11713400
                                -1.05113800
                                -1.12352700
                                -1.24759300
                                -1.29534600
                                -1.24270800
                                -1.31656100
                                -1.32693700
                                -1.39248500
                                -1.40155600
                                -1.62576200
                                -1.71444900
                                -2.21793200
```

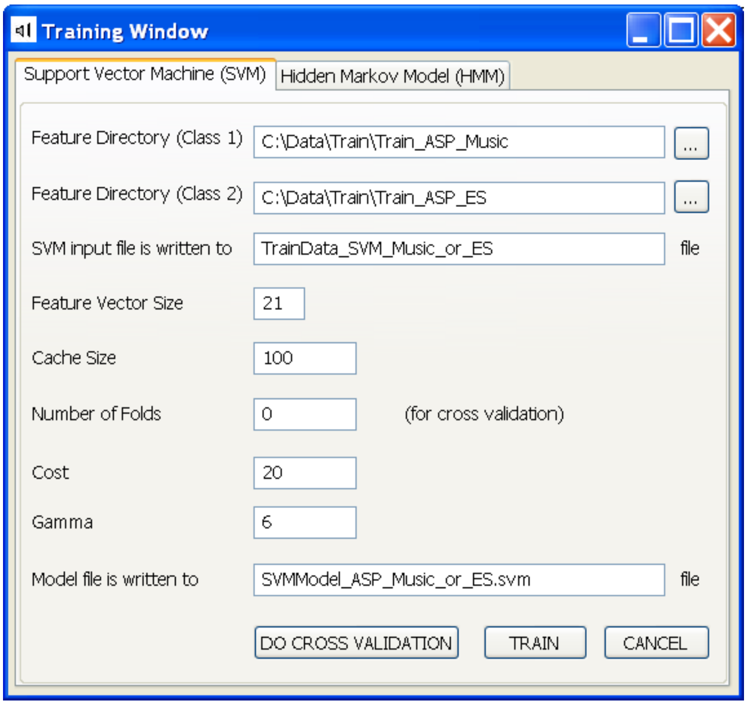
Status: Ready

Figure 5.3: Mpeg-7 Audio Annotator and Fact Extractor Tool:Feature Extraction

5.2 Train&Test Module

As mentioned in section 4.2, Support Vector Machine (SVM) and Hidden Markov Model (HMM) are two classification methods employed in our thesis work. First, we trained SVM and HMM models with our training data, and then we tested their classification accuracy with our test data by utilizing Train&Test module. Core of this module consists of *LIBSVM* [52] package and *JAHMM* [53] package employed for SVM and HMM classification respectively.

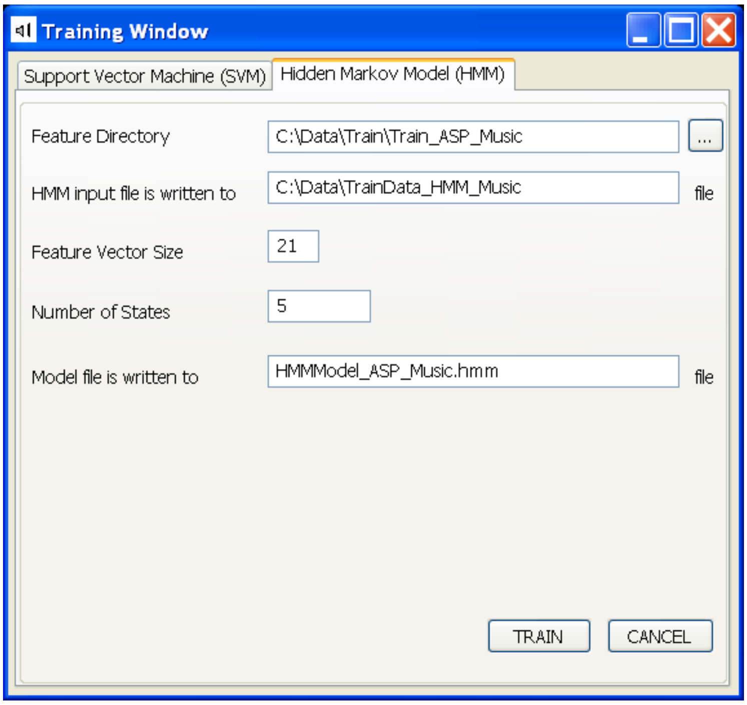
User may change the training dataset and retrain all the models or may change the test dataset and make experiments on them by using these modules. Our system supplies Train&Test interfaces for user to enter parameters for training and testing as shown in Figures 5.4–5.7.



Feature Directory (Class 1)	C:\Data\Train\Train_ASP_Music	...
Feature Directory (Class 2)	C:\Data\Train\Train_ASP_ES	...
SVM input file is written to	TrainData_SVM_Music_or_ES	file
Feature Vector Size	21	
Cache Size	100	
Number of Folds	0	(for cross validation)
Cost	20	
Gamma	6	
Model file is written to	SVMModel_ASP_Music_or_ES.svm	file

Figure 5.4: Training Window for SVM Classification

Figure 5.4 shows the training window for SVM classification. Via this window, user may train SVM models as well as do cross validation to select optimum *cost* and *gamma* parameters for the training set. In SVM training, two training data sets are needed for each binary classification. For instance, to model an SVM classifier for music/environmental sound separation, user enters one directory name for the music training data set and one directory name for the



Feature Directory: C:\Data\Train\Train_ASP_Music

HMM input file is written to: C:\Data\TrainData_HMM_Music file

Feature Vector Size: 21

Number of States: 5

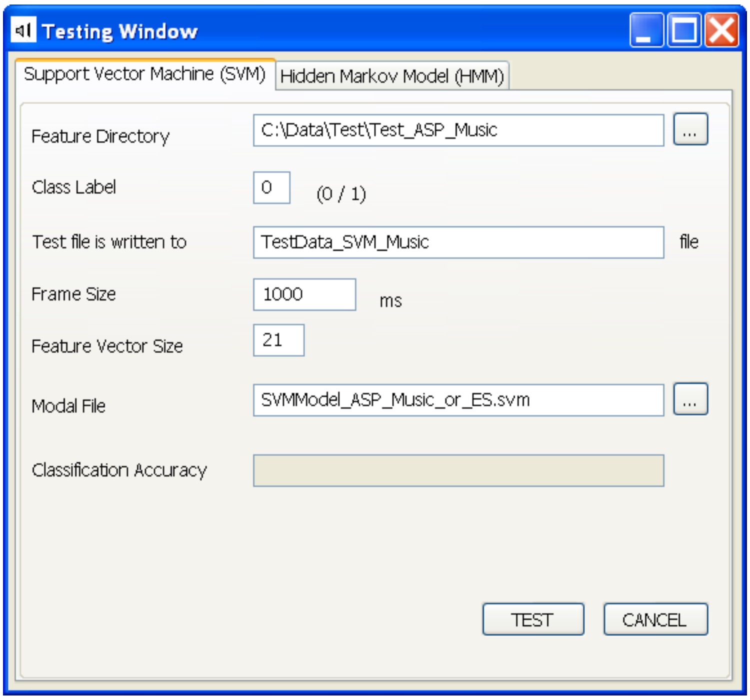
Model file is written to: HMMModel_ASP_Music.hmm file

Figure 5.5: Training Window for HMM Classification

environmental sound training data set. Data sets are composed of Mpeg-7 XML files organized according to their descriptor types, e.g., ASF directory for XML files of Mpeg-7 ASF descriptors. *LIBSVM* package requires the input file in a specific format, so feature values are written into the specified file accordingly. *Feature vector size* is the vector size of the feature set, *cache size* is the size of the kernel cache specified in megabytes, *number of folds* is the number of folds that the data is separated in cross validation, *cost* is the cost of constraints violation and one of the parameters of the kernel function, and *gamma* is the other user specified parameter of the kernel function.

