EFFICIENT AND ROBUST AUDIO FINGERPRINTING

FENG SHUYU

(B.Eng, Wuhan University, PRC)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2007

Acknowledgement

I am indebted to my supervisor, Prof. Ooi Beng Chin, for giving me the guidance, advice and encouragement throughout the period of my graduate study. He gave me unconditional support and freedom to persist in this research topic when I encountered problems. I feel fortunate to be taken into his group. His rigorous attitude and great passion towards research and work will influence my future career.

I would like to thank Dr. Wang Ye for the helpful discussions. From his module, CS5249, I learned lots of background knowledge in audio signal processing, which provides a foundation to my research work.

I am grateful for the encouragements, discussions and suggestions I have received from the friends in database group, especially Cui Bin, Chen Yueguo, Xu Linhao, Yu Bei, Dai Bintian, Yang Xiaoyan, Chen Su and Wu Sai.

Finally, I would like to thank my family for their deepest love and support, and all my friends for their encouragements.

Table of Contents

Summary

The explosive amount of music data available on the Internet in recent years has increased the demands to develop new methods to search and retrieve such data effectively. Currently, most music search engines rely on text labels or symbolic data, rather than the underlying acoustic contents. A content-based music information retrieval system has the ability to find similar songs based on the underlying acoustic features which are derived from the signals, regardless of metadata descriptions or file names. Potential applications include automatic music identification, copyright protection, and so forth.

In this thesis, we examine the problem of content-based music identification by efficient and robust audio fingerprinting. Audio fingerprinting is a technology to identify some piece of unknown audio based on a compact set of features derived from the audio signal. It provides reliable and fast means for content-based music information retrieval. Since music signals usually suffer from various distortions or modifications such as mp3 compression, noise addition and so forth, designing robust audio fingerprinting system which can resist effects of these distortions becomes crucial. Besides, retrieval efficiency is also an important requirement in practical applications when the size of music database increases rapidly.

We propose to improve the effectiveness and efficiency of audio fingerprinting system resistent to distortions. In particular, we focus on three important modules:

feature extraction, fingerprint modeling and matching, which affect the accuracy and efficiency of the whole system.

Firstly, we study and compare several spectral features, including Mel-Frequency Cepstral Coefficients, chroma spectrum, constant Q spectrum, and product spectrum. The former three features are derived only from magnitude spectrum, and have been widely used in music signal processing and modeling. Product spectrum takes advantage of the phase spectrum by using the product of magnitude spectrum and group delay function. It shows effectiveness in robust speech recognition. Experimental results show that product spectrum based feature is more robust than the former three features in audio fingerprinting.

Secondly, we propose a pattern accumulative similarity measure (PAS) which better captures the similarity between music data and is discriminative under distortions that may result in mismatches in both time and amplitude axes. Experimental results show that PAS has improvement in effectiveness and efficiency compared with Euclidean distance and DTW distance.

Thirdly, we use Gaussian mixture model (GMM) to boost the robustness of audio fingerprints. First, a GMM is trained for the music database by using the Expectation Maximization (EM) algorithm, which better describes the distribution of acoustic feature space. Then, based on the trained GMM, feature vectors of music database and test dataset are all converted into symbolic tokens. Experimental results show the advantages of GMM modeling that it maintains high accuracy under severe noise distortions.

Finally, we compare our method with an audio fingerprinting approach, AudioDNA. Our method is similar to AudioDNA except that the acoustic features and the similarity measure are different. Experimental results show that our method is more resistent to noise distortions than AudioDNA.

List of Tables

List of Figures

Chapter 1

Introduction

As the amount of music data in multimedia databases increases rapidly, there are strong needs to investigate and develop content-based music information retrieval systems in order to support effective and efficient analysis, retrieval and management for music data. Compared with content-based image retrieval, content-based music information retrieval (CBMIR) is a relatively new field, and the existing techniques are far from perfect [53]. Most of current used music information retrieval (MIR) systems are based on metadata of music, such as title, singer, composer, lyrics, and album. It requires users to recall and specify metadata of music, which becomes a major restriction on users' queries. However, at times users' request can be based on the contents of the music. For example, "tell me the name of the audio clip", "skip the repeated chorus of the song", or "who is singing the melody on this recording?". These queries are based on acoustic features, such as melody, harmony, rhythm, and so forth. Therefore, CBMIR systems are essentially required.

Audio fingerprinting aims to identify some piece of unknown audio in a labeled audio database. Compared with the conventional MIR systems which are based

on metadata of music like title or lyrics, audio fingerprinting systems are based on robust acoustic features, called audio fingerprints, which are extracted from the music signal. Robust audio fingerprints mean that they should have close resemblance to the fingerprints of a similar song with signal processing operations such as mp3 compression and noise addition, while still distinguish from fingerprints of different songs. It has vast applications, including music identification, broadcast monitoring, and surveillance of the transmission of audio over the Internet.

The main objective of our work is to improve the accuracy and efficiency of audio fingerprinting systems. Firstly, we study and compare several spectral features, and find that the feature derived from product spectrum which combines phase spectrum with magnitude spectrum is more robust than other spectral features which are derived only from magnitude spectrum. Secondly, a pattern accumulative similarity is proposed to better measure the similarity between audios under several types of distortions. Thirdly, Gaussian mixture model (GMM) is used to model audio fingerprints, boosting the robustness of audio fingerprints under noise distortions while making fingerprints more concise.

In this chapter, we first introduce the framework, properties and applications of audio fingerprinting systems. After analyzing the problems of audio fingerprinting due to distortions, we summarize our main contributions in tackling these problems. Finally, the structure of the thesis is given.

1.1 Audio Fingerprinting

Audio fingerprinting is a technology to identify some piece of unknown audio in a labeled audio database based on a compact set of features, called audio fingerprints, which are derived from the signal. It provides reliable and fast means for

content-based music information retrieval as the audio fingerprints are compact summarizations of music files. The function of audio fingerprint is similar to that of human fingerprint.

1.1.1 Framework

Figure 1.1 (adapted from [10]) illustrates the general framework of audio fingerprinting systems. It contains two major components: fingerprint extraction and matching. The former extracts and models digital audio signals into audio fingerprints which are discriminative enough to identify unlabeled distorted versions of a song as the same song stored in a song database. The latter efficiently looks up the audio fingerprints against the database and judges whether there is a matching song in the database. The whole system specifically consists of five modules: front-end, fingerprint modeling, fingerprints and metadata database, database look-up, and hypothesis testing.

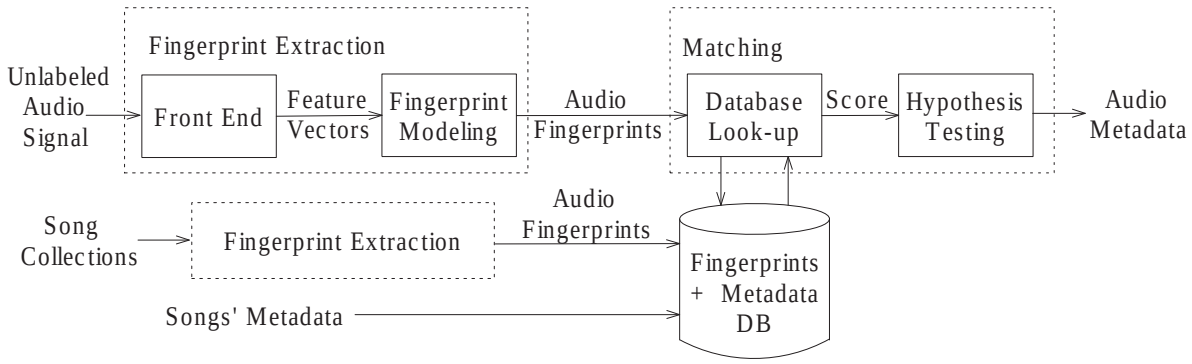


Figure 1.1: General framework of audio fingerprinting systems

Front-End converts an audio signal into acoustic features which are fed into the fingerprint modeling module. It further contains pre-processing, windowing

and overlapping, feature extraction, and post-processing four steps. Feature extraction is the core part.

Fingerprint Modeling usually receives a sequence of feature vectors passed from the front-end. In the most straightforward form, audio fingerprints can be modeled as a sequence (trace, trajectory) of feature vectors, without performing any further processing. As redundancies exist in successive frames in time, inside a song and across the whole database, further modeling steps are usually used to model audio fingerprints into more robust and concise representations.

Fingerprints and Metadata Database stores the fingerprints of song database, and links fingerprints of each song to relevant tag or metadata. The size and the structure of the database affect efficiency and accuracy of the system.

Database Look-up compares the fingerprints of the query song with the fingerprints in the song database. If a credible similarity exists, the query is considered to be found as the song in the database. It first defines the similarity measure between audio fingerprints and then performs fast search, using indexing or pruning strategies, to return a set of matching songs.

Hypothesis Testing aims to answer whether the query is in the labeled song database or not. During the comparison between query's fingerprints against audio fingerprints database, similarity scores are obtained. Song with a score beyond a certain threshold is regarded as a correct identification. The choice of the threshold depends on the used fingerprint model, the discriminative information of the query, the similarity of fingerprints in the database, and the database size.

1.1.2 Requirements

A practical audio fingerprinting system should meet accuracy and efficiency requirements.

A. Accuracy

Accuracy is the foremost requirement in most of audio fingerprinting systems. It depends on robustness of audio fingerprints and similarity measures.

- **Robustness**

  The robustness of audio fingerprints is related to acoustic features and fingerprint modeling methods. In reality, music signals usually suffer from various distortions or modifications, such as mp3 compression, noise addition, channel distortion, and so forth. Therefore, robust audio fingerprints which can resist these effects become essential. The audio fingerprints should not be easily affected by signal processing operations, but be still distinctive to the audio signal in order to distinguish between different songs.

- **Similarity measure**

  Audio fingerprints may suffer from various distortions which could result in misalignment in time or amplitude. Therefore, suitable similarity measures which can maximize the similarity between distorted version and original audio while minimize the similarity between different audios are needed to prevent mismatch.

B. Efficiency

Efficiency is a crucial requirement for many applications especially when the size of music database increases rapidly. However, there is a tradeoff between efficiency and accuracy in most cases. The efficiency is related to the computational costs of both fingerprint extraction and search algorithms, the size of fingerprints, and the query granularity.

- **Algorithm complexity**

  It mainly refers to the computational costs of both fingerprint extraction and search algorithms.

- **Fingerprint size**

  Compact fingerprint can reduce database storage, and moreover speed up the search, as most of the data can be stored in the main memory.

- **Granularity**

  Granularity means the length of an audio clip needed in order to identify the clip. It depends on applications. In some applications a whole song is used for identification, whereas in others only a short excerpt of audio is used.

1.1.3 Applications

There are several typical applications of audio fingerprinting systems.

- **Recording identification**

  One typical scenario is when a person with a cell phone hears a broadcasting song which he or she wishes to know more about, for instance, the song title, singer, or album. The user records a 10-second clip of the song using his cell

phone, sends it to some service provider like Shazam [38], and then waits a few minutes to get the feedback which contains relevant information of the song. At the server side, the audio fingerprinting system retrieves in the song database to find the desired song with relevant information using the received example recorded in a noisy environment with lossy encoding of cell phone. Therefore, the audio fingerprints must be robust in the face of distortions.

- **Copyright detection**

  Another application is to restrict users from illegally uploading music to the Internet. To protect copyrights, the uploaded music will be scanned and checked against a database of copyright protected songs so that any protected content will be blocked. Similar applications include integrating audio fingerprinting into p2p application which allows p2p technology to be used in a copyright respected manner, and radio station monitoring.

1.2 Problems and Motivations

In audio fingerprinting, mismatch between query and database occurs when queries suffer from various distortions or modifications. In our work, we focus on three major distortions, resulting from lossy transmission channel, source editing, and background noise. In the following, we will analyze problems in audio fingerprinting under these distortions.

Problem 1 When users send their audio data through Internet or wireless network, they will probably face packet losses due to lossy transmission channel. Figure 1.2 shows the differences between the reconstructed audio and the original version at the server side. We do not consider packet loss recovery

here, since our main objective is to define suitable similarity measure. As misalignment in time axis is often in company with this distortion, mismatch will occur if Euclidean distance is used as a similarity measure between these two audios. Dynamic Time Warping (DTW) can find the optimal alignment between two sequences, but it is computationally expensive. Although using global constraints can speed up DTW distance calculation, the value of r, which is the allowed range of warping, affects the matching accuracy.

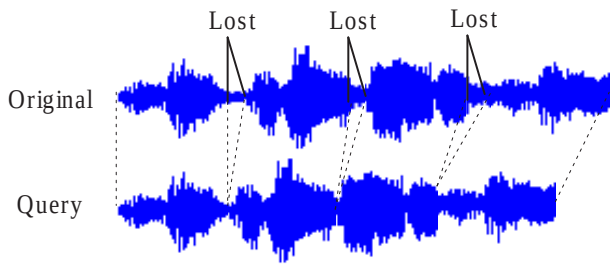


Figure 1.2: Mismatch due to lossy transmission channel

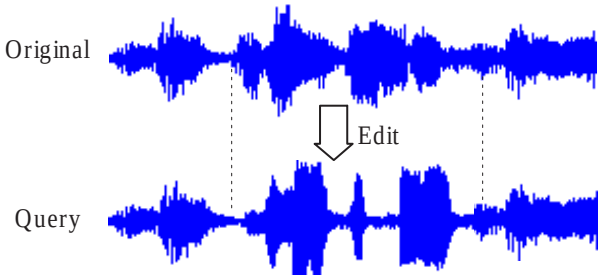


Figure 1.3: Mismatch due to source editing

Problem 2 In some applications, the audio is usually edited. For example, users can replace parts of a song before uploading. For another example, radio stations often add broadcaster's speech into a song while broadcasting. In the first case, parts of the original music are completely replaced by another

audio, as shown in Figure 1.3. In the second case, parts of the original music become background music when human speech is added in. Both cases could result in mismatch between query and the original music. Euclidean distance is not suitable here because the accumulative distance between the edited parts and the original parts could counteract the similarity between the unedited parts.

Problem 3 Whenever users record an audio clip in a real environment, background noise will affect the quality of the recorded clip. Figure 1.4 shows the effect of background noise on the waveform. Due to noise distortions, the waveform of the recorded clip is quite different from that of the original music. Therefore, robust audio fingerprints which can reduce the effect of background noises become essential.

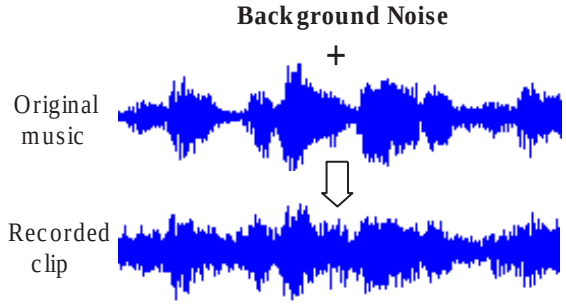


Figure 1.4: Mismatch due to background noise

In our work, we aim at improving accuracy and efficiency of audio fingerprinting systems. Specifically, in order to solve problem 1 and 2, we propose a new similarity measure which better captures the similarity between music data under distortions. It is not only effective under amplitude and time distortions, but also efficient in computation. To solve problem 3, we study acoustic features and fingerprint

modeling approaches to improve the robustness of audio fingerprints. First, we study and compare several typical spectral features, and then we use statistical modeling to generate robust and concise audio fingerprints.

1.3 Contributions

The main contributions of this thesis are as follows:

- We build a baseline of audio fingerprinting system on a database composed of 1000 songs. Using the standard acoustic feature, the Mel-frequency cepstral coefficient (MFCC) which is widely used in various audio-related applications, we obtain recognition accuracy and receiver operating characteristic (ROC) curve of the system, with regard to several types and levels of noise distortions.

  We also explore the effect of normalization and frame length on this baseline. Experimental results show that normalization can greatly improve recognition accuracy, and short frame, 46 ms, achieves better performance than long frame, 372 ms.

- We study and compare several typical spectral features in audio fingerprinting. Specifically, we have studied MFCC, chroma spectrum, constant Q spectrum, and product spectrum which incorporates magnitude spectrum as well as phase spectrum. Although these features have been used in many music/speech applications, their performance in audio fingerprinting are compared the first time.

Experimental results show that product spectrum based feature is more robust than the other three features in that it takes advantage of phase spectrum. It has better ROC performance under different noise distortions, and achieves 92.09% overall identification rate with 0.1% false alarm rate.

- A pattern accumulative similarity measure (PAS) is proposed, which accumulates the similarity of two audios along the matching path, whereas diminishes the effect of unmatch. It better captures the similarity between music data and is discriminative under distortions due to lossy transmission channel, source editing, and background noise.

  Experimental results show the effectiveness and efficiency of PAS compared with Euclidean distance and Dynamic Time Warping (DTW) distance. It can achieve 99% accuracy when a query audio is distorted with 10% data loss, and 100% accuracy when 50% of a query audio is edited, while keeping computationally efficient.

- Gaussian Mixture Model (GMM) modeling is used to generate robust and concise audio fingerprints, which reduces acoustic feature vectors into several types of tokens. First, the music database is trained using M Gaussian components with diagonal covariance matrices in an incremental procedure. Then, based on the trained Gaussian mixture model, acoustic feature vectors of music database and test dataset are all converted into symbolic tokens (acoustic events). GMM has advantages over other modeling approaches.

  Experimental results show the advantages of GMM modeling that it maintains high accuracies with respect to white noises of 6 different SNR levels from 20dB to -5dB, better than the performance when directly using feature vectors, or modeling with Principal Component Analysis (PCA) or Vector

Quantization (VQ). Besides, it reduces disk space and memory requirements, and speeds up the matching process as well.

- We compare our method with an existing audio fingerprinting approach, AudioDNA. Both methods model audio fingerprints as a sequence of acoustic events. Our method is different from AudioDNA in that the product spectrum based feature and the similarity measure PAS are used. Because AudioDNA is based on exact match of subsequence, its performance decreases as the noise distortions become more severe. As our method considers the effect of noise distortion, it achieves better performance.

  Experimental results show that our method is more resistent to noise distortions than AudioDNA. Our method can achieve 100% accuracy when queries are 5 seconds clips with 20dB babble noise distortion, but AudioDNA can only achieve 96%. As the noise distortion becomes severe, our method can maintain good accuracy whereas AudioDNA degenerates. Our method also shows good performance with queries of different lengths.

1.4 Structure of Dissertation

The remainder of the thesis is organized as follows:

- **Chapter 2. Related Work**

  In this chapter, we first briefly introduce the related work in feature extraction, fingerprint modeling and matching respectively, because these three modules comprise the core parts of an audio fingerprinting system. Then, we describe a few representative systems, and analyze their limitations. Finally, we give a brief overview of our system.

- **Chapter 3. Feature Extraction**

  Feature extraction is the basis for content-based music information retrieval. In this chapter, we focus on the extractions of spectral features. First, we briefly introduce the four steps of front-end processing and the importance of feature extraction. Then we introduce several typical spectral features and describe their calculations in detail. Finally, we compare these features to show their similarities and differences.

- **Chapter 4. Fingerprint Modeling**

  In this chapter, we study the methods for fingerprint modeling. We first briefly introduce the motivations behind GMM modeling. Then we describe in detail the GMM modeling, including theory for GMM, model training process, and GMM token sequence generation. Finally, the advantages of GMM are explained, compared with three modeling approaches.

- **Chapter 5. Matching**

  In this chapter, we first give an overview of the matching module of audio fingerprinting systems. After analyzing limitations of commonly used similarity measures, we introduce the pattern accumulative similarity measure and the search strategy. Finally, we describe the method for hypothesis testing.

- **Chapter 6. Evaluation**

  In this chapter, we will describe the experimental results of the proposed methods in previous chapters. Specifically, we will first present the music database used for the experiments. Then, we study the robustness of acoustic features by testing the effect of normalization and frame length, and comparing the ROC performance between different spectral features. Furthermore,

we evaluate the effectiveness and efficiency of PAS and GMM modeling. Finally, we compare our method with an existing audio fingerprinting method and test the system performance with respect to different query lengths.

- **Chapter 7. Conclusion**

  We conclude the thesis in this chapter. We summarize our work on improving the query effectiveness and efficiency for audio fingerprinting systems resistent to distortions, and indicate the areas of future work.

Chapter 2

Audio Fingerprinting System

In this chapter, we will first review the background of audio fingerprinting systems, including feature extraction, fingerprint modeling and matching three aspects. Then, we introduce and analyze some state-of-the-art systems. Finally, we present the system overview of our method.

2.1 Background

A number of audio fingerprinting systems have been developed in recent years. [10] has provided a comprehensive review. In this section, we will first introduce the related work in feature extraction, fingerprint modeling and matching respectively, because these three modules comprise the core parts of an audio fingerprinting system. Current systems vary from each other in these three modules. Then, we will describe and analyze some representative systems.

2.1.1 Feature Extraction

One major difference of existing audio fingerprinting systems lies in the used acoustic features. As audio signals are usually distorted due to noise addition, compression and so forth, robust features which can correctly identify a song regardless of the level of distortion are needed. Previous studies have explored various acoustic features that are robust to distortions [11, 27, 29, 48, 51, 58], most of which are based on spectral features that use short-time Fourier transform to convert signals from time domain into frequency domain.

Cano et al. [11] use Mel-Frequency Cepstrum Coefficient (MFCC) which is a widely used feature that closely approximates the human auditory system's response. Herre et al. [29] use Spectral Flatness Measure (SFM) which is an estimation of tone-like or noise-like quality for a band in the spectrum. Haitsma et al. [27] describe a system that uses the energies of 33 bark-scaled bands to obtain 32-bit sub-fingerprints which are the sign of the energy band differences (in both time and frequency axes). Wang [58] generates fingerprints in the form of hash values of pairs of spectrum peaks. First, a constellation map is generated by spectrum peak detection on spectrogram. Then, each peak point is sequentially paired with points within its associated target zone. Finally, the two frequency values of point pair plus the time difference of this pair are hashed into a 32-bit unsigned integer. Sukittanon et al. [51] propose the geometric mean of modulation frequency using 19 bark-spaced band filters, which characterizes the time-varying behavior of audio signals. Seo et al. [48] use normalized spectral subband moments.

2.1.2 Fingerprint Modeling

Existing methods for fingerprint modeling can be classified into time-unpreserving and time-preserving methods.

Time-unpreserving methods ignore time information in audio. One simple method is to summarize the multidimensional vector sequences of a whole song (or fragment of it) into a single vector [22, 40, 55]. eTantrum [22] calculates the means and variances of the 16 bank-filtered energies of 30 seconds clip into a vector. Musicbrainz [40] includes the average zero crossing rate, the average spectrum and some more features into a vector. In [55], the normalized square root across mean energy of each frequency band is concatenated to the normalized standard deviation across RMS (Root Mean Square) power of each frequency band, generating a 30-coefficient vector. This kind of methods can improve the efficiency of audio fingerprinting, but they degrade the accuracy especially when audio is under distortions, for much information is lost, such as vectors' distribution and order. A more sophisticated method is to train fragment of feature vectors into a single vector, using dimensionality reduction methods like Oriented Principal Component Analysis (OPCA) [8, 9]. Burges et al. [8, 9] use OPCA to train both undistorted and distorted data and project onto a set of non-orthogonal directions which minimize the variance of the true and distorted version of audio clips but maximize the variance of different audio clips. A vector of 64 coefficients is extracted from every 6 seconds audio clip. This method reduces the local statistical redundancies of feature vectors with respect to time. The third method is to model a sequence of feature vectors into a class, in the form of codebook [2] or probability model [47]. Each song in the database is modeled as a class, and the retrieval is regarded as a classification problem which assigns the query to the most similar class. Allamanche et al. [2] use Vector Quantization (VQ) to cluster feature vectors and

encode each song into a codebook which consists of a number of representative vectors. The feature vectors of the query are approximated by each song's codebook and the song with the smallest approximation error is selected. Ramalingam et al. [47] use Gaussian mixture model (GMM) to model each song, and the song with the highest likelihood is regarded as a match. Temporal evolution of audio is lost with this approximation.

In time-preserving methods, audio fingerprints are usually modeled as sequences of acoustic feature vectors [27, 48, 51]. For instance, audio fingerprints are modeled as bit vector sequences in [27], whereas vector sequences of real number in [48, 51]. When vectors are in the form of real numbers, these feature vector sequences can be regarded as multivariate time series (MTS). Generally, there are three approaches to deal with MTS. The first one is to treat it as multiple univariate time series, process separately and aggregate the final result [56]. However, as there are usually important correlations among the variables in MTS, an MTS should be treated as a whole. Besides, it costs much more time in calculation. The second approach is to reduce the dimensionality, transforming multivariate time series data into a univariate time series [52]. Analyzing and processing univariate time series is easier than multivariate time series and many researches have taken effort in studying univariate time series [1, 17]. The third approach is to model vectors into several classes by methods like clustering or statistical modeling. The whole sequence is transformed into a string of symbolic tokens. For example, Hidden Markov Models (HMMs) is used in [6, 11] to generate a string of acoustic events. In [6, 11], the feature vectors are first clustered into several classes, where each class is regarded as a type of acoustic event, and then modeled via HMM. Given a query, the feature vector sequence is converted into a string of acoustic events using the trained HMM model.

2.1.3 Matching

Similarity measure is a key component in the matching process. The choice of similarity measure depends on the representation of audio content. Appropriately choosing the similarity measure can greatly enhance the discriminating capability of the system, and increase the speed as well.

When audio fingerprints are modeled without preserving time information, measures like Euclidean distance, Itakura distance, Kullback-Leibler distance and likelihood are often used [9, 35, 47, 55]. In [9], Euclidean distance is used to measure the distance between two fingerprint vectors. In [55], a fingerprint is modeled as a vector with N coefficients. The Itakura distance between two fingerprints FP^m and FP^n is defined as the log ratio of the arithmetic mean of e_i to the geometric mean of e_i, where $e_i = \frac{FP_i^m}{FP_i^n}$ and $0 \leq i \leq N$. In [35], each audio segment is modeled by a Gaussian mixture model (GMM), and Kullback-Leibler distance is calculated between two GMMs. GMM is also used in [47]. Each song is modeled as a GMM. The query is compared with the database of pre-computed GMMs and the GMM that gives the highest likelihood for the query is identified as a correct match.

When time information is preserved, measures such as Euclidean distance (L2) or its variations, Dynamic Time Warping (DTW) distance, Hamming distance and so forth are often used [16, 27, 48, 51, 59]. Euclidean distance is used in [48, 51] to calculate the distances of two feature vectors, whereas DTW distance is used in [16]. In [27], fingerprints are modeled into bit vector sequences, and thus Hamming distance is used. In [59], the feature vector sequences are converted into strings, and edit distance is applied. Among these similarity measures, some are sensitive to amplitude distortions, i.e., Euclidean distance; some are computational expensive, i.e., DTW and edit distance.

2.1.4 Some State-of-the-art Systems

1. Philips scheme

Philips audio fingerprinting system [29] is one of the most widely used systems, and has been commercially deployed. For example, the Musiwave music identification service is available on the Spanish mobile carrier Amena, which uses the Philips fingerprinting method. Users can identify a song playing on radio via this mobile service.

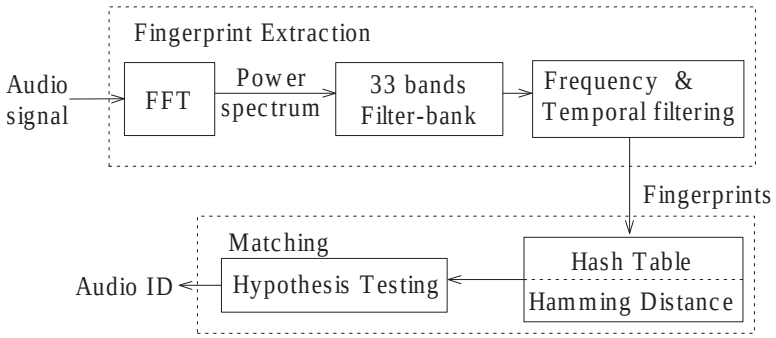


Figure 2.1: Overview of Philips audio fingerprinting scheme

An overview of Philips scheme is depicted in Figure 2.1. Signal is broken into a sequence of 370 ms frames with an overlap of 31/32. The large overlap ensures that sub-fingerprints vary slowly over time. Power spectrum is extracted from each window, and passed to a 33 bands filter-bank of a range 300-2000Hz. The filter-bank reflects the perceptual characteristics of an audio signal. A sub-fingerprint for each frame is calculated based on the sign of the power spectrum, differentiated simultaneously along the time and frequency axes. This differentiation of spectrum along the frequency and time axes benefits in two ways. First, it mimics high-pass filtering and may be possible to remove undesirable perturbations. Second, the differentiated power spectrum is uncorrelated with its temporal and frequency

neighbors. In this way, a sub-fingerprint is typically represented as a 32-bit code for each frame. The 32-bit code is usually indexed by a hash table. The bit is assigned as

$$H(n,m) = \begin{cases} 1 & \text{if } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) > 0 \\ 0 & \text{if } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) \leq 0 \end{cases}$$

where $E(n,m)$ is the energy of the n-th frame and the m-th band.

A fingerprint block which contains 256 sub-fingerprints is the basic unit to identify a song. For fast database lookup, a two-phase search algorithm is used. In the first phase, the positions that match any sub-fingerprint in the query fingerprint block are quickly found by looking up on the hash table. And full fingerprints comparisons are only performed at candidate positions pre-selected in the first phase. The best-match result is determined under the Hamming distance between fingerprint blocks.

This scheme is quite efficient when the assumption that at least one sub-fingerprint in the query fingerprint block has an exact match at the optimal position in the database is valid. Experiments show that the assumption almost always holds for audio signal with slight distortions [29]. However, for signal with heavy distortions the assumption is not always valid. At this time, sub-fingerprints with an N-bit difference also need to be checked. Therefore, the matching process slows down. Besides, the scheme is insufficient in a real-noise condition. When some bands are corrupted by noise, the Hamming distance between a distorted sub-fingerprint and the original one could be large.

2. Shazam

Shazam [58] is a deployed commercial system available in the United Kingdom which uses audio fingerprinting to let a cell phone user identify a broadcasting song. Figure 2.2 is the system overview. Shazam's fingerprints are based on spectrogram peaks. Peaks are defined as time-frequency points with higher energy than their local neighbors. Pairs of peaks are identified according to some locality and time restrictions. The frequency components of peak pair plus their time difference form a triple, (f_1, f_2, Δt), to be hashed into a key value of 32 bit. The time offset t_1 (which is the time duration from the beginning of the audio to the first element in peak pair) and the audio ID form another 32 bits which are appended to the hash key. In this way, a 64-bit value, (key, t_1, ID), is generated and sorted according to the key value.