Training window for HMM classification is shown in Figure 5.5. Since an HMM model is trained for each class, user enters the directory name of the training set for the modeled class. Likewise, *JAHMM* package requires the input file in a specific format, and so feature values are written into the specified file accordingly. *Feature vector size* is the vector size of the feature set and *number of states* is the state count in HMM model.

Testing window for SVM and HMM classification is shown in Figure 5.6 and Figure 5.7 respectively. User may test only one class type of data set at a time. User enters directory



Feature Directory C:\Data\Test\Test_ASP_Music

Class Label 0 (0 / 1)

Test file is written to TestData_SVM_Music file

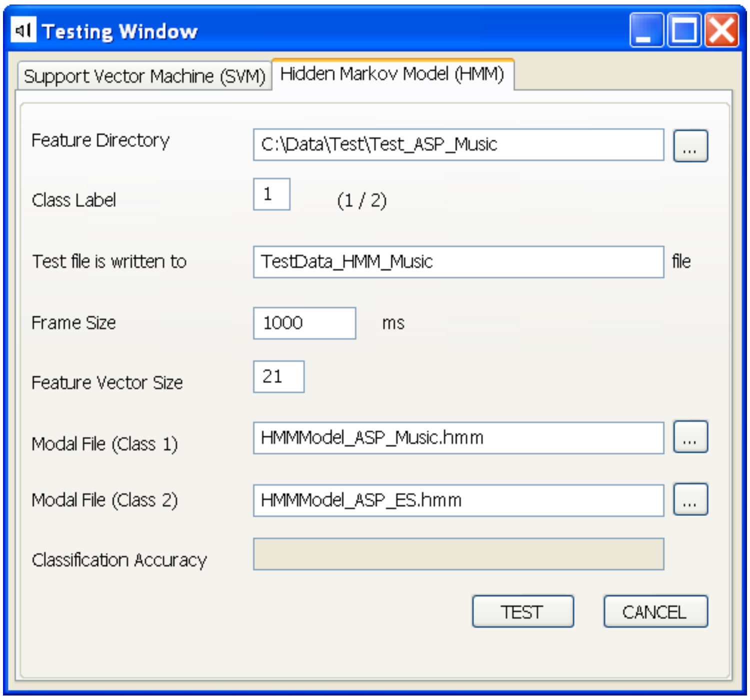
Frame Size 1000 ms

Feature Vector Size 21

Modal File SVMModel_ASP_Music_or_ES.svm

Classification Accuracy

Figure 5.6: Testing Window for SVM Classification



Feature Directory C:\Data\Test\Test_ASP_Music

Class Label 1 (1 / 2)

Test file is written to TestData_HMM_Music file

Frame Size 1000 ms

Feature Vector Size 21

Modal File (Class 1) HMMModel_ASP_Music.hmm

Modal File (Class 2) HMMModel_ASP_ES.hmm

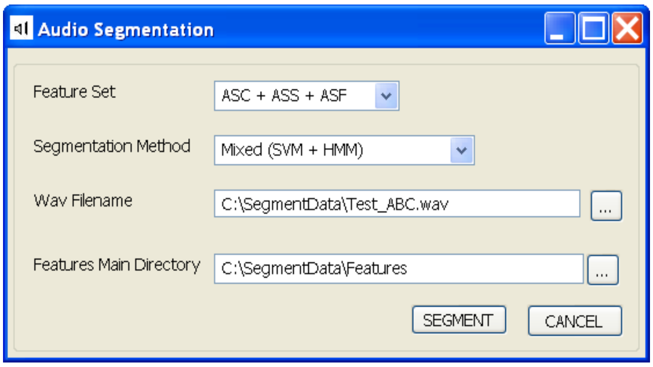
Classification Accuracy

Figure 5.7: Testing Window for HMM Classification

name as well as class label of the test data set. As previously mentioned, read feature values are written into the specified file accordingly. Audio files in the test data set are divided into clips and these clips are classified individually. In addition, length of these clips is specified by the *frame size* parameter. User enters one model file name for SVM classification and two model file names for HMM classification. After the testing process, system retrieves the classification accuracy rate to the user.

5.3 Classification&Segmentation Module

In our system, user has option to classify and segment an audio stream into predefined classes and visualize the segmentation result via a visual interface. Classification&Segmentation module is responsible for the audio classification and segmentation process which is explained in section 4.1. and 4.2. Segmentation window is shown in Figure 5.8.



Audio Segmentation

Feature Set: ASC + ASS + ASF

Segmentation Method: Mixed (SVM + HMM)

Wav Filename: C:\SegmentData\Test_ABC.wav

Features Main Directory: C:\SegmentData\Features

Figure 5.8: Segmentation Window

User has two options to select the feature set. Those are Audio Spectrum Projection (ASP) coefficients and ASCSF feature set consisting of Audio Spectrum Centroid (ASC), Audio Spectrum Spread (ASS), and Audio Spectrum Flatness (ASF) features.

Hidden Markov Model (HMM) and Support Vector Machine (SVM) are two classification methods those can be employed in audio classification. User also has option to use classification methods according to their classification accuracy rates, i.e., HMM for non-speech and with-speech classification, SVM for music and environmental sound classification, SVM for pure-speech and mixed-speech classification, HMM for speech over environmental sound and speech over music classification.

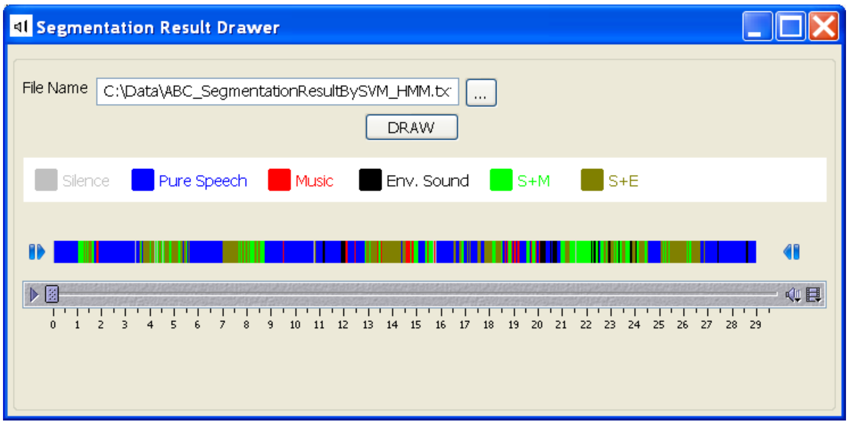
Audio file and main directory of the Mpeg-7 feature files of the selected audio file should be selected. After providing all necessary parameters, Classification&Segmentation module performs segmentation and writes the segmentation result into a TXT file in a simple format as described in the followings.

In the first line, full path of the audio file is written. The format of the remaining lines is:

```
<label> <accuracy> <start time> <end time>
.
.
.
```

Each line contains an instance and is ended by a `'\n'` character. `<label>` is a string indicating the class label. `<accuracy>` is the accuracy rate of the classified segment after the smoothing phase. `<start time>` and `<end time>` are starting and end time of the segment respectively.

The “Segmentation Result Drawer” window, shown in Figure 5.9, provides user to view the segmentation result in a colored timeline. User may listen to the audio and switch between the segments with one button click.



File Name C:\Data\ABC_SegmentationResultBySVM_HMM.tx

Figure 5.9: Segmentation Result Drawer Window

5.4 Indexing Module

The outcome of the audio segmentation algorithm is a sequence of audio segments ordered by their starting time (or end time). When user queries on the segmentation result, the system may have performed a sequential search, starting from the first segment to the last segment. However this method does not perform well on queries having constraints on class type or (and) duration. Concerning that, to reduce the retrieval time we propose a simple indexing technique that does not require extensive analysis. A general overview of the proposed index structure is shown in Figure 5.10.

We present an index structure based on hash table, namely "2LH-index". *2L* denotes "two-level" and *H* denotes "hash". This index structure can index segmentation result data, and support temporal and temporal relationship queries in point or in range type.