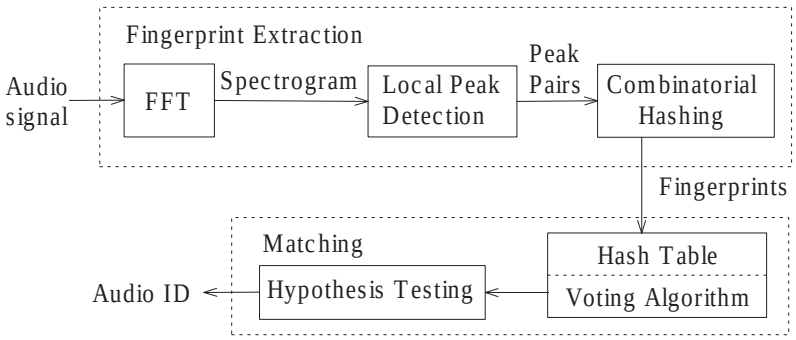


Figure 2.2: Overview of Shazam audio fingerprinting scheme

Given a query, a set of (key, t_1, ID) records are generated, and compared with the database. First, the matching key values are found and subsequently filtered according to the time offset information t_1. Then the match are counted for each track in the audio database until a significant match is found.

There is a drawback in this scheme, because it is based on an assumption that

even if the query signal is heavily distorted, a large number of local peaks will be at the same relative positions in both the query and the corresponding database signal. When the assumption does not hold under severe distortions, the fingerprints of the query and the database signal, both of which are generated from hashing, will be quite different.

3. RARE

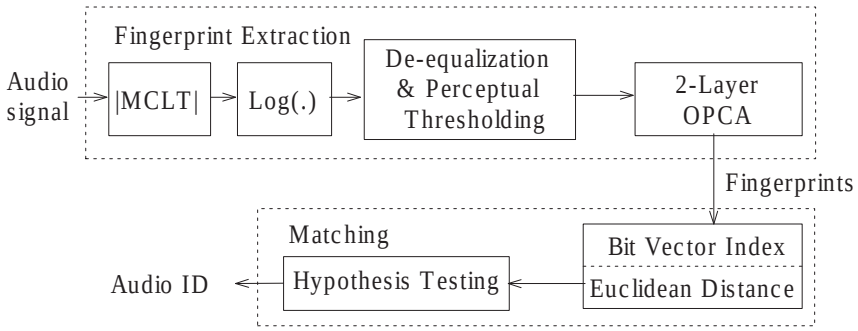


Figure 2.3: Overview of Microsoft audio fingerprinting scheme

Microsoft's Robust Audio Recognition Engine (RARE) [9, 24] uses dimensionality reduction techniques based on training. As shown in Figure 2.3, the signal is first converted to mono, downsampled to 11.025 kHz, and segmented into 372 ms frames overlapping by half. Then, the Modulated Complex Lapped Transform (MCLT) is applied and log spectrum is extracted. After de-equalization which removes distortions caused by frequency equalization and volume adjustment, and perceptual thresholding which removes distortions that cannot be perceived by a human, 2048 coefficients are obtained for each frame. The two layer DDA is based on Oriented Principal Component Analysis (OPCA) which uses both undistorted and distorted data for training. DDA projects the data onto directions that minimize the variance of the true and distorted version of audio clips but maximize the

variance of different audio clips. The first layer DDA projects 2048 coefficients into 64 coefficients. These projections are then concatenated into a vector with length of 2048 and projected into another 64 coefficients by the second layer DDA. In this way, a fingerprint of 64-coefficient vector is extracted from every 6 seconds audio clip and mapped into a point in a 64 dimensional space. For each fingerprint, a radius is computed using a validation set, generating a fingerprint hypersphere.

In the search process, the query fingerprint is mapped into the same 64 dimensional space, and the fingerprint hypersphere which contains the mapped query is found as a match. To avoid brute-force search, a two pass bit vector index [24] is used. Each dimension is divided into bins, and each bin has a bit vector index storing a list of data objects that overlap the bin. When the query is performed, exactly one bit vector index is selected for each dimension, and "AND" together to result in a set of candidate objects. In the second pass, linear scan is performed on these objects to find true matches.

4. AudioID

AudioID [2, 3] follows a general pattern recognition paradigm. As shown in Figure 2.4, the system has two modes: training and classification. Feature vectors are calculated from audio signals, which are subsequently interpreted as points in a high dimensional space. The set of psychoacoustic features studied in [2] includes loudness, spectral flatness measure (SFM) and spectral crest factor (SCF). In the training process, Vector Quantization (VQ) [31] is used to cluster feature vectors and encode each song of the database into a codebook which contains a smaller number of representative vectors. In the classification process, feature vectors of the query are extracted and approximated by all stored codebooks. For each class (codebook), the approximation error is accumulated and the query is assigned to

the class which yields the smallest accumulated approximation error.

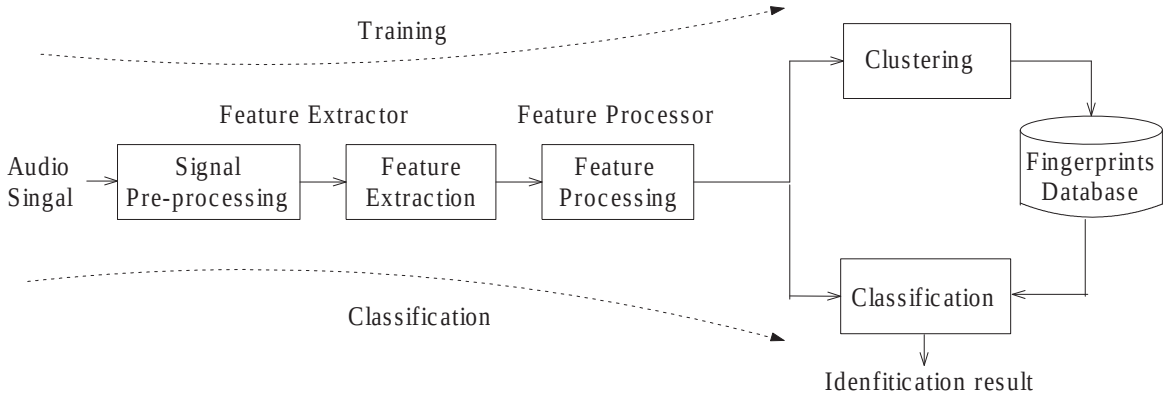


Figure 2.4: Overview of AudioID scheme

This system loses temporal evolution because it does not keep time information in fingerprint modeling. Besides, as VQ is a hard clustering, the space is divided into discrete cells. This is unnatural as the continuities of the vector space are broken. Soft clustering approaches which obtain continuous "smooth" classification can achieve better performance [31].

5. AudioDNA

AudioDNA [11] is the first prototype system designed for robust song detection in broadcast audio. Figure 2.5 illustrates its architecture. For the original songs, MFCCs are extracted from each audio waveform in the front-end module, and converted into a sequence of acoustic events, called AudioDNA, by modeling via Hidden Markov Models (HMM) [31]. This results in an AudioDNA database. In the query process, AudioDNA for each unlabelled query audio is extracted in the same manner, and compared with the AudioDNA database by approximate string matching to obtain the best resemblances to the query.

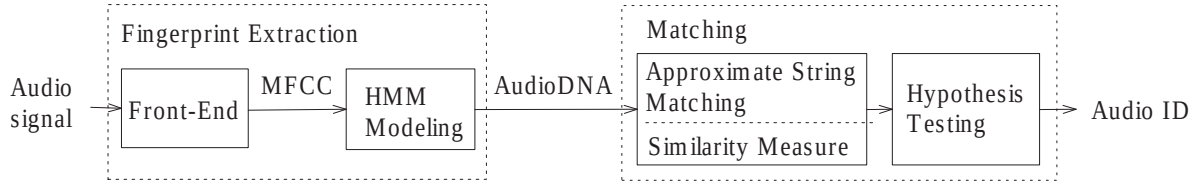


Figure 2.5: Overview of AudioDNA scheme

The matching method is based on exact matches of short subsequences of the query, which is subsequently validated via some time gap restrictions. Matches that do not satisfy the restrictions are rejected. The similarity S between AudioDNA sequences is defined as the percentage of the sum of time intervals $\Delta t_{equal}(i)$ for exact matching within a period of time Δt_{obs}:

$$S(\Delta t_{obs}) = \frac{\sum_{i=1}^{n} \Delta t_{equal}(i)}{\Delta t_{obs}}$$

In the defined time period Δt_{obs}, sequences with similarity higher than a predefined threshold are returned as matching results.

This method is extremely efficient and effective when query is with little noise distortion, compared with original audio. However, when severe distortions exist, it becomes difficult to obtain exact matches to short subsequences of the query, and thus the similarity between query and the original song becomes small. A false recognition is more likely to occur.

2.2 Our System

Our work target at improving accuracy and efficiency of audio fingerprinting systems subjected to distortions due to lossy transmission channel, source editing, and

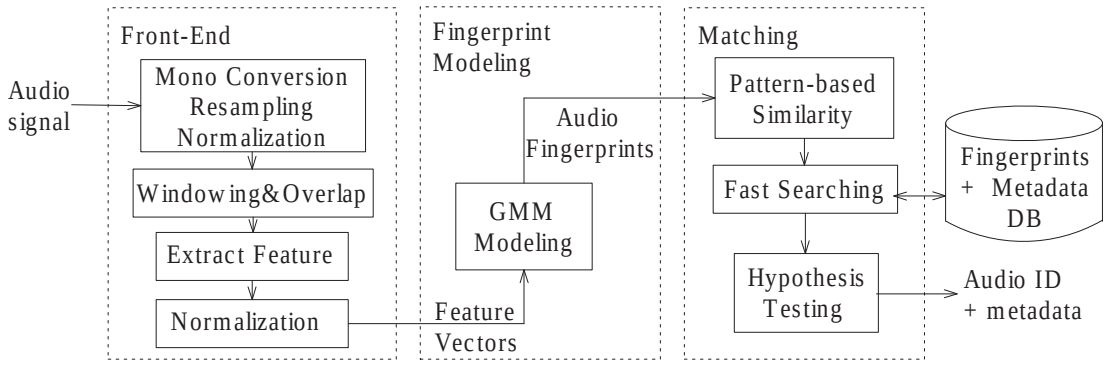


Figure 2.6: Overview of our system

background noise. We focus on three important modules of audio fingerprinting systems: feature extraction, fingerprint modeling, and matching. These three modules affect accuracy and efficiency of the whole system. Figure 2.6 is the framework of our system.

- In feature extraction, we study and compare several spectral features, including Mel-Frequency Cepstral Coefficient (MFCC), chroma spectrum, constant Q spectrum and product spectrum. Both chroma spectrum and constant Q spectrum express energy distribution related to the equal tempered scale in western music, making them superior in music signal analysis, such as key detection and chord recognition. MFCC is based on Mel scale filter-bank which mimics the human auditory's response. It has been highly frequently used in speaker/speech recognition and music modeling. Product spectrum takes advantage of the phase spectrum by using the product of magnitude spectrum and group delay function, and has shown effectiveness in robust speech recognition. However, its effect in music signal has not be studied yet. Therefore, we study its effect in our work. Although these features have been used in many music/speech applications, their performance in audio

fingerprinting are compared the first time. We compare the robustness of these features in the experiments. Since phase spectrum carries half of the information about the audio signal, product spectrum is more robust than the other three features which ignore the phase spectrum.

- In fingerprint modeling, we study the effect of GMM modeling in generating robust and concise audio fingerprints to facilitate both accuracy and efficiency of the system. Proper modeling methods can enhance the robustness of audio fingerprints subjected to noise distortions, reduce the storage space and speed up the matching process. GMM modeling has several advantages over other modeling methods in music-related applications because of its better precision and efficiency. It models the feature space globally and converts acoustic feature vectors into symbolic tokens (acoustic events) in a time-preserving way. First, the music database is trained using M Gaussian components with diagonal covariance matrices in an incremental procedure, which better describes the global distribution of acoustic feature space. Then, based on the trained Gaussian Mixture Model, acoustic feature vectors of music database and test dataset are all converted into symbolic tokens (acoustic events). Experimental results show the advantages of GMM modeling that it maintains high accuracy under severe noise distortions.

- In matching, we propose a Pattern Accumulative Similarity measure (PAS) and its search approaches. Based on the observation that similar audios have more short segments that match each other than that of dissimilar audios, PAS accumulates the similarity of two audios along the matching path, while diminishes the effect of unmatch. It better captures the similarity between

music data and is discriminative under distortions that may result in mismatches in both time and amplitude axes. Experimental results show that PAS has improvement in effectiveness and efficiency compared with Euclidean distance and DTW distance.

2.3 Summary

In this chapter, we first review related work of feature extraction, fingerprint modeling and matching three aspects because they are important modules that affect the accuracy and efficiency of the whole system. Then, we introduce five state-of-the-art systems, including Philips scheme, Shazam, RARE, AudioID and AudioDNA, and analyze their limitations. These systems represent the main techniques in audio fingerprinting systems, and cover all the core modules. Finally, we present the structure of our system and its advantages in effective and efficient audio fingerprinting when distortions exist in music signal. Specifically, we study and compare several spectral features, including Mel-Frequency Cepstral Coefficients, chroma spectrum, constant Q spectrum and product spectrum in feature extraction, study the effect of GMM modeling in fingerprint modeling to generate robust and concise audio fingerprints, and propose a pattern accumulative similarity measure which better captures the similarity between music data and is discriminative under several kinds of distortions.

Chapter 3

Feature Extraction

3.1 Introduction

Digital audio is represented as a sequence of discrete audio samples obtained by sampling and quantization on analog audio signal. However, these discrete samples in time domain can not be used directly in content-based audio analysis. Firstly, the amount of samples is usually huge, which incurs high computational cost. Secondly, the samples are highly correlated, resulting in data redundancy. Thirdly, the information contained in each sample is too small to be meaningful for human perception. Finally, these samples are quite sensitive to distortions, such as channel distortion and background noise. Therefore, it is necessary to extract acoustic features from digital audio in order to manipulate more meaningful information and to facilitate further processing.

As shown in Figure 1.1, front-end module converts an audio signal into acoustic features. It consists of four steps: pre-processing, windowing and overlapping, feature extraction, and post-processing. Figure 3.1 shows the steps for front-end processing. The rounded rectangles show the techniques and parameters used in

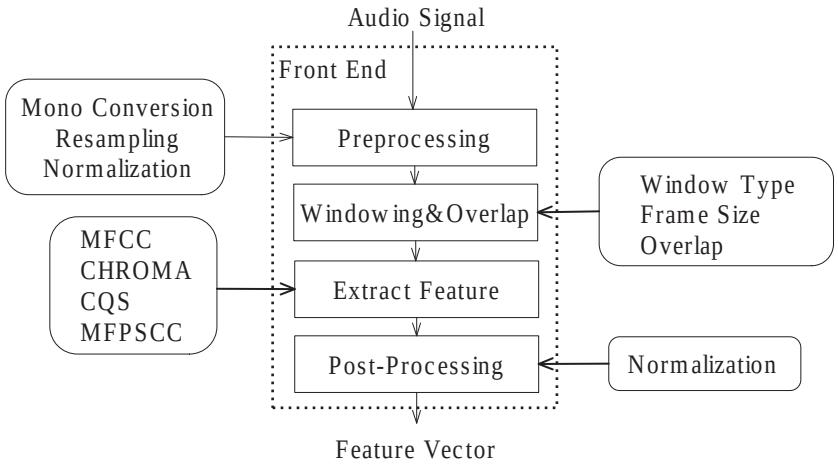


Figure 3.1: Steps for front-end processing

our implementations.

- **Pre-processing**

  The audio is converted to a general format, e.g., mono 16-bit PCM (Pulse-Code Modulation) with a fixed sampling rate of 22.05 kHz. Other types of processing like pre-emphasis and amplitude normalization can also be applied.

- **Windowing and overlapping**

  The signal is divided into frames of small size, typically 23 ms for speech signal and 46 ms or longer for music signal, under the assumption that the signal can be regarded as stationary over an interval of a few milliseconds. These frames can have overlaps. Window functions such as the Hamming window can be applied to each frame to attenuate the discontinuities at window edge [41]. In our implementation, 46 ms and 372 ms Hamming window with 50% overlap are used and compared in the experiments, for these parameters have been widely used in music signal processing [9, 21, 48, 51].

- **Feature extraction**

  Most of the acoustic features are extracted by performing time-frequency analysis, such as the STFT (Short-Time Fourier Transform). The frequency content can be represented as a magnitude spectrum that represents the energy distribution over frequency for the particular frame. Such a magnitude spectrum is usually viewed as a feature vector. Log magnitude spectrums of successive frames constitute a spectrogram. Although the magnitude spectrum can be used directly to represent audio signals, it contains lots of unimportant information, and the dimensionality of the feature vectors is high. It is better to use feature vectors of small dimensionality which are as informative as possible. Therefore, based on magnitude spectrum, a set of features that characterize the gross spectral shape are calculated, for instance, the Mel-frequency cepstral coefficients (MFCCs). Some features such as chroma spectrum and constant Q spectrum are specially designed to suit the equal tempered scale in western music, making them superior in music signal analysis. All these spectral features have been widely used in Computer Audition and Speech Recognition algorithms.

- **Post-processing**

  The feature vectors of each song, $\{c_t, t = 1, \ldots, T\}$, are normalized to follow the standard normal distribution by using the transformation of $\tilde{c}_{td} = (c_{td} - \mu_d)/\sigma_d$, where μ_d and σ_d are respectively mean and standard deviation of the d-th dimensional feature values of the song. Normalization can reduce the effects of small noise distortion and channel distortion, which is studied in the experiments.

The most distinct differences between existing audio fingerprinting systems are

due to the used time-frequency features. Therefore, feature extraction forms the major contents of this chapter. Related work about feature extraction is summarized in Section 2.1.1. Most of these acoustic features are extracted from spectral features.

Spectral features are based on short-time Fourier transform that generates two components: magnitude spectrum and phase spectrum. Existing features are mostly extracted from the magnitude spectrum, while the phase spectrum is discarded. The phase spectrum has been recently studied in human speech perception and automatic speech recognition [20, 42]. Product spectrum takes advantage of the phase spectrum by using the product of magnitude spectrum and group delay function (GDF), and has shown effectiveness in robust speech recognition [61]. In our work, we investigate the effectiveness of using the product spectrum in audio fingerprinting.

In the following sections, we will first introduce several spectral features, including magnitude spectrum, Mel-Frequency Cepstral Coefficients (MFCC), chroma spectrum, constant Q spectrum, and product spectrum. Their calculations are described in detail. Then, we compare these spectral features to show their similarities and differences.

3.2 Spectral Features

3.2.1 Magnitude Spectrum

Most of the acoustic features are based on the DFT (Discrete Fourier Transform) or more specifically the STFT (Short Time Fourier Transform). For efficient computation, the FFT (Fast Fourier Transform) is often used instead of the DFT. The STFT $X(n, k)$ of a signal $x(n)$ is a function of both time n and frequency k, which

can be calculated by [41]:

$$X(t,k) = \sum_{m=-\infty}^{\infty} x(n)w(n-t)e^{-j(2\pi/N)kn} \tag{3.1}$$

where $k = 0, ..., N-1$, $w(n)$ is the window function, commonly a hamming window or gaussian window, $x(n)$ is the input signal, and N is the size of the transform. The output $X(t,k)$ for any particular value of k is a frequency shifted, band-pass filtered version of the input.

$X(t,k)$ can be decomposed into

$$X(t,k) = |X(t,k)|e^{j\psi(t,k)} \tag{3.2}$$

where $|X(t,k)|$ is the short-time magnitude spectrum and $\psi(t,k) = \angle X(t,k)$ is the short-time phase spectrum. The STFT for a particular frequency k at particular time t is a complex number. For feature calculation, only magnitude of these complex numbers is retained.

Based on the STFT, spectral shape features which describe the shapes of magnitude spectrum $|X(t,k)|$ or power spectrum $|X(t,k)|^2$ of a signal frame are calculated. These features include centroid, spread, kurtosis, slope, roll-off frequency, flux (local spectral change), Mel-frequency cepstral coefficients (MFCCs), and so forth [54].

3.2.2 Mel-Frequency Cepstral Coefficients

Mel-frequency cepstral coefficients (MFCCs) [18] are perceptually motivated features that are based on the magnitude spectrum. After performing the STFT, the magnitude spectrum is mapped onto the Mel scale, using triangular overlapping

windows called Mel-frequency filter-bank. This results in FBEs (FilterBank Energies) which accumulate total energy within each band. Whereafter, in order to decorrelate the FBEs, a discrete cosine transform is performed on log FBEs. It transforms features from the log-spectral domain to the cepstral domain, where the size of the cepstral features is often less than that in the log-spectral domain. Mel scale reflects the human auditory perception, making MFCC robust to noise distortions [50]. MFCC has been widely used in various areas, such as speaker recognition, speech recognition, music/speech classification, and music modeling [11, 23, 37].

The MFCCs are computed in the following steps:

1. Compute the FFT spectrum of $x(n)$, denoted by $X(k)$.
2. Compute the power spectrum $|X(k)|^2$.
3. Apply a Mel-frequency filter-bank to $|X(k)|^2$ to get the filter bank energies.
4. Calculate DCT of log FBEs to get the MFCCs.

Figure 3.2 shows the calculation steps. More details about the calculation of MFCCs can be found in [46].

3.2.3 Chroma Spectrum

In the 1960's, Shepard [49] reported two distinct attributes of pitch perception, the *tone height* (octave number) and the *chroma* (pitch class). Based on these attributes, the chorma spectrum [57], also called the pitch class profile (PCP), is proposed in order to map the values of the magnitude spectrum to the 12-semitone pitch class. Usually, the chroma spectrum is a 12-dimension representation, corresponding to chroma scale. All notes are mapped to a single octave. The main

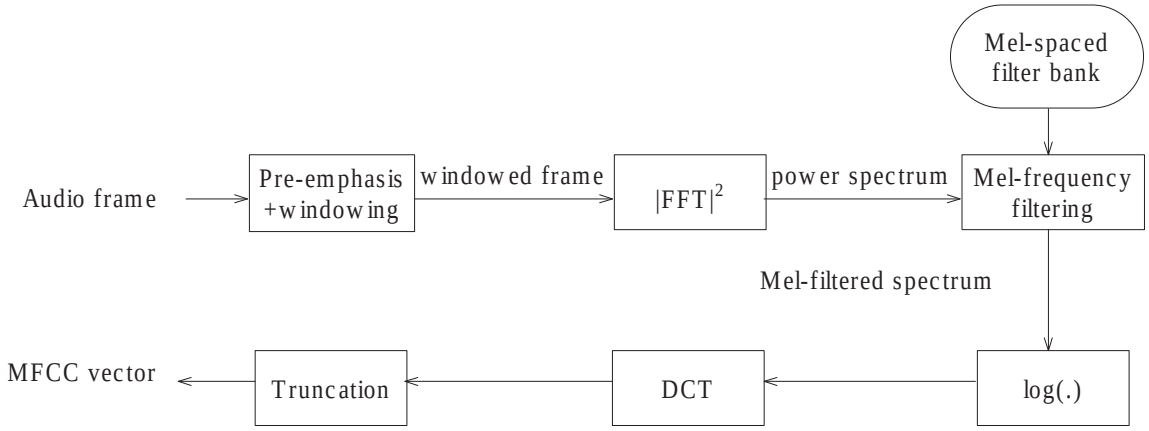


Figure 3.2: Steps for Mel-Frequency Cepstral Coefficients (MFCC)

concept of chroma spectrum is shown in Figure 3.3. A sequence of chroma spectrums constitute the chromagram. Chroma spectrum has been used in musical key extraction [44], chord recognition [34] and chorus detection [5, 25].

For chromagram $C = [\overline{x}_1, \overline{x}_2, ..., \overline{x}_n]$, $\overline{x}_i$ is a chroma spectrum, $0 \leq i \leq N$. $\overline{x}_i = [x_{i1}, x_{i2}, ..., x_{iD}]^T$, where $D = 12$ in most of the cases. D could also be 24, 36 in generalized versions.

Specifically, chroma spectrum can be computed from magnitude spectrum following the formula [13]:

$$X_{chroma}(\bar{k}) = \sum_{k:P(k)=\bar{k}} X(k) \tag{3.3}$$

where $X(k)$ denotes the magnitude spectrum of signal $x(n)$. k is the frequency index, $1 \leq k \leq \lceil (NFFT + 1)/2 \rceil$, where $NFFT$ is FFT length. The spectral warping between frequency index k in magnitude spectrum $X(k, n)$ and frequency

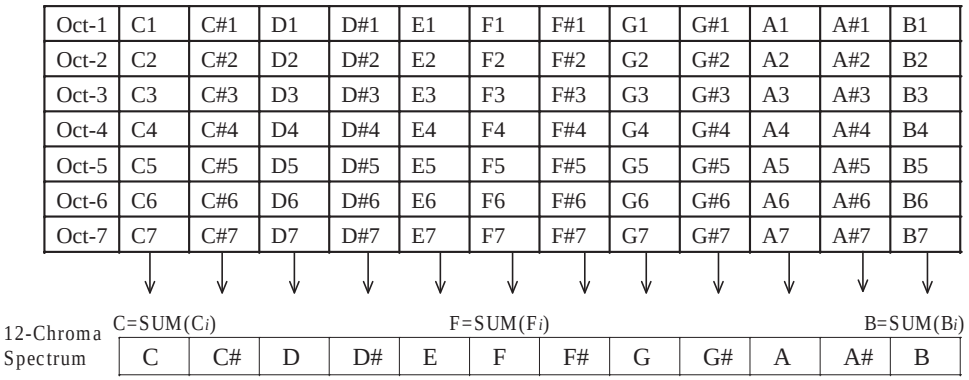


Figure 3.3: Overview of calculating a 12-dimensional chroma spectrum

index $\bar{k}$ in chroma spectrum $X_{chroma}(\bar{k})$ is

$$\bar{k} = P(k) = [D \cdot \log_2(k/NFFT \cdot f_s/f_0)] \mod D \tag{3.4}$$

where f_s is the sampling rate and f_0 is the frequency of a reference note in the standard tuning system.

3.2.4 Constant Q Spectrum

Constant Q spectrum (CQS) is derived by constant Q transform (CQT) [7] which uses a bank of filters whose center frequencies are geometrically spaced [39], as opposed to the linear spacing that occurs in the DFT. In modern western music, the frequencies of musical notes in the equal tempered scale are geometrically spaced [60]. As the frequency resolution can be set to match that of the equal tempered scale, CQT has considerable advantages for music signal analysis, such as pattern discovery [39] and key detection [62].

Given an minimum frequency f_0 that we are interested in computing the CQT,

the center frequencies of each subband can be obtained from

$$f_k = f_0 * 2^{k/b} \tag{3.5}$$

where b is the number of filters per octave (1 octave = 12 semitones), and $k = 0, 1, 2, ..., N * b$ (for N octaves). b is usually with a value of 12, 24 or 36.

The bandwidth of the k-th filter is

$$\Delta_k^{cq} = f_{k+1} - f_k = f_k(2^{1/b} - 1) \tag{3.6}$$

In CQT, the bandwidth Δ_k^{cq} varies proportionally to its center frequency f_k. Therefore, the constant ratio of frequency to resolution is

$$Q = f_k/\Delta_k^{cq} = f_k/(f_{k+1} - f_k) = (2^{1/b} - 1)^{-1} \tag{3.7}$$

The desired bandwidth $\Delta_k^{cq} = f_k/Q$ can be obtained by choosing a window of length

$$N_k = \lfloor f_s/\Delta_k^{cq} \rfloor = \lfloor Q f_s/f_k \rfloor \tag{3.8}$$

where f_s denotes the sampling rate.

The CQT is defined as

$$X(k) = \frac{1}{N_k} \sum_{n-0}^{N_k-1} W_{N_k}(n) x(n) e^{\frac{-j2\pi Qn}{N_k}} \tag{3.9}$$

where $X(k)$ represents the spectral energy of the k-th filter with the center frequency f_k, $x(n)$ is the time domain signal, and $W_{N_k}(n)$ is a window function, such as the hanning window, of length N_k.

CQT has two advantages. The first one is that by choosing f_0 and b appropriately, the center frequencies directly correspond to musical notes. For instance, if $b = 12$ and f_0 is the frequency of MIDI note m, f_k equals the frequency of MIDI note $m + k$.

Another advantage is that CQT has increasing time resolution at lower frequencies and higher frequency resolution at higher frequencies, which resembles the situation in our auditory system.

The chroma spectrum also has a similar idea as CQT and gives the spectral energy of 12 pitch classes. However, it is derived from DFT directly and ignores the differences between octaves. Therefore, it does not have finer resolution and is not as accurate as the features obtained by CQT.

3.2.5 Product Spectrum

Most of the acoustic features are mainly calculated from the magnitude spectrum whereas the phase spectrum is discarded. The product spectrum integrates the phase spectrum into feature extraction by multiplying the magnitude spectrum by group delay function (GDF) [61].

Given a frame of audio signal $\{x(n), n = 0, \ldots, N-1\}$, the Fourier transform is given by

$$X(\omega) = |X(\omega)|e^{j\theta(\omega)} , \tag{3.10}$$

where $|X(\omega)|$ is the magnitude spectrum and $\theta(\omega)$ is the phase spectrum.

Based on the phase spectrum, the GDF is defined as

$$\tau_p(\omega) = -\frac{d\theta(\omega)}{d\omega} . \tag{3.11}$$

Equation (3.11) can be simplified as follows [41]:

$$\tau_p(\omega) = -\mathrm{Im}\frac{d(\log\theta(\omega))}{d\omega} \tag{3.12}$$

$$= \frac{X_R(\omega)Y_R(\omega) + X_I(\omega)Y_I(\omega)}{|X(\omega)|^2} , \tag{3.13}$$

where $Y(\omega)$ is the Fourier transforms of $nx(n)$, and the subscripts R and I denote the real and imaginary parts.

The product spectrum is defined as the product of the power spectrum and the GDF as follows [61]:

$$Q(\omega) = |X(\omega)|^2\tau_p(\omega) \tag{3.14}$$

$$= X_R(\omega)Y_R(\omega) + X_I(\omega)Y_I(\omega) . \tag{3.15}$$

Therefore, the product spectrum is influenced by both the magnitude spectrum and the phase spectrum. Because the product spectrum may have negative values, it needs to be clipped by a nonnegative floor before calculating the dB values. Usually, a dynamic range threshold [45] is used, i.e., discarding the values below a certain threshold from the peak in the spectrum. Then Equation (3.19) can be rewritten as:

$$Q(\omega) = \max(X_R(\omega)Y_R(\omega) + X_I(\omega)Y_I(\omega), \rho) , \tag{3.16}$$

where

$$\rho = 10^{\sigma/10}\max(X_R(\omega)Y_R(\omega) + X_I(\omega)Y_I(\omega)) , \tag{3.17}$$

σ is the threshold in dB and is set to be $-60dB$ in our work.