Hash table [54] is a high performance data structure used for data look up by a particular aspect of that data, called a *key*. The idea behind the hash table is to process the key with a function that returns a *hash value*; that hash value then determines where in the data structure the record will (or probably will) be stored. When the same key is used to search for the stored information, the same hash value is generated, leading the searcher either directly to the location of the information or to the best place to start looking.

The basic general algorithm for a look up is simply

$$index = f\,(key, arrayLength) \tag{5.1}$$

where:

- f is the hash function that returns an index within the array
- *index* is an index into the array
- *key* is the key data
- *arrayLength* is the length of the array

In our index structure, audio *class type* is the first level hash key. That means the first level hash table contains one slot for each class type (silence, pure speech, music, env. sound, speech over music, speech over env. sound), totally six slots. Each slot in the first level hash

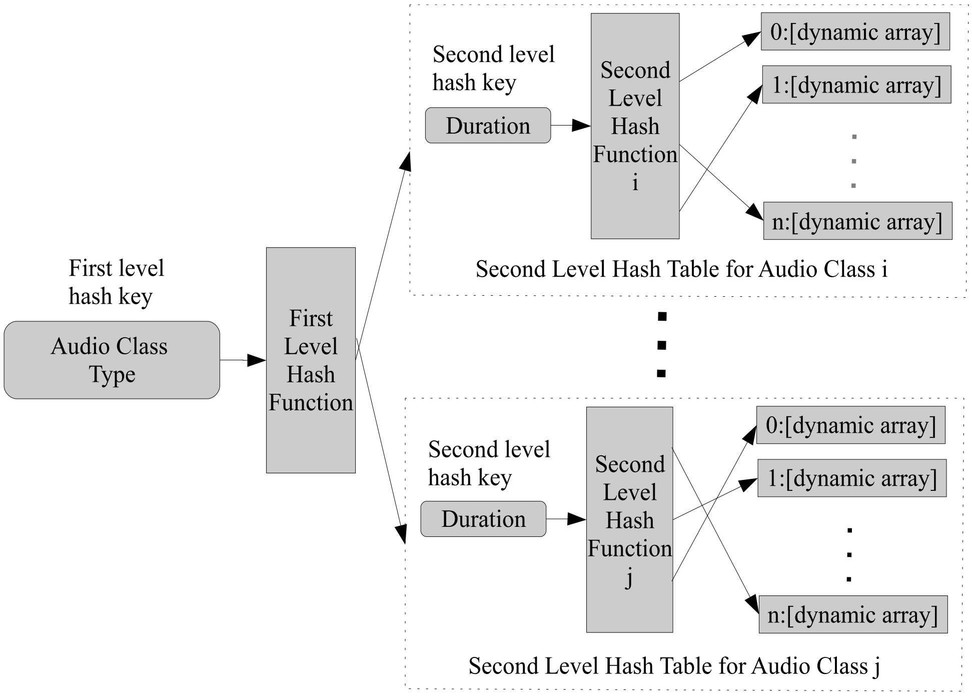


Figure 5.10: General Overview of the Proposed Index Structure

table is another hash table, which we call second level hash table. Key of the second level hash table is the *duration* data of the segment and each slot in the second level hash table is a dynamic array (called *vector* in java). This array is composed of temporally ordered segment data which are stored in structures. Structure storing segment data is called “bucket” and defined as the following:

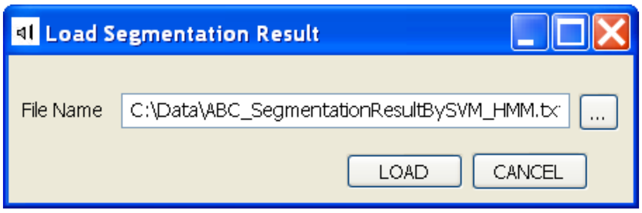
```
struct bucket
{
  tint classIDKey;
  tint durationKey;
  tint bucketNo;
  double s_accuracy;
  unsigned int startingtime;
  unsigned int endtime;
  int prevClassIDKey;
  int prevDurationKey;
  int prevBucketNo;
  int nextClassIDKey;
  int nextDurationKey;
  int nextBucketNo;
}
```

In the structure named “bucket”, *classIDKey* is acoustic class type of the segment, *durationKey* is duration of the segment in seconds, *bucketNo* is order of the bucket in the array, *s_accuracy* is smoothing accuracy (see section 4.2.3), *startingtime* is starting time of the segment, and *endtime* is end time of the segment. For temporal relationship queries like “Find the intervals in which anchor speaking after the transition music”, previous and next segments of the current segment are accessed directly in $O(1)$ time by using their classIDKey, durationKey, and bucketNo. *prevClassIDKey*, *prevDurationKey*, and *prevBucketNo* are classIDKey, durationKey, and bucketNo of the previous segment respectively. Additionally, *nextClassIDKey*, *nextDurationKey*, and *nextBucketNo* are classIDKey, durationKey, and bucketNo of the subsequent segment respectively.

For instance, when user requests “music segments longer than five seconds”, “music” is used as the first level key for the first-level hash table and hash table in the “music” slot is retrieved. Since the requested query is in range type, initially the indexes of the retrieved second-level hash table are sorted, and then only the slots (dynamic arrays) having keys equal or bigger than “five” are retrieved.

Choosing an alternative file from preloaded file repository is supported by the proposed system. User may load segmentation result files to the system via “Load Segmentation Result” window, as shown in Figure 5.11. Thereafter, system builds an index structure for that file and stores it to be queried.

As we mentioned earlier, the system supports queries of type QBE and “Expressive Summary Search” (ESS) also. However, these types of queries perform an exhaustive search and do not depend on the segmentation results directly. Therefore, the system does not use any indexing for these types of queries.



File Name C:\Data\ABC_SegmentationResultBySVM_HMM.tx

Figure 5.11: Load Segmentation Result Window

5.5 Query and User Interface Module

Users query the audio data via query interfaces supplied by the system and Query module computes the results by taking the index structure or directly Mpeg-7 feature files as input. The system clears the query result list for new querying and plays the selected temporal information.

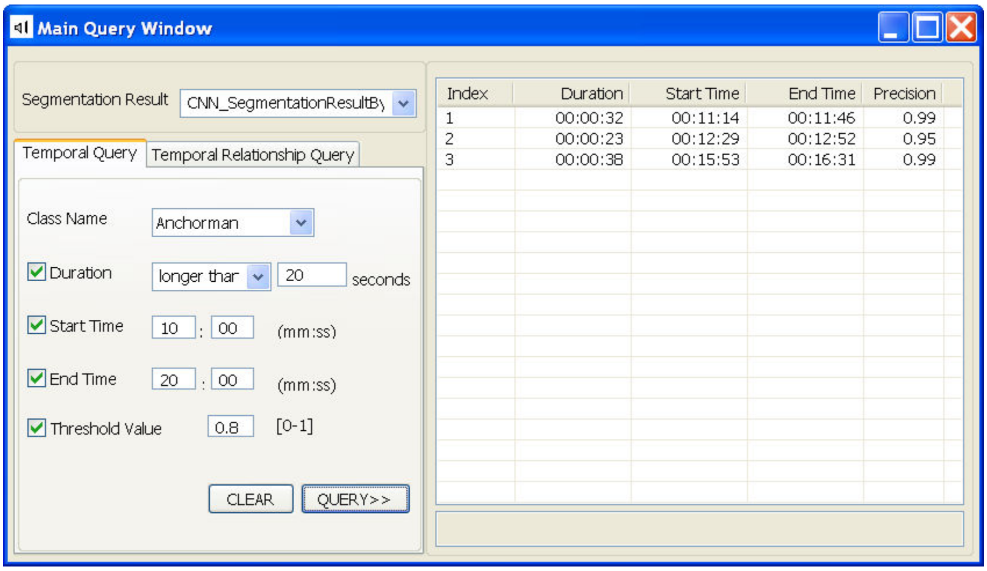
The system supports the query types given below:

1. Temporal Queries
2. Temporal Relationship Queries
3. Similarity Search Queries (QBE)
4. Expressive Summary Search Queries (ESS)

The system has query interfaces, but no specific query language is defined. The user converts his or her question to the structure of information that can be entered to the interface. Indexing module provides input to the Query module for temporal and temporal relationship queries. QBE and ESS queries do not depend on the segmentation results directly, so the results of these queries are calculated solely based on Mpeg-7 feature files.