Figure 3.4 shows a frame of audio signal, its power spectrum, group delay function, and product spectrum. The frame is a 46ms clip from a digital song recorded at the sampling rate of 22.05kHz. Before the Fourier transform, the audio frame is pre-emphasized by a filter of $H(z) = 1 - 0.97z^{-1}$ and multiplied with the Hamming window. The power spectrum can illustrate clearly the pitch harmonics and the spectral contour. However, there are only meaningless peaks and valleys in the GDF. It occurs due to the power spectrum in the denominator in Equation (3.12). The product spectrum enhances the region at the peaks of the power spectrum and has an envelope comparable to that of the power spectrum.

Based on product spectrum, Mel-frequency product-spectrum cepstral coefficients (MFPSCCs) [61] can be derived. The MFPSCCs are computed in the following steps:

1. Calculate the FFT spectrum of $x(n)$ and $nx(n)$. Denote them by $X(k)$ and $Y(k)$.

2. Calculate the product spectrum

$$Q(k) = \max(X_R(k)Y_R(k) + X_I(k)Y_I(k), \rho) \ , \tag{3.18}$$

   where

$$\rho = 10^{\sigma/10} \max(X_R(k)Y_R(k) + X_I(k)Y_I(k)) \ , \tag{3.19}$$

   σ is the threshold in dB.

3. Apply a Mel-frequency filter-bank to $Q(k)$ to get the filter bank energies.

4. Calculate DCT of log FBEs to get the MFPSCCs.

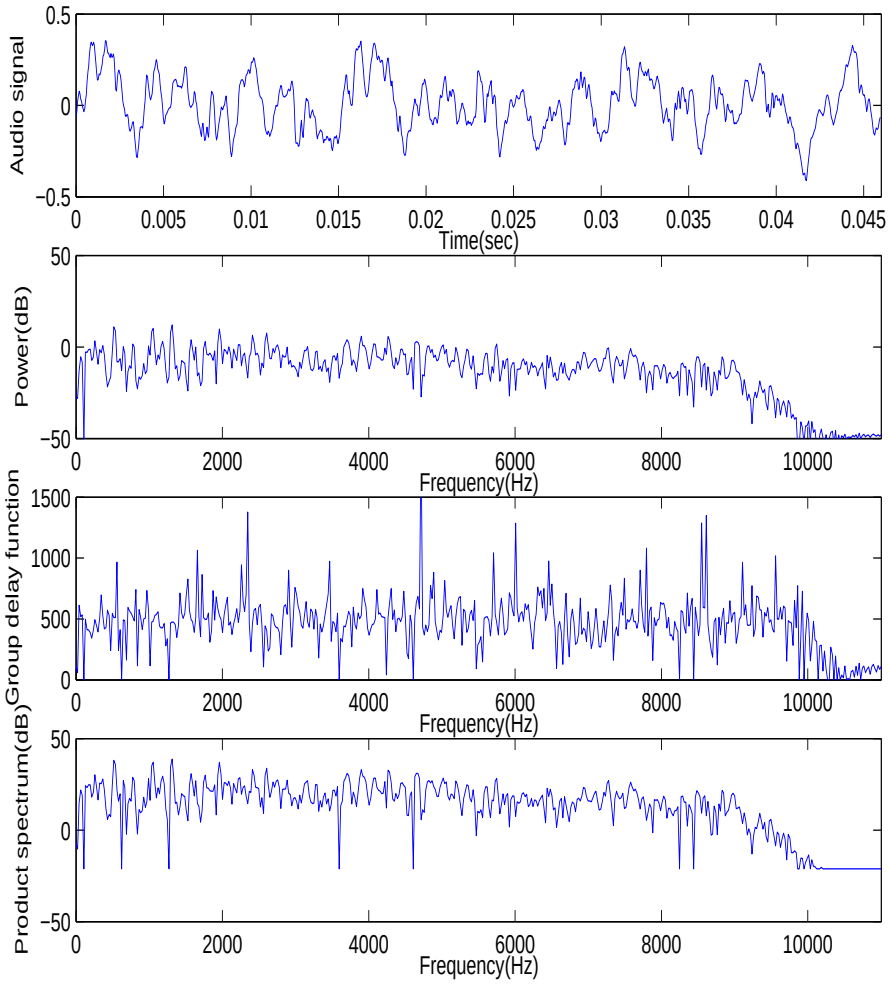


Figure 3.4: A frame of audio signal and its power spectrum (dB), group delay function, and product spectrum (dB)

3.3 Comparison

In the previous section, four acoustic features are introduced: MFCC, chroma spectrum, constant Q spectrum and product spectrum. Figure 3.5 shows an audio waveform and the corresponding acoustic features, drawn in Matlab (7.0). For waveform, the X axis is the time in second, and the Y axis is the amplitude. For all acoustic features, the X axis is the frame number. The Y axis for spectrogram is from 1 to 512 corresponding to frequencies up to 11.025 kHz, as a result of 1024-point FFT exclusive DC (Direct Current) component. The Y axis for CQS is from 1 to 60 with f_{min} = 55 Hz (A_1) and f_{max} = 1760 Hz (A_6). For both MFCC and MFPSCC, the Y axis is from 1 to 13, corresponding to 12 coefficients in addition to the value of normalized energy. For chromagram, the Y axis is from 1 to 12, corresponding to the 12-semitone pitch class.

The reason we study these features is that they have been widely used in music/speech area. Both chroma spectrum and constant Q spectrum are designed for music signal because they express energy distribution related to the equal tempered scale in western music, making them superior in music signal analysis, such as key detection and chord recognition. MFCC is based on Mel-frequency filter-bank which mimics the human auditory's response. It has been highly frequently used in speaker/speech recognition and music modeling. Product spectrum combines magnitude spectrum and phase spectrum and has shown effectiveness in robust speech recognition. However, its effect in music signal has not been studied yet. Therefore, we study its effect in our work.

These features are compared in the following four aspects:

- **DFT vs. CQT**

  One major difference between these features lies in the filter-bank. Constant

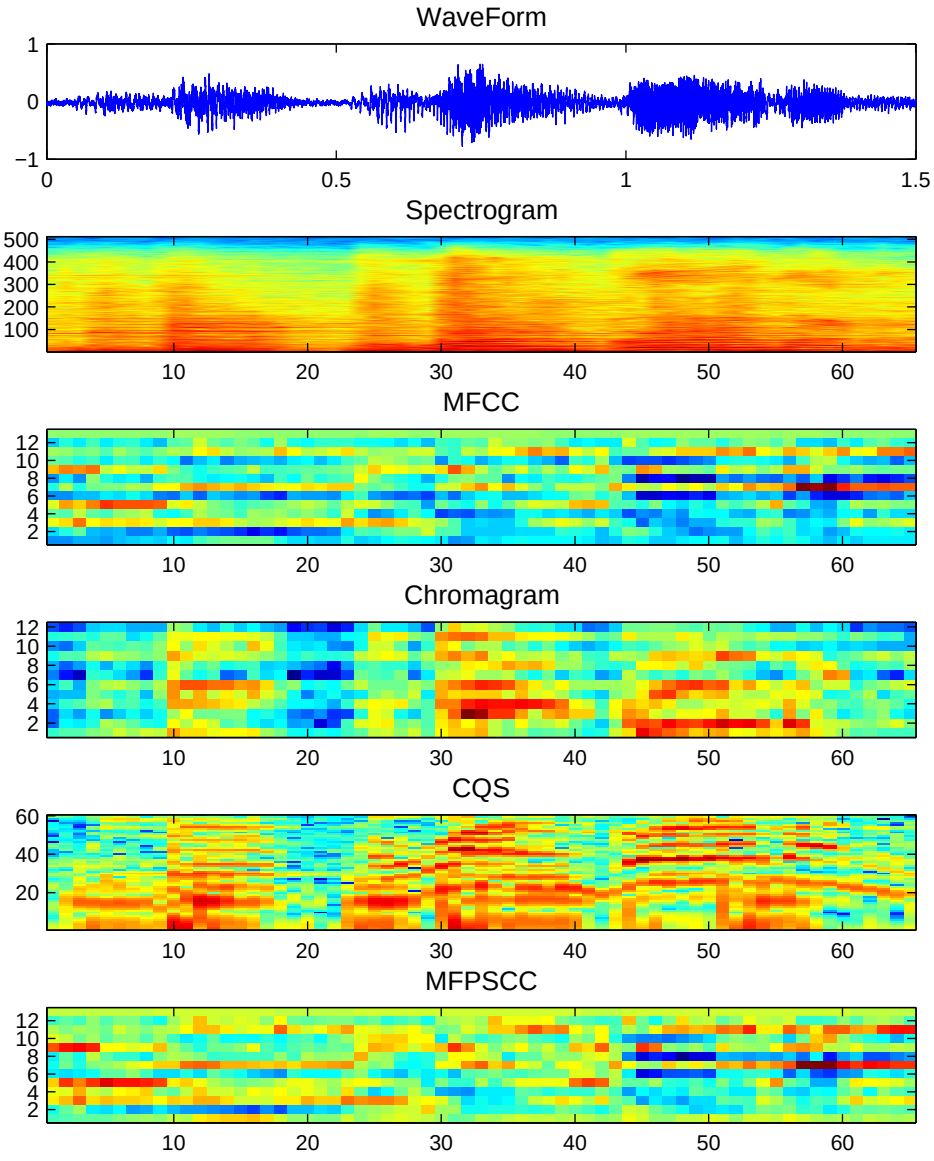


Figure 3.5: An audio waveform and its different acoustic feature representations

Q spectrum is extracted via constant Q transform (CQT). Constant Q filter-bank is a kind of auditory filter-bank which imitates the frequency resolution of human hearing. The filter-bank is geometrically spaced. MFCC, chroma spectrum and product spectrum are all derived from magnitude spectrum which are extracted via DFT or FFT. DFT filter-bank is linearly spaced. CQT has two advantages: 1) it combines a trade-off between time and frequency. As the bandwidth varies proportionally to its center frequency, it results in more frequency resolution at higher frequencies. The frequency resolution can be adjusted to match that of the equal tempered scale in western music. 2) Fewer filters are needed than conventional Fourier transform (FT). However, CQT is not as fast as FFT. Besides, it is not necessarily invertible, as is FT.

Based on the magnitude spectrum from FFT, MFCC uses Mel-frequency filter-bank to accumulate energies of each band which mimics the human auditory's response. MFPSCC is derived from the product spectrum via Mel-frequency filter-bank as well. Chroma spectrum uses a kind of filter-bank that are equally and symmetrically spaced in the geometric semi-tone pitch scale, and subsequently maps the energies to the 12-semitone chroma scale.

- **Speech vs. Music**

  Both MFCC and MFPSCC are designed for speech analysis. MFCC is a dominant feature used for speech recognition. It is used in music analysis because of its success in speech recognition. MFPSCC has shown effectiveness in speech recognition, but the effect in music analysis has not been studied.

Both chroma spectrum and constant Q spectrum are designed for music analysis. The filter-banks of both chroma spectrum and constant Q spectrum are highly related to the equal tempered scale in western music.

- **Spectral domain vs. Cepstral domain**

  Both MFCC and MFPSCC are features in the cepstral domain, but chroma spectrum and constant Q spectrum are in the spectral domain. In the calculations of MFCC and MFPSCC, DCT is performed in the last step to transform features from the log-spectral domain to the cepstral domain, which reduces the dimensionality of features and de-correlates the coefficients.

- **Magnitude spectrum vs. Phase spectrum**

  Product spectrum is different from the other three features in that it takes advantage of phase spectrum as well as magnitude spectrum, whereas the other features ignore phase spectrum.

3.4 Summary

Feature extraction is the basis for all content-based music information retrieval and is the core step of front-end processing. In this chapter, we focus on the extraction of spectral features. First, we briefly introduce the four steps of front-end processing and the importance of feature extraction. Then several spectral features and their calculations are described in detail. Specifically, we have studied MFCC, chroma spectrum, constant Q spectrum and product spectrum. Finally, we compare these features in four aspects to show their similarities and differences. Although these features have been used in many music/speech applications, their performance in audio fingerprinting are compared the first time. We evaluate their performance in

the experiments.

Chapter 4

Fingerprint Modeling

4.1 Introduction

The fingerprint modeling module usually receives a sequence of feature vectors passed from the front-end. After exploring redundancies in successive frames in time, inside a song and across the whole database, it further reduces the fingerprints into more concise representations. This may result in three advantages: first, the signal will become more robust to noise distortions because proper modeling methods can reduce the effect of noise addition. Second, the storage space is saved as only compact representations are stored in the database. Third, the speed of matching could be improved.

As summarized in Section 2.1.2, existing approaches for fingerprint modeling can be classified into time-unpreserving and time-preserving approaches. Generally speaking, time-preserving modeling is better because time information is an important factor in music. Research shows that temporal information of audio signals plays a crucial role in music perception [26]. The same set of notes will result in absolutely different music if their arrangements in time are different. And these

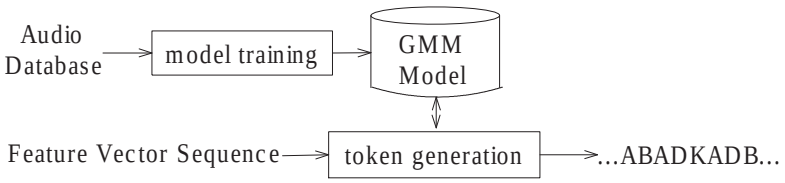


Figure 4.1: Steps for fingerprint modeling

differences can be easily perceived by human. Therefore, we choose time-preserving modeling in our work.

In time-preserving modeling, acoustic feature vectors can be regarded as multivariate time series (MTS). We adopt two methods to avoid direct computation of the similarity between MTSs. One is to model feature vectors into several acoustic events and encoded as symbolic tokens by using modeling methods such as Gaussian Mixture Models (GMM) and Vector Quantization (VQ). In this way, fingerprints of an audio are represented as a string. Figure 4.1 illustrates the steps of fingerprint modeling using GMM. First, a GMM is trained for the music database by using the Expectation Maximization (EM) algorithm, which better describes the distribution of acoustic feature space. Then, based on the trained GMM, each feature vector sequence is converted into a string of tokens. The other is to model fingerprints of an audio into a time series, by using the dimensionality reduction methods, such as Principal Component Analysis (PCA). We will show in the experiments that GMM has advantages over other modeling approaches.

In the following, we first introduce in detail GMM modeling, including theory of GMM, training process, and token sequence generation. Then, the advantages of GMM will be explained and compared with three modeling approaches.

4.2 GMM Modeling

In this section we present the GMM modeling approach, which aims to convert a feature vector sequence to a token sequence. The symbolic tokens denote the Gaussian components in the GMM. For example, assuming a GMM is composed of M Gaussian components, we may construct a set of symbolic tokens as $\{1, 2, ..., M\}$. The GMM modeling of feature vectors mainly has two advantages: 1) the obtained symbol sequences can be compared using string matching approaches, which have lower computational costs than calculation of distance between feature vectors, and 2) clustering feature vectors to discrete symbols enhances robustness of the system against acoustic distortions.

4.2.1 Gaussian Mixture Model

GMM(Gaussian Mixture Model) is a standard technique used for clustering with soft assignment of the data sample x to clusters [31].

A multivariate Gaussian probability density function is defined as:

$$N(x|\nu, \Sigma) = (\frac{1}{2\pi})^{D/2}|\Sigma|^{-1/2} \exp(-\frac{1}{2}(x-\nu)^T\Sigma^{-1}(x-\nu)) \tag{4.1}$$

where x is an observational feature vector, ν is a mean vector, Σ is a covariance matrix and D is the dimensionality of the feature vector. In consideration of computational complexity, Σ is usually defined as a diagonal covariance matrix $\Sigma = \{\sigma_1^2, ...\sigma_D^2\}$.

A GMM is a mixture of M Gaussians. The probability density for the observable

data x is the weighted sum of each Gaussian component:

$$p(x|\Phi) = \sum_{m=1}^{M} c_m p(x|m, \Phi) = \sum_{m=1}^{M} c_m N(x|\nu_m, \Sigma_m) \tag{4.2}$$

where $1 \leq m \leq M$, $0 < c_m \leq 1$, and $\sum_{m=1}^{M} c_m = 1$. Φ are the parameters that need to be estimated per GMM: $\Phi = \{\nu_m, \Sigma_m, c_m; m = 1...M\}$.

The optimal estimate for Φ maximizes the likelihood that the observations $X = \{x_1, ..., x_N\}$ are generated by the GMM, where N is the number of observations. The standard measure used is the log-likelihood which is computed as:

$$L(X|\Phi) = \log p(X|\Phi) = \log \prod_n p(x_n|\Phi) = \sum_n \log p(x_n|\Phi) \tag{4.3}$$

To find good estimates for Φ, a standard approach is to use the Expectation Maximization (EM) algorithm. The EM algorithm is iterative and converges relatively fast after a few iterations [19]. The initial estimates can be completely random, or can be computed by using other clustering algorithms such as k-means.

The EM algorithm consists of two steps. First, the expectation is computed, which is the probability (expectation) that an observation x_n is generated by the m-th component. Second, the parameters in Φ are recomputed to maximize the expectations.

The expectation step is:

$$\gamma_n(m) = p(m|x_n, \Phi) = \frac{p(x_n|m, \Phi)c_m}{p(x_n|\Phi)} = \frac{N(x_n|\nu_m, \Sigma_m)c_m}{\sum_{m=1}^{M} N(x_n|\nu_m, \Sigma_m)c_m} \tag{4.4}$$

The maximization step is:

$$\hat{c}_m = \frac{\sum_{n=1}^{N} \gamma_n(m)}{N} \tag{4.5}$$

$$\hat{\nu}_m = \frac{\sum_{n=1}^{N} \gamma_n(m) x_n}{\sum_{n=1}^{N} \gamma_n(m)} \tag{4.6}$$

$$\hat{\Sigma}_m = \frac{\sum_{n=1}^{N} \gamma_n(m)(x_n - \hat{\nu}_m)(x_n - \hat{\nu}_m)^T}{\sum_{n=1}^{N} \gamma_n(m)} \tag{4.7}$$

GMM uses a family of Gaussian probability density functions to partition the feature space into clusters. As the probability density functions can overlap, GMM performs a soft assignment of data sample to clusters.

4.2.2 Training Process

We train a GMM using a database consisting of 1000 songs [54]. The GMM is designed to be composed of M Gaussian components with diagonal covariance matrices. An incremental training procedure is adopted, which includes the following steps:

Step 1. Initialization In the beginning the GMM is designed only consisting of one Gaussian, where values of the mean vector and the variance vector are respectively set to that of the global mean and variance over the whole database.

Step 2. Increasing the number of Gaussian components The Gaussian component that has the maximum weight value $\hat{c}_m$ is selected and split to two Gaussian components. The weight values of these two generated Gaussian components are half of the original weight value. The two new mean vectors are disturbances from the original mean vector: $\nu_{+/-} = \nu \pm 0.2 \cdot \sigma_m$. The new

variances are copied from the original one.

Step 3. Re-estimate parameters GMM parameters are re-estimated via Equations (4.5) - (4.7). Several EM iterations can be performed.

Step 4. Repeat Gaussian-increase and re-estimation Repeat Step 2 and Step 3 until desired number of Gaussian components is achieved.

4.2.3 Token Sequence Generation

After the GMM is trained, we may use it to convert a feature vector sequence $X = \{x_1, ..., x_T\}$ to a token sequence composed of Gaussian labels $M = \{m_1, ..., m_T\}$. Each frame t is labeled with the top-1 Gaussian component as follows

$$m_t = \arg\max_m p(m|x_t, \Phi) \ . \tag{4.8}$$

Figure 4.2 shows the waveform, MFCC , and its corresponding GMM token sequence of a song clip, drawn in Matlab (7.0). The number of Gaussian components in the GMM is set to 64. MFCCs are extracted using a 46 ms window with 50% overlap.

4.3 Advantages

In this section, we will compare GMM with three modeling approaches to show its advantages.

- **Principal Component Analysis**

  Principal Component Analysis (PCA) [14] is a widely used method to reduce the dimensionality of the dataset. It examines the variance structure in

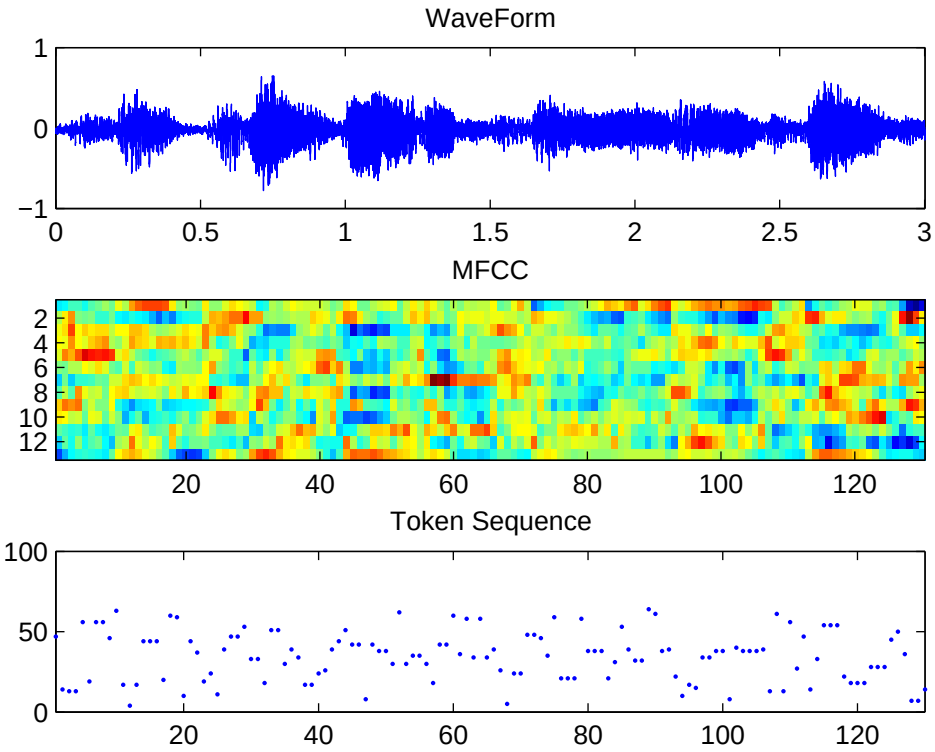


Figure 4.2: An example of token sequence generation

the dataset and determines the directions along which the data exhibit high variance. The first principal component corresponds to the eigenvector with the largest eigenvalue of the dataset's covariance matrix and has the largest variance. The second component corresponds to the eigenvector with the second largest eigenvalue and has the second largest variance, and so forth. All principal components are orthogonal to each other. In the following, we will briefly introduce how to transform a multivariate time series into a univariate time series by using PCA.

Let the dataset contains N d-dimensional feature vectors. First, we calculate a covariance matrix A by using the following equation:

$$A = \begin{pmatrix} \sum_t x_{1t}x_{1t} & \sum_t x_{1t}x_{2t} & ... & \sum_t x_{1t}x_{dt} \\ \sum_t x_{2t}x_{1t} & \sum_t x_{2t}x_{2t} & ... & \sum_t x_{2t}x_{dt} \\ ... & ... & ... & ... \\ \sum_t x_{dt}x_{1t} & \sum_t x_{dt}x_{2t} & ... & \sum_t x_{dt}x_{dt} \end{pmatrix}$$

Each eigenvalue λ_i of matrix A is ordered as $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d$. The eigenvector is represented as $[e_{1\lambda_i}, e_{2\lambda_i}, ..., e_{d\lambda_i}]$. Then, the i-th principal component pc_{t,λ_i} is calculated as:

$$pc_{t,\lambda_i} = e_{1\lambda_i}(x_{1t} - \bar{x}_1) + e_{2\lambda_i}(x_{2t} - \bar{x}_2) + ... + e_{d\lambda_i}(x_{dt} - \bar{x}_d)$$

where $\bar{x}_i$ is the mean of x_i.

Finally, we use the first principal component to effectively transform MTSs into univariate time series data. For each MTS T_m, we obtain univariate time

series data T as follows:

$$\begin{aligned} T &= x_1, ..., x_t, ..., x_N \\ x_t &= e_{1\lambda_1}(x_{1t} - \bar{x}_1) + e_{2\lambda_1}(x_{2t} - \bar{x}_2) + ... + e_{d\lambda_1}(x_{dt} - \bar{x}_d) \end{aligned}$$

Comparison: The density modeled by PCA is relatively simple in that it is unimodal and has fairly restricted parametric forms (Gaussian). However, it is not suitable to model data with more complex structure such as clusters. GMM considers mixture models, and therefore it is more suitable to model feature vector space.

- **Vector Quantization**

  Vector Quantization (VQ) [31] is an efficient source-coding technique which is widely used in data compression. Given a d-dimensional vector x whose coefficients x_k are real-valued, continuous-amplitude random variables ($1 \leq k \leq d$), VQ maps (quantizes) x to another d-dimensional discrete-amplitude vector z. Typically, z is a vector from a finite set $Z = \{z_j | 1 \leq j \leq M\}$, where the set Z is referred to as the codebook, M is the size of the codebook, and z_j is the j-th codeword.

  The VQ is realized in two steps: 1) Design a codebook by training dataset with the LBG (Linde-Buzo-Gray) algorithm. The d-dimensional space is partitioned into M regions or cells C_i, and each cell C_i is associated with a codeword vector z_i, $1 \leq i \leq M$. 2) Map (quantize) the vector x to codeword z_i which minimizes the quantization error:

$$q(x) = z_i, \quad iff \quad i = \arg\min_k d(x, z_k)$$

Euclidean distance is usually used as the distortion measure $d(x, z_k)$ between x and z_k.

Comparison: VQ partitions the vector space into separate regions, which performs hard assignment of data samples to clusters. Since the partitions are based on some distance measure regardless of the probability distributions of original data, the errors in partitions could potentially destroy the original structure of data. Compared with VQ, GMM uses a family of Gaussian probability density functions to partition the vector space. The probability density functions can have overlap, meaning that GMM performs a soft assignment of data samples to clusters. Since the distribution properties of the data are taken into account, GMM better models the vector space.

- **Hidden Markov Model**

  Hidden Markov Model (HMM) [31] is a very powerful statistical method of characterizing the observed data samples of a multivariate time series, which has been successfully used in areas such as speech recognition, statistical language modeling and machine translation. Given a sequence of observable feature vectors, HMM finds a sequence of hidden states from the observable data. First, the HMM parameters are trained using Baum-Welch algorithm. Then, each hidden state sequence of test dataset is generated using Viterbi algorithm.

  Comparison: Compared with HMM, GMM is less complex and more efficient. A GMM can be viewed as a single-state HMM with a Gaussian mixture density. It is used to globally model acoustic feature vector space. GMM has a number of advantages.

  1. GMMs are conceptually less complex than HMMs, consisting of only

one state and one output distribution function.

2. The training dataset is represented by exactly one Gaussian mixture model, and only the parameters of the output distribution function need to be estimated. This leads to significantly shorter training time.

3. HMM training is based on labeled data. For example, in HMM-based speech recognition system, the training speech is labeled with a phonetic based transcription and the phoneme specific frames are uniquely assigned to one of the HMM phoneme models. However, in music, no explicit 'phonemes' exist, and they need to be inferred from the dataset via unsupervised training or labeled manually. On the contrary, GMM does not use any phonetic knowledge, and can be trained in an unsupervised way.

4.4 Summary

Proper modeling methods can enhance the robustness of audio fingerprints subjected to noise distortions, reduce the storage space and speed up the matching process. In this chapter, we introduce fingerprint modeling by GMM in detail. GMM has been used to model music in some work without preserving the time information, where a GMM is trained for each song and the song with the highest likelihood is regarded as a match. However, time is a key factor in music, and therefore it should not be ignored. In our work, GMM is used to model the feature space globally and convert acoustic feature vectors into symbolic tokens (acoustic events) in a time-preserving way. First, the motivations of using GMM to model robust and concise audio fingerprints are explained. Then, the theory of GMM is presented, followed by steps for mixture model training and token sequence

generation. Moreover, we compare GMM with PCA, VQ and HMM to show its advantages. Fingerprint modeling results in robust and concise fingerprints which are ready to the matching process.

Chapter 5

Matching

5.1 Introduction

Fingerprint matching is fulfilled by comparing fingerprints of the query song with fingerprints of the songs in the database. If a credible similarity between a pair of fingerprint sequences exists, the query is considered to be found as the song in the database. As shown in Figure 5.1, the matching component consists of two modules: database look-up and hypothesis testing. The database look-up module defines the similarity measure between audio fingerprints and performs fast search in the fingerprints database to return a set of matching songs. Usually, indexing or pruning strategies are used to speed up the search. The hypothesis testing module is used to judge whether the identification is correct by comparing the similarity score with a threshold.

Similarity measure is very important in the matching process as it affects effectiveness as well as efficiency of the system. Section 2.1.3 summarizes the related work about similarity measure.

When audio is represented as a feature vector sequence, Euclidean and DTW

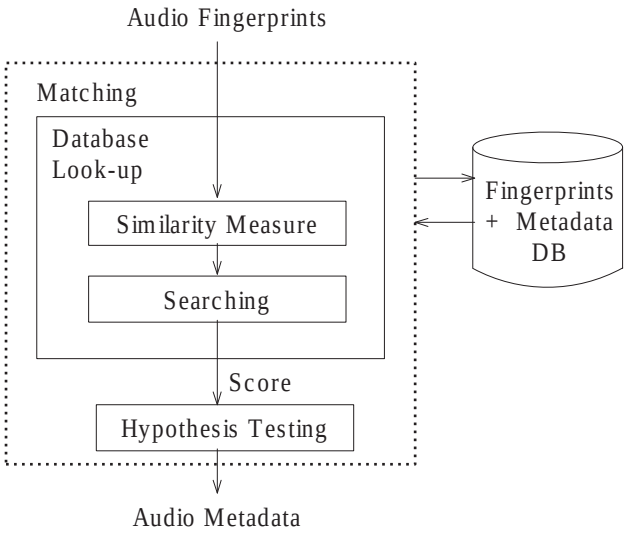


Figure 5.1: Steps for fingerprint matching

distances become candidate distance measures. Euclidean distance is very sensitive to distortions in time axis and amplitude axis. DTW can handle local time shifting and scaling, but is sensitive to amplitude distortions as well. In audio fingerprinting, queries are often affected by channel distortion incurred in transmission, source distortions due to audio editing, or noise addition. Some frames may be corrupted or even lost, which results in distortions in both time and amplitude axes. To solve this problem, we define a new similarity measure which is based on matches of local patterns. From observations, we notice that two similar sequences have more short patterns that can match each other than those of dissimilar sequences. Inspired from time series and string searching approaches [15, 33] which are based on matches of local patterns, we define a pattern accumulative similarity measure that better captures the similarity between distorted music and original music. The new similarity measure is based on accumulative similarity of matching patterns, rather than the gap distances of matching patterns [15].