5.5.1 Temporal Queries

The system has a temporal query interface shown in Figure 5.12. In this interface user can perform the following queries:



Segmentation Result: CNN_SegmentationResultBy

Class Name: Anchorman

☑ Duration: longer thar 20 seconds

☑ Start Time: 10 : 00 (mm:ss)

☑ End Time: 20 : 00 (mm:ss)

☑ Threshold Value: 0.8 [0-1]

Index	Duration	Start Time	End Time	Precision
1	00:00:32	00:11:14	00:11:46	0.99
2	00:00:23	00:12:29	00:12:52	0.95
3	00:00:38	00:15:53	00:16:31	0.99

Figure 5.12: Temporal Query Interface

1. Given a fuzzy class type, find its occurrences. For example: “Find me all possible *commercial* segments in this news video.”

2. Given a fuzzy class type and its duration, find the corresponding segments. For example: “Find me all possible *sports* segments of length (longer than/shorter than) 15 seconds.”

3. Given a fuzzy class type and its starting time or (and) end time, find the corresponding segments. For example: “Find me all possible *anchor* speaking segments between the minutes 10 and 20.”

4. Given a fuzzy class type and its threshold value, find its occurrences. For example: “Find me all possible *weatherforecast* segments with a threshold value 0.6.”

5. Any combinations of the first four queries. For example: “Find me all possible *commercial* segments of length 30 seconds between the minutes 10 and 20 with a threshold value 0.8.”

The index structure built for the selected audio file is adequate to answer temporal queries. Initially, if the user states the threshold value for the selected fuzzy class set, members (acoustic class types) having membership degree higher than the given threshold value are retrieved.

If not, all members are retrieved. These acoustic class types are used as the first level hash keys. For each member, the query is solved individually and the results are combined. For example, for the query “Find me all possible *anchor* segments with a threshold value 0.6.”, the system retrieves only the “pure speech” class type and uses it as the first-level hash key. Other members, “speech over music” and “speech over environmental sound”, are not retrieved as their membership degrees are lower than 0.8.

If the duration value is set, it is used as the second-level hash key and the related slot is retrieved. If the duration option is in range type (i.e., shorter than or longer than), multiple slots may be retrieved.

If there is not any constraint on the temporal information, all buckets in the slot are retrieved; else the buckets are examined regarding their starting time, end time, or both. Since the buckets in the array are sorted according to their temporal data, search on this list is not exhaustive.

5.5.2 Temporal Relationship Queries

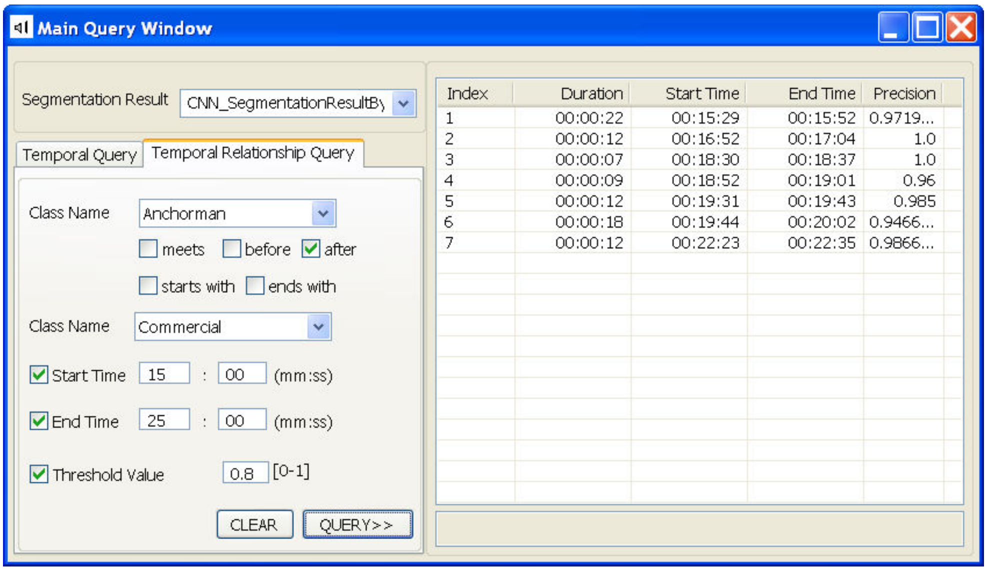
Temporal operators implemented in our system (i.e., meets, before, after, starts with, ends with) are discussed in Section 4.3.3. The system has a temporal relationship query interface shown in Figure 5.13. In this interface user can perform the following queries:

1. Given a fuzzy class type and its relation operator (*starts with* or *ends with*) with its adjacent segment, find the intervals. For example: “Find me the intervals *starting with transition* music.”

2. Given two fuzzy class types and their relation operator (*meets*, *before*, or *after*), find the intervals. For example: “Find me the intervals when *anchor* speaking *meets* the *transition* music in this news video.”

3. Given two fuzzy class types, their relation operator and starting time or (and) end time, find the corresponding intervals. For example: “Find me the intervals when *reporter* speaking *before* the *venue-sound* heard between the minutes 10 and 20.”

4. Given a fuzzy class type, its relation operator and threshold value, find the corresponding intervals. For example: “Find me the intervals *starting with sports* with a threshold



Segmentation Result: CNN_SegmentationResultBy

Class Name: Anchorman

☐ meets ☐ before ☑ after

☐ starts with ☐ ends with

Class Name: Commercial

☑ Start Time 15 : 00 (mm:ss)

☑ End Time 25 : 00 (mm:ss)

☑ Threshold Value 0.8 [0-1]

Index	Duration	Start Time	End Time	Precision
1	00:00:22	00:15:29	00:15:52	0.9719...
2	00:00:12	00:16:52	00:17:04	1.0
3	00:00:07	00:18:30	00:18:37	1.0
4	00:00:09	00:18:52	00:19:01	0.96
5	00:00:12	00:19:31	00:19:43	0.985
6	00:00:18	00:19:44	00:20:02	0.9466...
7	00:00:12	00:22:23	00:22:35	0.9866...

Figure 5.13: Temporal Relationship Query Interface

value 0.6.”

5. Any combinations of first four queries. For example: “Find me the intervals of length 30 seconds between the minutes 10 and 20 when *anchor* speaking with a threshold value 0.8 *meets* the *transition* music with a threshold value 0.8.”

Temporal relationship queries also use solely built index structure for the selected file. Additionally, in this type of queries search of adjacent segments (previous or next) of the current segment is also required. Retrieval of the queried intervals is performed according to the following algorithm:

1. If the relation operator is “starts with” or “ends with”, retrieve the segments of the given fuzzy class set as explained in the temporal queries. Otherwise, retrieve the segments of the first of the class sets as explained in the temporal queries.

2. Access to the temporally adjacent segments of the retrieved segments, and test whether the given constrains are satisfied.

   (a) If the relation operator is “starts with”, access to the next segment by using *nextClassIDKey*, *nextDurationKey*, and *nextBucketNo* of the current segment and test if the constrains (start time, end time, threshold value) are satisfied.

(b) If the relation operator is “ends with”, access to the previous segment by using *prevClassIDKey*, *prevDurationKey*, and *prevBucketNo* of the current segment and test if the constrains (start time, end time, threshold value) are satisfied.

(c) If the relation operator is “meets”, access to the next segment by using *nextClassIDKey*, *nextDurationKey*, and *nextBucketNo* of the current segment and test if the constrains (second fuzzy class set, start time, end time, threshold value) are satisfied.