The new similarity measure can be generalized to string representation as well. When GMM modeling is used to generate robust and concise audio fingerprints, the audio search is transformed into an approximate string matching problem. String matching has been intensively studied for genomic and proteomic sequence [4, 33, 36]. A general search strategy for homologous sequence is based on finding perfect or near perfect seed (x-mer, subsequence of length x) matches, i.e., the Blast [4]. Although the proposed similarity measure shares a similar concept by finding matching patterns, the search methods for genomic data cannot be applied directly here, due to three reasons. Firstly, the similarity between genomic sequences is based on certain hypotheses in genetics. For example, genes that share a high sequence identity or similarity support the hypothesis that they share a common ancestor and are therefore homologous [43]. But the similarity between music is more based on human perception. For instance, a different version of a song which is recorded in background noise environment is regarded the same as the original song, although they may have acoustic features of great differences. Secondly, homologous sequences share a large number of perfect match x-mers [4]. In our work, although we also assumes similar music share a large number of patterns, it does not always hold under distortions. For example, most of background noises have continuous frequency spectrum and are additive in nature, making the spectrum of clean song distorted. The distortions in frequency also continue in the time domain, making few exact match x-mers if x is relatively large, or many false matches if x is small. Thirdly, both the alphabet size and the value of x are different. The alphabet size is 20 for amino acids and 4 for nucleotides, and x is typically 8-16 for nucleotide comparisons and 3-7 for amino acid comparisons [33]. The alphabet size of audio fingerprints modeled by GMM is adaptive, i.e., 32 or 64, and x, the length of pattern, can be adjusted freely. Due to the above differences,

we will adopt a k-Radius Nearest Neighbor (k-RNN) search in the search process for the new similarity measure. Given a pattern, the k-RNN search returns a set of neighbors to the pattern, regarded as matches.

In this chapter, we first introduce the new similarity measure. Then, the search strategy and parameters are discussed. Finally, we introduce how to perform hypothesis testing.

5.2 Pattern Accumulative Similarity

Pattern Accumulative Similarity (PAS) is based on the observation that similar songs have more short segments that match each other than that of dissimilar songs. By using a fixed size window sliding on sequences, short segments, called patterns here, can be extracted.

A short pattern p from a time sequence S is defined as $p = (\lambda_{pos}, \lambda_{amp})$, with λ_{pos} and λ_{amp} representing the position of p in S and the amplitude values of p, respectively. The distance of two short patterns p_1 and p_2 can be measured as

$$D_p(p_1, p_2) = F(p_1.\lambda_{amp}, p_2.\lambda_{amp}) \tag{5.1}$$

where F is a distance function. When $D_p(p_1, p_2) < \epsilon$, we say a matching pattern m is formed from pattern p_1 and p_2. A matching pattern m between pattern q of Q and pattern s of S is shown in Figure 5.2. The matching pattern m is described as

$$m = (m.x, m.y, D_p, \lambda_{tscl}, \lambda_{ascl}) \tag{5.2}$$

where $m.x$ and $m.y$ are projections of m on x axis and y axis. D_p is the distance between q and s. λ_{tscl} and λ_{ascl} are respectively relative scaling in time

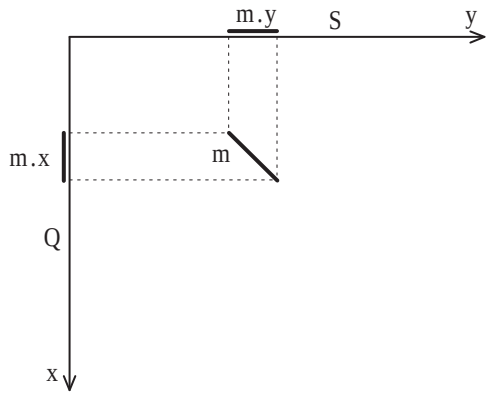


Figure 5.2: An example of matching pattern in a matching matrix

and amplitude of q with respect to s. If Q is similar to S, the number of matching patterns could be large.

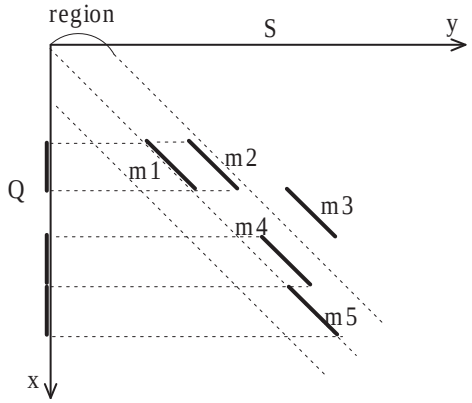


Figure 5.3: An example of pattern accumulative similarity between two sequences

The matching patterns are stored in a list with key value equals to $m.y - m.x$. All matching patterns in the same list share the same key and are sorted according to $m.x$. All the lists are sorted according to the key values. As shown in Figure 5.3, m_1 and m_5 have a same key and lie on the same diagonal.

Based on the diagonal with key k, we define the similarity between Q and S as

$$sim(k) = \bigcup_{key(m_i) \in [k-\delta, k+\delta]} Projection(m_i) \tag{5.3}$$

which means the union of projections of all matching patterns within a certain region with δ deviation from k.

$$Projection(m_i) = m.x * \omega_1 * \omega_2 * \omega_3 * \omega_4 \tag{5.4}$$

where ω_1, ω_2, ω_3, ω_4 are weights corresponding to D_p, λ_{tscl}, λ_{ascl} and δ, $w_i \in [0, 1]$. For example, we can set

$$\begin{aligned} \omega_1 &= \mu_1(1 - \frac{D_p}{\epsilon}) \\ \omega_2 &= \mu_2(1 - |\lambda_{tscl}|) \\ \omega_3 &= \mu_3(1 - |\lambda_{ascl}|) \\ \omega_4 &= \mu_4(1 - |\Delta|/(\delta + 1)) \end{aligned}$$

$\Delta \in [-\delta, \delta]$ is the deviation from diagonal k, and $\mu_i \in [0, 1]$. μ_i can be set to emphasize certain distortions. In the simplest form, all $\mu_i = 1$.

Finally, the PAS between sequence Q and S is:

$$PAS\_sim(Q, S) = \max_t sim(t) \tag{5.5}$$

where $t \in [0, S]$. For example, in Figure 5.3, there are only four matching patterns m_1, m_2, m_4 and m_5 in the region. So $sim(0) = \bigcup Projection(m_i)$, $i \in 1, 2, 4, 5$. Projections of m_1 and m_2 have overlap. If $Projection(m_1) \geq Projection(m_2)$, $sim(0) = Projection(m_1) + Projection(m_4) + Projection(m_5)$.

Therefore, $PAS\_sim(Q, S) = sim(0)$.

5.3 Search Process

Before searching the query in database, we need to extract short patterns from database sequences. Sliding window with width w and sliding step *step* is used. There is a trade-off between accuracy and efficiency regarding *step*. Larger *step* results in fewer patterns, and thus better efficiency. However, accuracy may decrease due to time-shifting between patterns.

In the search process, patterns are extracted from each query sequence, using disjoint windows with width w. For each pattern, range query is performed to get patterns that are within distance ϵ from the query pattern. Since the parameter ϵ is affected by dataset, we can use kNN query instead, which finds k nearest neighbors that "match" the query pattern. δ is set according to applications. In applications with severe distortions, δ can be set large value in order to obtain high accuracy, while incurring extra computational cost. In applications without distortions, we can set $\delta = 0$.

When the audio is modeled as a feature vector sequence, it can be viewed as a multivariate time series. One possible method is to reduce it into a univariate time series using the dimensionality reduction approaches, such as PCA [28]. PCA is used in [52] to discover motif in multivariate time series. When large parts of the query remain the same as the original music, for example, in the case of distortion due to partial source editing, PCA transformations on both query and original music will not affect the match between the same part, which means patterns which match each other before transformation can still match after transformation. Therefore, it will not affect the accuracy of PAS. Indexing methods for

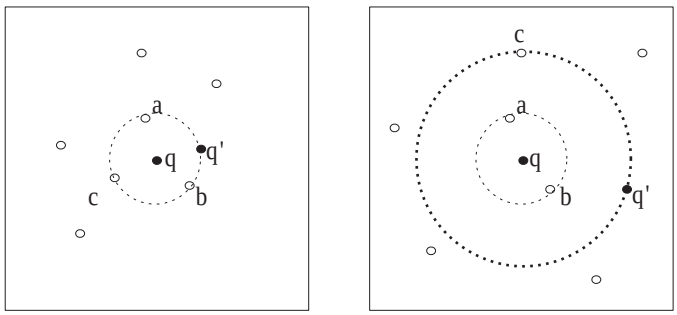
1-dimensional time series can be applied on these transformed patterns to speed up the search process. In some applications when the whole query is distorted by background noise, PCA will decrease recognition accuracy. In the transformed space, a distorted pattern may become more similar to another mismatched pattern than to the original pattern. In such case, noise resistent modeling, like GMM, can be used instead.

GMM modeling transforms the audio search into approximate string matching problem. We adopt a k-Radius Nearest Neighbor (k-RNN) search in our work, which returns a set of neighbors to the pattern, regarded as matching patterns.

Definition: Given a dataset D, a distance function $d(a, b)$, and an integer k, the k-RNN query returns a set of data which are within the k-th distance to the query (inclusive), if all distances to the query are sorted in ascending order.

Compared with kNN and range query, k-RNN is more suitable here, since true matching patterns may not be close in distance due to noise distortions. kNN has the difficulty for a suitable choice of k. When pattern length is small, many data may share a same short distance to the query pattern. kNN randomly returns k data as matches, which may miss the true match. When pattern length is large, the distant true match may be missed. Figure 5.4 illustrates this problem. The dark point q' is a true match to the query q. When pattern length is small, as shown in (a), a, b, c and q' all share the nearest distance to q. However, if $k = 3$, kNN search may return a, b and c but miss q'. When pattern length is large, as shown in (b), q' may be distant to q due to distortions. Then, if $k = 3$, kNN search may return a, b and c but miss q' again. For range query, the choice of *radius* may incur problem as well. Small *radius* may return empty result set, while large *radius* may return all patterns, which is computationally expensive. k-RNN can avoid the problem by choosing suitable k, depending on pattern length and the estimated

degree of distortions. In Figure 5.4 (a), as pattern length is small, we can set $r = 1$ which returns all the patterns closest to q. In Figure 5.4 (b), as pattern length is large and possible distortions exist, we can set $r = 2$ to return all patterns a, b, c, and q'. Since the true matches are not missed, the similarity between true matching sequences will not decrease. Although the returned set contains false matches, these matching patterns may contribute to different sequences, reducing the possibility of false hit.



(a) (b)

Figure 5.4: An example of different search methods

5.4 Hypothesis Testing

The problem of fingerprint matching can be formulated as a hypothesis testing problem that tests two complementary hypotheses, namely the null hypothesis H_0 and the alternative hypothesis H_1 as follows:

H_0: F^q is similar to F^i.

H_1: F^q is NOT similar to F^i.

According to Neyman-Pearson Lemma [30], under some conditions, the optimal solution to the above testing is based on a likelihood ratio testing as follows:

$$\ell = \log p(F^q|H_0) - \log p(F^q|H_1) \underset{H_1}{\overset{H_0}{\gtrless}} \tau \ , \tag{5.6}$$

where τ is the critical decision threshold. The logarithmic likelihood $\log p(F^q|H_h)$ can be generalized to other similarity measures which are consistent with the hypothesis testing. In our case, Equation (5.6) is rewritten as

$$\ell = S(F^q, F^i) - S(F^q, \bar{F}^i) \underset{H_1}{\overset{H_0}{\gtrless}} \tau \ , \tag{5.7}$$

where $\bar{F}^i$ denotes the fingerprints of songs excluding the i-th song. An open issue is how to calculate $S(F^q, \bar{F}^i)$. We adopt an N-best approach that has been widely used in automatic speech recognition [32]. For a query F^q, we collect its top-N recognition scores $\{S_i, i = 1, \ldots, N\}$. The similarity of the alternative hypothesis is computed using the $N-1$ scores as follows:

$$S(F^q, \bar{F}^i) = \frac{1}{\eta} \log \left[\frac{1}{N-1} \sum_{m=2}^{N} e^{S_m \eta} \right] \ , \tag{5.8}$$

where η is a positive number. When η approaches ∞, the term in the bracket becomes $\max_{m=2}^{N} S_m$. By varying the value of η and N, one can take all the competing songs into consideration, according to the individual significance.

By adjusting τ, a receiver operating characteristic (ROC) can be found, which reflects the relationship between false alarm rate P_{FA} and identification rate P_{IR}. The false alarm rate P_{FA} is the probability to declare different songs as similar. The identification rate P_{IR} is the probability to declare right songs to be similar. The system is expected to achieve high P_{IR} with low P_{FA}.

5.5 Summary

In this chapter, we propose a pattern accumulative similarity measure, PAS, which better captures the similarity between music data under signal distortions. First, we introduce the modules and steps of the matching process. The matching process defines the similarity measure between audio fingerprints and performs fast search which returns a result set. Hypothesis testing is subsequently used to judge the credibility of the result set. Then, after analyzing the motivations behind PAS, we introduce its definition and the search approaches in detail. Based on short matching patterns, PAS accumulates the similarity of two audios along the matching path, while diminishes the effect of unmatch. It is more suitable to measure similarities between audios with distortions. To increase accuracy, we adopt a k-radius nearest neighbor (k-RNN) search in the search process. Finally, the theory of hypothesis testing is introduced.

Chapter 6

Experiments

In this chapter, we will describe the experimental results of the proposed methods in previous chapters. Specifically, we will first present the music database used in the experiments. Then, we study the robustness of acoustic features by testing the effects of normalization and frame length and comparing the receiver operating characteristic (ROC) performance between different spectral features. Furthermore, we evaluate the effectiveness and efficiency of PAS and GMM modeling. Finally, we compare our method with an existing audio fingerprinting method and test the system performances with respect to different query lengths.

6.1 Music Database

The database we used in experiments includes 1000 songs grouped by 10 genres [54]: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, and rock. These songs, recorded at 22.05 kHz sampling rate, are framed using a 46 ms or 372 ms analysis window with 50% overlap. Acoustic features described in Chapter 3 are used as music representations in the experiments.

6.2 Evaluation on Acoustic Feature

In this group of experiments, we study and compare the robustness of spectral features in music identification, under different kinds of noise conditions. The details of spectral features are introduced in Chapter 3. Each frame of the songs is converted to a feature vector, consisting of 12 coefficients in addition to the value of normalized energy for both MFPSCC and MFCC, 12 coefficients for chroma spectrum, and 72 coefficients for constant Q spectrum. For both chroma spectrum and constant Q spectrum, we set $f_{min} = 55$ Hz (A_1) and $f_{max} = 3520$ Hz (A_7). Since 72-coefficient vectors are not practical in computation, we collapse the constant Q spectrum to a 12-coefficient representation, the same as in [12].

The query set consists of 7200 songs generated from 400 clean songs, in order to test the robustness of acoustic features under different types and different degrees of noise distortions. In the clean set, 300 songs are randomly selected from the database which form the in-set test dataset, and 100 songs are from outside of the database which form the out-of-set test dataset. The out-of-set test dataset contains songs of various genre. There are 18 different distortions applied to each song of the clean dataset. The distortions are generated by adding three types of noises, white noise, babble noise and airport noise, respectively, with 6 different signal-to-noise ratios (SNR), -5dB, 0dB, 5dB, 10dB, 15dB and 20dB.

We compare the recognition accuracy and the receiver operating characteristic (ROC) of the features.

Recognition accuracy is the percentage of times the correct song is found as the top match, measured over the in-set test dataset. The calculation is based on one nearest neighbor classification (1NN). For each query in the test set, we derive its title from its nearest neighbor in the music database. If the derived

title is the same as the original title of the query, we get a hit; Otherwise, we get a miss.