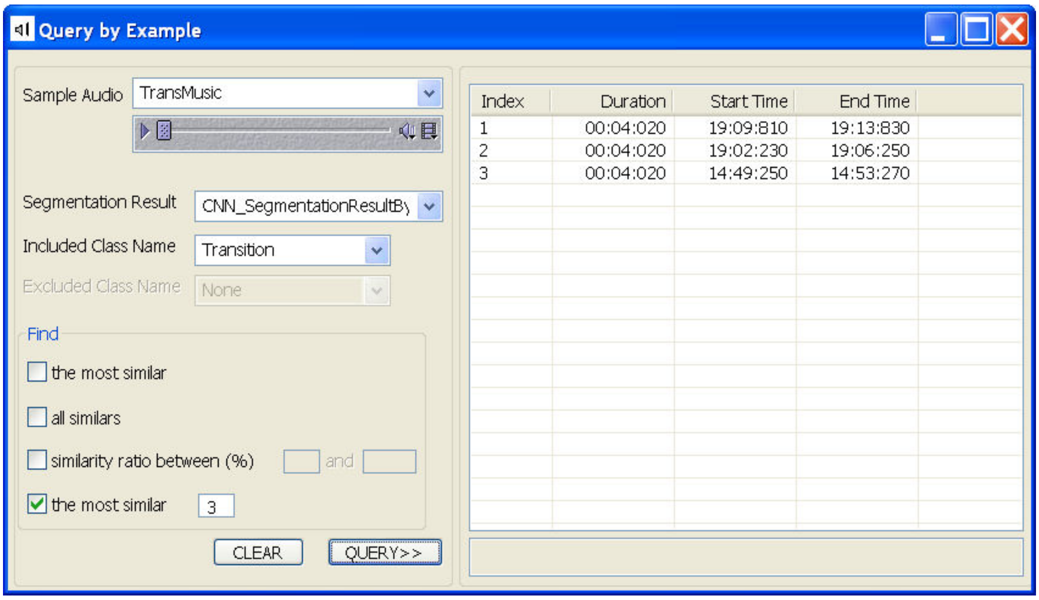
(d) If the relation operator is “before”, access to the succeeding segments by using *nextClassIDKey*, *nextDurationKey*, and *nextBucketNo* of the current segment until the “end time” constrain is violated or the constrains “second fuzzy class set” and “threshold value” are satisfied.

(e) If the relation operator is “after”, access to the preceding segments by using *prevClassIDKey*, *prevDurationKey*, and *prevBucketNo* of the current segment until the “start time” constrain is violated or the constrains “second fuzzy class set” or “threshold value” are satisfied.

5.5.3 Similarity Search Queries

Given an audio sample, the Query module performs the similarity search and retrieves the results according to the selected search option. The system has a “Query by Example” interface shown in Figure 5.14. In this interface user can perform the following queries:

1. Given an audio sample and the search option, find the similar audio interval(s) in the selected audio stream. For instance:
   - “Find me the explosion sound most similar to the given audio sample.”
   - “Find me the gun shot sounds similar to the given audio sample with a degree of 80% to 100%.”
   - “Find me three traffic sounds most similar to the given audio sample.”
2. Given an audio sample, the search option, and the search space; find the similar audio interval(s) in the selected audio stream. For example: “Only search for the *venue sound*s and find me the explosion sound most similar to the given audio sample.”



Query by Example

Sample Audio: TransMusic

Segmentation Result: CNN_SegmentationResultBy

Included Class Name: Transition

Excluded Class Name: None

Find

□ the most similar

□ all similars

□ similarity ratio between (%) □ and □

☑ the most similar 3

Index	Duration	Start Time	End Time
1	00:04:020	19:09:810	19:13:830
2	00:04:020	19:02:230	19:06:250
3	00:04:020	14:49:250	14:53:270

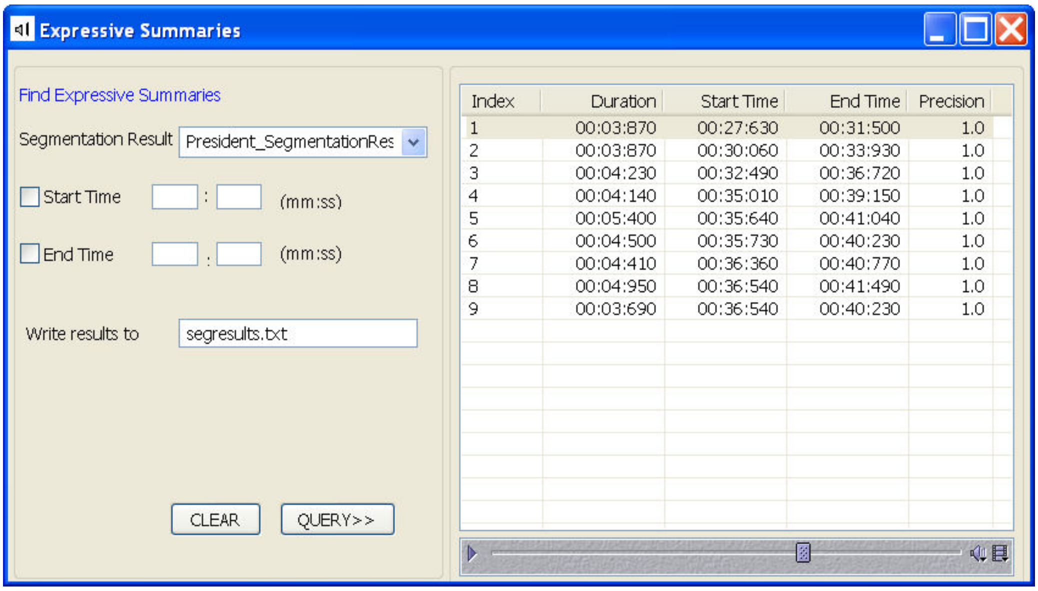
Figure 5.14: Query by Example Interface

3. Given an audio sample, the search option, and the excluded search space; find the similar audio interval(s) in the selected audio stream. For example: "Search all audio segments but *commercial*s, find me all audio intervals similar to the man recorded speech."

User selects the sample audio through the "Query by Example" interface. The system provides domain related sound effects to be queried. Besides, user may also find or record sample audio files and add them to the "Sound Effects" directory.

5.5.4 Expressive Summary Search Queries

In Section 4.3.5, we explain the expressive summary search algorithm in detail. We perform the expressive summary search on with-speech audio segments due to the fact that only with-speech segments are meaningful for the summary search in news domain. This means, we discard non-speech segments (silence, music, environmental sound) and combine each consecutive with-speech segments (pure speech, speech over music, speech over environmental sound) into one with-speech block, and then perform search on these blocks separately. By this way, we reduce the search space and divide the news audio into story blocks.



Segmentation Result: President_SegmentationRes

□ Start Time : (mm:ss)

□ End Time : (mm:ss)

Write results to: segresults.txt

Index	Duration	Start Time	End Time	Precision
1	00:03:870	00:27:630	00:31:500	1.0
2	00:03:870	00:30:060	00:33:930	1.0
3	00:04:230	00:32:490	00:36:720	1.0
4	00:04:140	00:35:010	00:39:150	1.0
5	00:05:400	00:35:640	00:41:040	1.0
6	00:04:500	00:35:730	00:40:230	1.0
7	00:04:410	00:36:360	00:40:770	1.0
8	00:04:950	00:36:540	00:41:490	1.0
9	00:03:690	00:36:540	00:40:230	1.0

Figure 5.15: Expressive Summaries Interface

The system has “Expressive Summaries” interface shown in Figure 5.15. In this interface user can perform the following queries:

1. Find the expressive summary intervals in the selected audio stream. For example: “Find me all possible key concepts in the selected video.”
2. Given starting time or (and) end time, find the expressive summary intervals in the selected audio stream. For example: “Find me all possible key concepts in the selected video in its first 10 minutes.”

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, we propose a complete content-based retrieval and management system for news broadcasts. To evaluate the power of the proposed system, we have concentrated on generating a general solution set to classify, segment, analyze, and retrieve audio data from news videos. Accordingly we have worked on real news broadcasts from TRECVID 2003 those can have subjects of any kind. Besides, we designed the system to be consisting of different stages each of which can be replaced by alternative methods without affecting other stages.

In the classification and segmentation stage, we introduce new mixed type classes (e.g., speech over music, speech over environmental sound) which are suitable for multimedia indexing and retrieval. Also, in this stage we evaluate and analyze the recognition accuracy of different classifiers and different Mpeg-7 low level audio descriptors. Our experimental results show that, ASCSF feature set can capture the characteristic of mixed type audio data considerably better than ASP feature set in news domain. In all binary classifications of which results are shown in Appendix A from Table A.1 to Table A.4, SVM-based and HMM-based classifiers using ASCSF feature set yield approximate accuracy rates to each other. Thus apart from HMM, the widespread applied method for MPEG-7 based sound classification, SVM based on MPEG-7 features can also be employed in general sound classification and segmentation tools if complexity issue is undervalued. The kernel parameters of SVM have crucial effects on the performance of it. However, we must keep in mind that selection of these parameters is still a research issue, which in this work is practiced based on experiments.

The proposed system also gives users an opportunity for flexible querying of audio data semantically by providing various alternative ways via query interfaces. Domain based fuzzy classes (i.e., anchor, commercial, reporter, sports, transition, weatherforecast, and venue-sound) are included into the retrieval stage and so, information extracted from experts are provided to respond to user queries. The proposed retrieval method is scalable and extensible since the model can be altered by only changing the defined semantic classes and their membership degrees without requiring any retraining. The performance of the proposed QBE technique based on acoustic segmentation of audio stream was tested by conducting two kinds of experiments. The results obtained from the first experiment show that ASF feature in MPEG-7 standard performs better in music audio samples compared to other kinds of audio samples. Additionally, results from the second experiment clearly show that the proposed retrieval method is robust under different conditions.

We use the Mpeg-7 Audio Annotator and Fact Extractor Tool [36] while extracting the required Mpeg-7 features of audio data used in our experiments. By this means, we do not need to go into detail about the way in which the low level descriptors are extracted. Furthermore, we employed *LIBSVM* library [52] for SVM classification and *JAHMM* library [53] for HMM classification. By using these tools, we have saved a lot of time in the implementation phase.

For further studies, an automatic Mpeg-7 feature extraction tool may be embedded to this solution. In the current solution, first the Mpeg-7 XML documents of the audio file are created and stored via the Mpeg-7 Audio Annotator and Fact Extractor Tool [36] interactively. Then, the system performs operations on these XML documents. However, by embedding a feature extraction tool, the system may be converted to a system in which the user uploads an audio file remotely and queries the data without any manual operation.

A database specifically designed for the storage and retrieval of XML formatted documents may be used to store and retrieve the required Mpeg-7 documents. Oracle Berkeley DB XML is a good example for this purpose [55]. Oracle Berkeley DB XML is an open source, embeddable XML database with XQuery-based access to documents stored in containers and indexed based on their content. It runs in process with the application with no need for human administration.

Membership degrees of each acoustic class to the defined fuzzy sets are illustrated in Table 4.1 in Chapter 4. As mentioned previously, parameters in that table have been experimentally determined. To improve the performance of the retrieval stage, fuzzy membership functions associated with some high level features may be introduced. These features can be extracted from each segment individually and represent the content of that segment. They may be extracted from audio content of the segment as well as visual or textual content of the news video.

The proposed system consists of different stages and we claim that each stage can be replaced by alternative methods without affecting other stages. A future work item may be developing more robust methods for the retrieval stage. For instance, the defined domain based classes (e.g., commercial, anchor) may be trained by HMM classification method using visual features. By this way, one HMM model will be constructed for each domain based class. Initially, queried audio segments are retrieved depending on the segmentation results and the membership degrees as in our proposed system. Then, some of the retrieved audio segments may be eliminated by using these HMM models. If the probability that the retrieved audio segment (e.g., commercial segments) is produced by the model (e.g., commercial HMM model) is lower than a threshold, then that segment can be deleted from the query result.

Text captions may be used to get additional information on segments of the news video, such as the names of the people shown in the video or the site where the action takes place. Extraction of text information from video caption may be performed by integrating a traditional OCR (Optical Character Recognition) within our system. Techniques for the extraction and OCR of caption text for the news video have been examined in [56]. Another way to get additional information may be adapting automatic speech recognition (ASR) techniques to our system.

Another future work may be experiencing on different features or combined feature sets on Query-By-Example (QBE) and Expressive Summary Search (ESS) methods. For instance, MFCC is suggested for speech signals as the underlying feature set for these methods [32], thus MFCC might be the first base for the trials.

A final future work item may be diversifying the supported query types in our system. A new query type, “query by keywords”, would be defined. This type of query is also based on acoustic features of an audio stream and may handle generic sounds, including sound effects, animal sounds and natural scenes. If the user does not have a recorded sample at hand, he may select one keyword from a vocabulary and perform a query to retrieve similar sounds to the selected keyword. This requires learning a mapping between the keyword and acoustic features of the sound recording. To build such a vocabulary, first a number of audio recordings, best representing the keyword should be found. Then, an HMM model should be trained for each keyword. Finally, for each HMM model, an optimum sequence of acoustic features should be extracted by using Viterbi algorithm [45]. In the retrieval phase, the proposed similarity measurements could be adapted between the vectors of the selected keyword and audio segments.

REFERENCES

[1] Kim, H.G., Moreau, N., Sikora, T.: MPEG-7 Audio and Beyond. John Wiley and Sons, Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England (2005)

[2] Technology, I.: Part4 Audio. ISO/IEC 15938-4:2002. In: Multimedia Content Description Interface. International Organisation for Standardisation (2002)

[3] Chen, L., Şule Gündüz, Özsu, M.T.: Mixed type audio classification with support vector machine. In: International Conference on Multimedia and Expo, IEEE (July 2006) 781–784

[4] Liu, Z., Huang, Q.: Classification of audio events in broadcast news. In: IEEE 2nd Workshop Multimedia Signal Processing. (December 1998) 364–369

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

[6] Zhu, Y., Ming, Z., Huang, Q.: Svm-based audio classification for content-based multimedia retrieval. In: Multimedia Content Analysis and Mining. (July 2007) 474–482

[7] Mehta, D., E.S.V.N.L.S.Diwakar, C.V.Jawahar: A rule-based approach to image retrieval. In: IEEE Region 10 Conference on Convergent Technologies (TENCON). (October 2003) 586–590

[8] Doğan, E., Sert, M., Yazıcı, A.: Content-based classification and segmentation of mixed-type audio by using mpeg-7 features. In: Int. Conf. on Advances in Multimedia (MMEDIA 2009), IEEE (July 2009)

[9] Doğan, E., Sert, M., Yazıcı, A.: Content-based retrieval of audio in news broadcasts. In: To be appear in Flexible Query Answering Systems (FQAS 2009). (October 2009)

[10] Zhang, T., , Zhang, T., c. Jay Kuo, C.: Hierarchical system for content-based audio classification and retrieval. In: Proc. of Int. Conf. on Acoustics, Speech, and Signal Processing. 398–409

[11] Lu, L., Jiang, H., Zhang, H.: A robust audio classification and segmentation method. In: ACM Multimedia. (2001) 203–211

[12] Breebaart, J., McKinney, M.: Features for audio classification. In: Philips Symposium of Intelligent Algorithms. (2002)

[13] Comeau, M.: Acoustic Segmentation. Technical Report, CRIM (2006)

[14] Dong, R., Hermann, D., Cornu, E., Chau, E.: Low-power implementation of an hmm-based sound environment classification algorithm for hearing aid application. In: 15th European Signal Processing (EUSIPCO). (September 2007)

[15] Zdansky, J., David, P.: Automatic audio segmentation of tv broadcast news. In: Radioelektronika 2004. (April 2004) 358–361

[16] Kim, H.G., Moreau, N., Sikora, T.: Audio classification based on mpeg-7 spectral basis representations. IEEE Transactions on Circuits and Systems for Video Technology **14**(5) (May 2004) 716–725

[17] Wang, J.C., Wang, J.F., He, K.W., Hsu, C.S.: Environmental sound classification using hybrid svm/knn classifier and mpeg-7 audio low-level descriptor. In: International Joint Conference on Neural Networks, IEEE (July 2006) 1731–1735

[18] Simmermacher, C., Deng, D., Cranefield, S.: Feature analysis and classification of classical musical instruments: An empirical study. In: Industrial Conference on Data Mining. (2006) 444–458

[19] Xiong, Z., Radhakrishnan, R., Divakaran, A., Huang, T.: Comparing mfcc and mpeg-7 audio features for feature extraction, maximum likelihood hmm and entropic prior hmm for sports audio classification. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Volume 5. (April 2003) 628–631

[20] Xiong, Z., Radhakrishnan, R., Divakaran, A., Huang, T.: Temporal audio segmentation using mpeg-7 descriptors. In: SPIE Storage and Retrieval for Media Databases. Volume 5021. (January 2003)

[21] Zibert, J., Vesnicer, B., Mihelic, F.: A system for speaker detection and tracking in audio broadcast news. Informatica (April 2008)

[22] Meinedo, H., Neto, J.A.: Audio segmentation, classification and clustering in a broadcast news task. In: Int. Conf. Acoust., Speech, Signal Process, ICASSP 2003. Volume 2. (April 2003) 5–8

[23] NWE, T.L., LI, H.: Broadcast news segmentation by audio type analysis. In: ICASSP, IEEE International Conference (March 2005) 1065–1068

[24] Munoz-Exposito, J.E., Galan, S.G., Reyes, N.R., Candeas, P.V.: Speech/music discrimination based on warping transformation and fuzzy logic for intelligent audio coding. Applied Artificial Intelligence **23**(5) (2009) 427–442

[25] Nitanda, N., Haseyama, M.: Audio-based shot classification for audiovisual indexing using pca, mgd and fuzzy algorithm. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E90-A**(8) (2007) 1542–1548

[26] Tao, Y., Zu, D., Du, P.: A fuzzy logic based speech extraction approach for e-learning content production. In: Audio, Language and Image Processing. (2008) 298–302

[27] Liu, M., Wan, C., Wang, L.: Content-based audio classification and retrieval using a fuzzy logic system: towards multimedia search engines. Soft Computing **6**(5) (2002) 357–364

[28] Spevak, C., Favreau, E.: Soundspotter - a prototype system for content-based audio retrieval. In: COST G-5 Conf. on Digital Audio Effects (DAFX-02). (2002)

[29] Wan, C., Liu, M.: Content-based audio retrieval with relevance feedback. Pattern Recogn. Lett. **27**(2) (2006) 85–92

[30] Chechik, G., Ie, E., Rehn, M., Bengio, S., Lyon, D.: Large-scale content-based audio retrieval from text queries. In: 1st ACM int. conf. on Multimedia information retrieval, (MIR-08), New York, NY, USA, ACM (2008) 105–112

[31] Virtanen, T., Helen, M.: Probabilistic model based similarity measures for audio query-by-example. In: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. (2007) 82–85

[32] Sert, M., Baykal, B., Yazıcı, A.: Structural and semantic modeling of audio for content-based querying and browsing. In: Lecture Notes in Artificial Intelligence, Springer-Verlag (2006) LNAI 4027:319–330

[33] Zhu, Y., Ming, Z.: Svm-based video scene classification and segmentation. In: MUE. (2008) 407–412

[34] Nakamura, Y., Kanade, T.: Semantic analysis for video contents extraction - spotting by association in news video. In: Fifth ACM Int. Multimedia Conf. (1997)

[35] Bertini, M., Del Bimbo, A., Pala, P.: Content-based indexing and retrieval of tv news. Pattern Recogn. Lett. **22**(5) (2001) 503–516

[36] Sert, M., Baykal, B.: Web-based query engine for content-based and semantic retrieval of audio. In: Consumer Electronics, 2004 IEEE International Symposium. (2004) 485–490

[37] Vapnik, V.: The Support Vector Method of Function Estimation. In: Generalization in Neural Network and Machine Learning. Springer-Verlag, New York (1998) 239–268

[38] Gunn, S.: Support Vector Machines for Classification and Regression. Technical Report, Image Speech and Intelligent Systems Research Group (1998)

[39] Burges, C.J.C.: A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery 2 (1998)

[40] Baum, L., Petrie, T.: Statistical inference for probabilistic functions of finite state markov chains. Ann. Math. Stat. **37** (1966) 1554–1563

[41] Baum, L., Egon, J.: An inequality with applications to statistical estimation for probabilistic functions of a markov process and to a model for ecology. Bull. Amer. Meteorol. Soc. **73** (1967) 360–363

[42] Baum, L., Sell, G.: Growth functions for transformations on manifolds. Pac. J. Math. **27**(2) (1968) 211–227

[43] Baum, L., Petrie, T., Soules, G., Weiss, N.: A maximization technique occuring in the statistical analysis of probabilistic functions of markov chains. Ann. Math. Stat. **41**(1) (1970) 164–171

[44] Baum, L.: An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. Inequalities **3** (1972) 1–8

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

[46] Partners In Rhyme, Inc: Royalty Free Music and Sound Effects. http://www.partnersinrhyme.com, Last date accessed: August, 2009.

[47] Kiranyaz, S., Qureshi, A.F., Gabbouj, M.: A generic audio classification and segmentation approach for multimedia indexing and retrieval. In: European Workshop on the Integration of Knowledge, Semantics and Digital Media Technology, EWIMT 2004. (2004) 55–62

[48] Allen, J.: Maintaining knowledge about temporal intervals. Communications of ACM **26**(11) (1983) 832–843

[49] Wellhausen, J., Crysandt, H.: Temporal audio segmentation using mpeg-7 descriptors. In: SPIE Storage and Retrieval for Media Databases. (2003)

[50] Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on In Information Theory **28**(2) (1982) 129–137

[51] W3C: Extensible Markup Language (XML). http://www.w3.org/XML, Last date accessed: August, 2009.

[52] Chang, C.C., Lin, C.J.: Libsvm – A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/ cjlin/libsvm/, Last date accessed: August, 2009.

[53] François, J.M.: Jahmm- Hidden Markov Model. http://www.run.montefiore.ulg.ac.be/ francois/software/jahmm/, Last date accessed: August, 2009.

[54] Wikimedia Foundation, Inc.: Hash table. http://en.wikipedia.org/wiki/Hash_table, Last date accessed: August, 2009.

[55] Oracle: Oracle Berkeley DB XML. http://www.oracle.com/database/berkeley-db/xml/index.html, Last date accessed: August, 2009.

[56] Sato, T., Kanade, T., Hughes, E.K., Smith, M.A.: Video ocr for digital news archive. Content-Based Access of Image and Video Databases, Workshop on (1998) 52–60

APPENDIX A

EXPERIMENTAL RESULTS FOR THE CLASSIFICATION APPROACH

Table A.1: Classification Result of Non-Speech and With-Speech

Feature	ASP				ASC + ASS + ASF (ASCSF)			
Classifier	SVM		HMM		SVM		HMM √	
hit/miss	hit	miss	hit	miss	hit	miss	hit	miss
Non-Speech	1749	614	2078	285	1791	572	2069	294
With-Speech	3814	271	3675	410	4042	43	3854	231
Accuracy Rate	86.3%		89.2%		90.5%		91.9%	

Table A.2: Classification Result of Pure Speech and Mixed Speech

Feature	ASP				ASC + ASS + ASF (ASCSF)			
Classifier	SVM		HMM		SVM √		HMM	
hit/miss	hit	miss	hit	miss	hit	miss	hit	miss
Pure-Speech	1659	290	1817	132	1773	176	1726	223
Mixed-Speech	1737	399	1558	578	1879	257	1786	350
Accuracy Rate	83.1%		82.6%		89.4%		86.0%	

Table A.3: Classification Result of Speech over Environmental Sound (S+E) and Speech over Music (S+M)

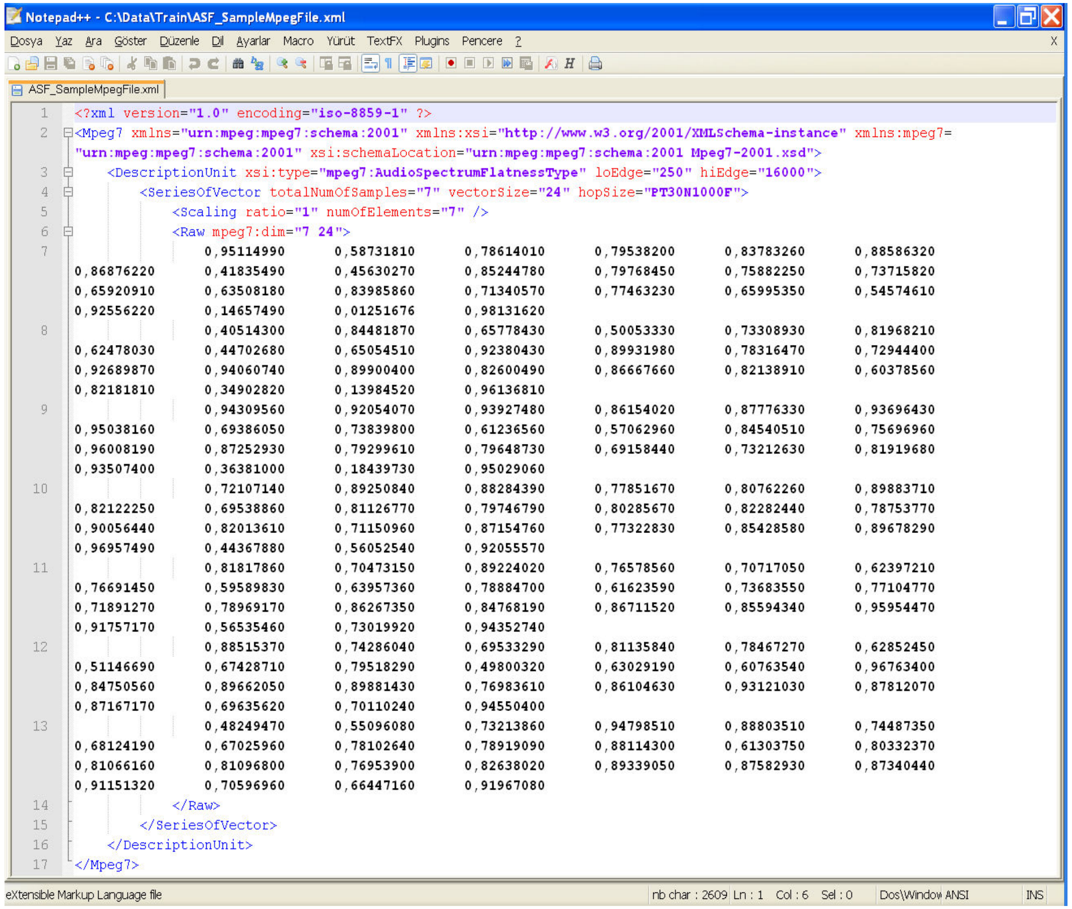
Feature	ASP				ASC + ASS + ASF (ASCSF)			
Classifier	SVM		HMM		SVM		HMM √	
hit/miss	hit	miss	hit	miss	hit	miss	hit	miss
S+E	537	231	639	129	446	322	637	131
S+M	682	686	735	633	1322	46	1182	186
Accuracy Rate	57.1%		64.3%		82.8%		85.2%	

Table A.4: Classification Result of Environmental Sound and Music

Feature	ASP				ASC + ASS + ASF (ASCSF)			
Classifier	SVM		HMM		SVM √		HMM √	
hit/miss	hit	miss	hit	miss	hit	miss	hit	miss
Env.Sound	552	492	929	115	941	103	952	92
Music	1101	218	1032	287	1317	2	1305	14
Accuracy Rate	70.0%		83.0%		95.6%		95.5%	

APPENDIX B

A SAMPLE OF MPEG-7 XML DOCUMENT



```
<?xml version="1.0" encoding="iso-8859-1" ?>
<Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:mpeg7=
"urn:mpeg:mpeg7:schema:2001" xsi:schemaLocation="urn:mpeg:mpeg7:schema:2001 Mpeg7-2001.xsd">
    <DescriptionUnit xsi:type="mpeg7:AudioSpectrumFlatnessType" loEdge="250" hiEdge="16000">
        <SeriesOfVector totalNumOfSamples="7" vectorSize="24" hopSize="PT30N1000F">
            <Scaling ratio="1" numOfElements="7" />
            <Raw mpeg7:dim="7 24">
                0,95114990      0,58731810      0,78614010      0,79538200      0,83783260      0,88586320
0,86876220      0,41835490      0,45630270      0,85244780      0,79768450      0,75882250      0,73715820
0,65920910      0,63508180      0,83985860      0,71340570      0,77463230      0,65995350      0,54574610
0,92556220      0,14657490      0,01251676      0,98131620
                0,40514300      0,84481870      0,65778430      0,50053330      0,73308930      0,81968210
0,62478030      0,44702680      0,65054510      0,92380430      0,89931980      0,78316470      0,72944400
0,92689870      0,94060740      0,89900400      0,82600490      0,86667660      0,82138910      0,60378560
0,82181810      0,34902820      0,13984520      0,96136810
                0,94309560      0,92054070      0,93927480      0,86154020      0,87776330      0,93696430
0,95038160      0,69386050      0,73839800      0,61236560      0,57062960      0,84540510      0,75696960
0,96008190      0,87252930      0,79299610      0,79648730      0,69158440      0,73212630      0,81919680
0,93507400      0,36381000      0,18439730      0,95029060
                0,72107140      0,89250840      0,88284390      0,77851670      0,80762260      0,89883710
0,82122250      0,69538860      0,81126770      0,79746790      0,80285670      0,82282440      0,78753770
0,90056440      0,82013610      0,71150960      0,87154760      0,77322830      0,85428580      0,89678290
0,96957490      0,44367880      0,56052540      0,92055570
                0,81817860      0,70473150      0,89224020      0,76578560      0,70717050      0,62397210
0,76691450      0,59589830      0,63957360      0,78884700      0,61623590      0,73683550      0,77104770
0,71891270      0,78969170      0,86267350      0,84768190      0,86711520      0,85594340      0,95954470
0,91757170      0,56535460      0,73019920      0,94352740
                0,88515370      0,74286040      0,69533290      0,81135840      0,78467270      0,62852450
0,51146690      0,67428710      0,79518290      0,49800320      0,63029190      0,60763540      0,96763400
0,84750560      0,89662050      0,89881430      0,76983610      0,86104630      0,93121030      0,87812070
0,87167170      0,69635620      0,70110240      0,94550400
                0,48249470      0,55096080      0,73213860      0,94798510      0,88803510      0,74487350
0,68124190      0,67025960      0,78102640      0,78919090      0,88114300      0,61303750      0,80332370
0,81066160      0,81096800      0,76953900      0,82638020      0,89339050      0,87582930      0,87340440
0,91151320      0,70596960      0,66447160      0,91967080
            </Raw>
        </SeriesOfVector>
    </DescriptionUnit>
</Mpeg7>
```

Figure B.1: A Sample Mpeg-7 ASF Document