ROC is mentioned in Section 5.4. Fingerprint matching can be formulated as a hypothesis testing in which two types of errors are concerned: the false-alarm rate P_{FA} and the identification rate P_{IR}. The ROC curve which plots P_{IR} against P_{FA} is used in order to compare acoustic features fairly. Both in-set and out-of-set test datasets are used to calculate ROC curve.

6.2.1 Effect of Normalization

First, we test the effect of feature normalization. In the post-processing step of the front-end module, feature vectors of each song are normalized to follow the standard normal distribution. We use cosine distance as a similarity measure between two feature vectors. The recognition accuracies of unnormalized and normalized MFCC features are shown in Table 6.1. The results show that normalized test data achieve significant improvement in accuracy, especially for severe noise distortions. It proves the importance of normalization. Normalization converts features to the same baseline and scale, which reduces the effects of small noise distortion and channel distortion. The experiments in later sections are all based on normalized features.

Table 6.1: Comparison of recognition accuracy (in %) between unnormalized and normalized data

Noise	Normalized	SNR(dB)					
		20	15	10	5	0	-5
White	No	86.00	64.33	44.33	25.33	16.67	8.67
	Yes	99.00	99.00	98.67	98.67	98.67	98.67
Babble	No	100	99.67	98.67	96.33	88.00	58.00
	Yes	100	100	100	100	100	100
Airport	No	100	99.67	98.67	96.33	92.33	71.00
	Yes	100	100	100	100	100	100

In the table, normalization can achieve 100% recognition accuracy under babble and airport noise distortions at all SNR levels. Three factors contribute to such performance: 1) the size of song database, 2) the similarities between songs in the database, and 3) the query length. If larger database is used, or the similarities between songs in the database are higher, or shorter queries are used, all the recognition accuracy values may decrease. White noise is a more severe distortion compared with babble and airport noise distortions, because white noise has a power spectrum of equal power in any band, corrupting the whole spectrum of clean signal. Therefore, the corresponding recognition accuracies are lower, and the effect of normalization is obvious.

6.2.2 Effect of Frame Length

Based on the assumption that signals can be regarded as stationary over an interval of a few milliseconds, audio signals are usually divided into frames of small size before analyzing and processing. Therefore, in most of the content-based music retrieval, frame length affects the performance. In this experiment, we compare the performance of short frame, 46 ms, and long frame, 372 ms, because these frame lengths are typical in music signal processing [9, 21, 48, 51]. Table 6.2 shows the recognition accuracy of MFCC, with 46 ms and 372 ms analysis window. 46 ms frame achieves better performance than 372 ms frame. The differences are obvious for white noise distortions. For babble and airport noise distortions, 372 ms has already achieved 100% accuracy when SNR is above 0 dB, due to the three factors analyzed in Section 6.2.1. Therefore, no improvement can be obtained when 46 ms frame is used. However, we can still see the differences when SNR is -5dB.

Table 6.2: Comparison of recognition accuracy (in %) between different frame lengths

Noise	frame(ms)	SNR(dB)					
		20	15	10	5	0	-5
White	46	99.00	99.00	98.67	98.67	98.67	98.67
	372	98.33	98.33	98.33	98.33	98.33	98.33
Babble	46	100	100	100	100	100	100
	372	100	100	100	100	100	99.33
Airport	46	100	100	100	100	100	100
	372	100	100	100	100	100	99.00

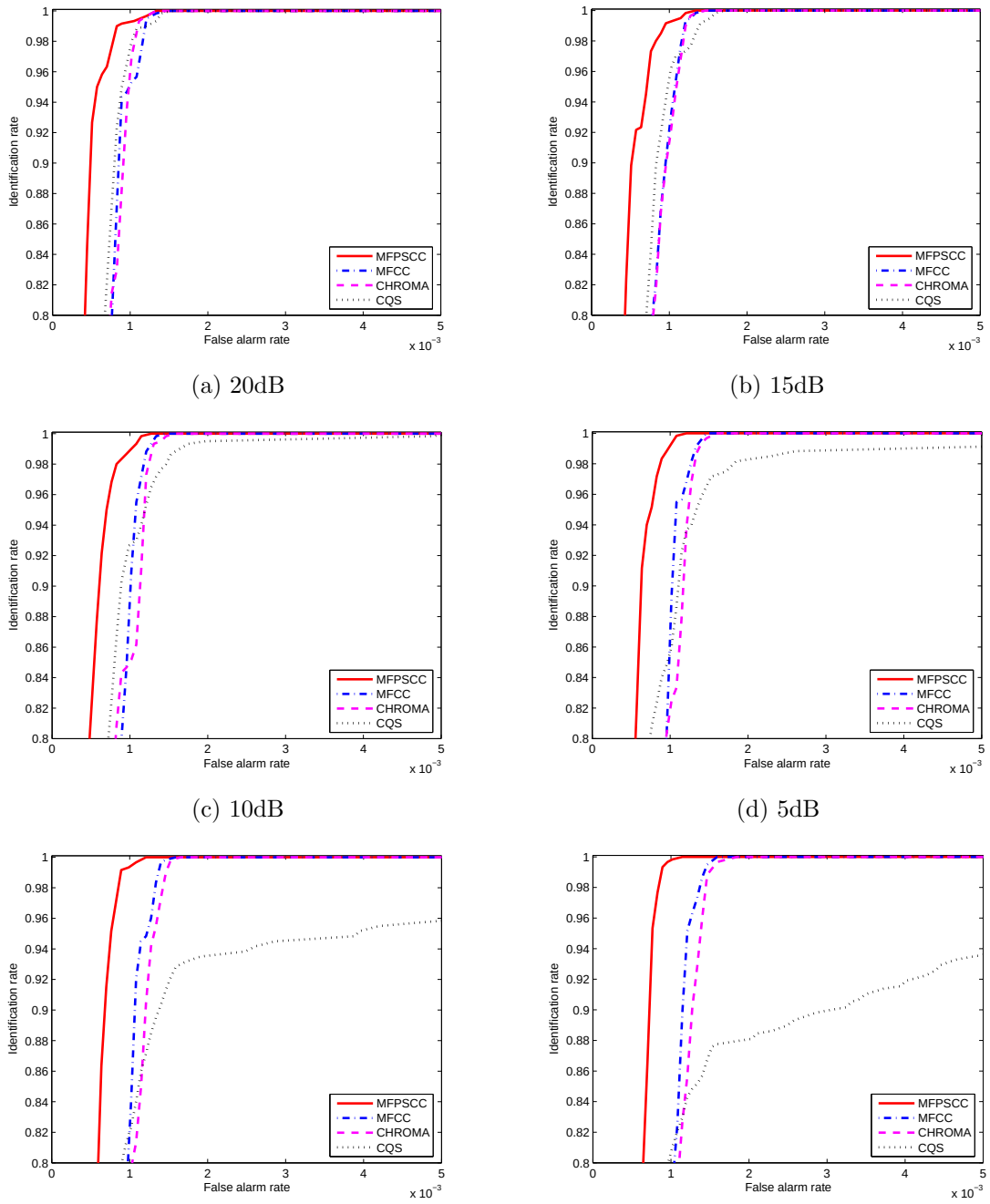
6.2.3 Robustness of Acoustic Features

In this section, we compare the robustness of four acoustic features: Mel-Frequency Cepstral Coefficients (MFCC), chroma spectrum (CHROMA), constant Q spectrum (CQS), and product spectrum (MFPSCC). 46 ms frame length is used here as it shows better performance.

Figure 6.1, 6.2 and 6.3 show the ROC curve of the four features under white, babble and airport distortions, respectively. The false alarm rate P_{FA} is the probability to declare different songs as similar. The identification rate P_{IR} is the probability to declare right songs to be similar. Acoustic features which are more robust have higher P_{IR} with a same fixed P_{FA}. It is shown that MFPSCC achieves better performance in all cases. MFCC and CHROMA have similar performance when noise distortions are slight. But when noise distortions become more severe, MFCC is better than CHROMA. CQS is better than MFCC when noise distortions are slight. However, it is quite sensitive to noise distortions. When noise distortions become more severe, CQS degenerates greatly. Table 6.3 shows the overall identification rate with a fixed false alarm rate 0.1%.

Table 6.3: Identification rate (in %) with a fixed false alarm rate 0.1%

Feature	MFPSCC	MFCC	CHROMA	CQS
P_{IR}	92.09	82.47	81.75	84.72



(a) 20dB

(b) 15dB

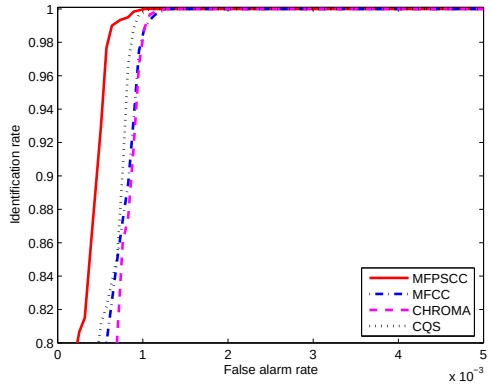
(c) 10dB

(d) 5dB

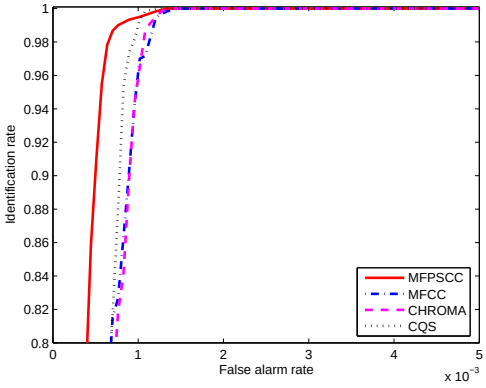
(e) 0dB

(f) -5dB

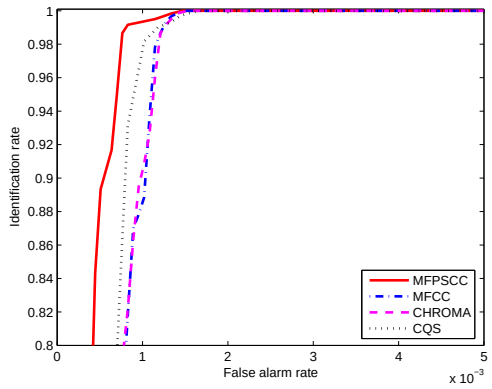
Figure 6.1: Receiver operating characteristic (ROC) comparison between spectral features under white noise



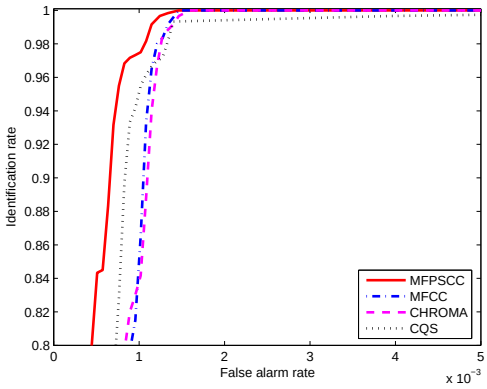
(a) 20dB



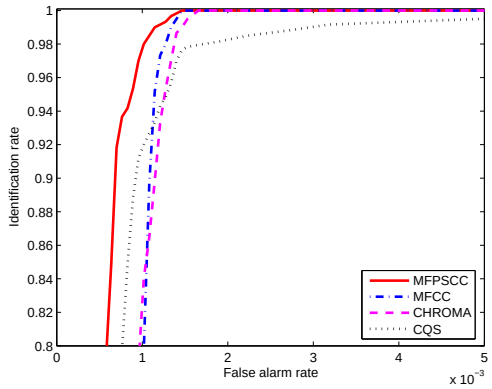
(b) 15dB



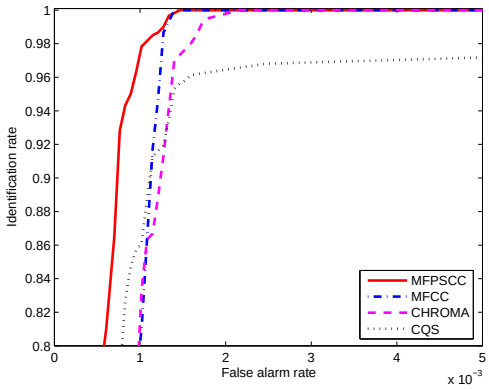
(c) 10dB



(d) 5dB

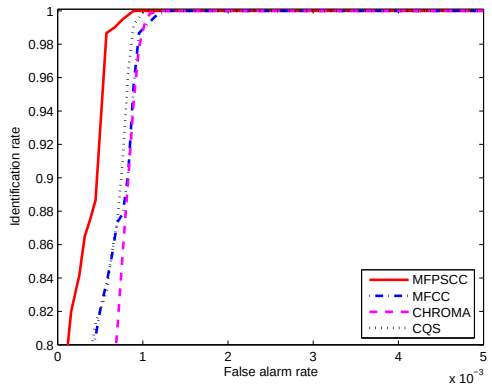


(e) 0dB

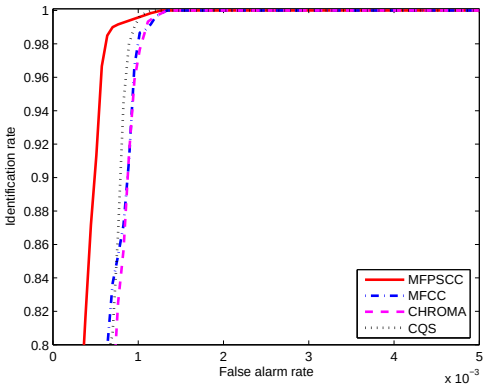


(f) -5dB

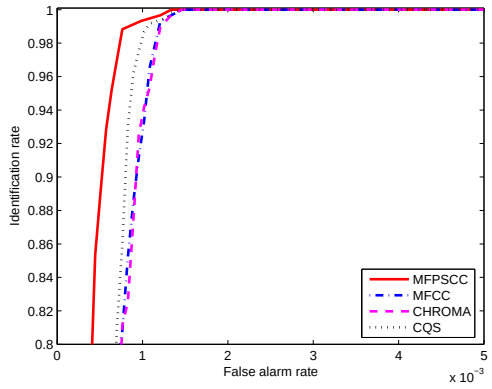
Figure 6.2: Receiver operating characteristic (ROC) comparison between spectral features under babble noise



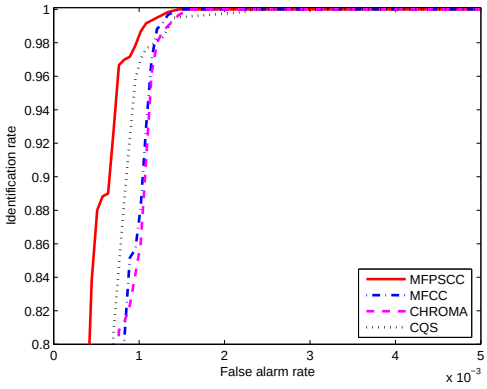
(a) 20dB



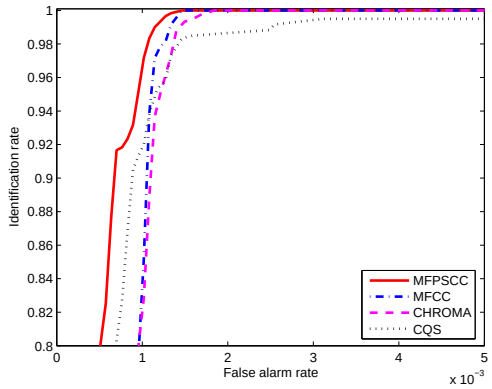
(b) 15dB



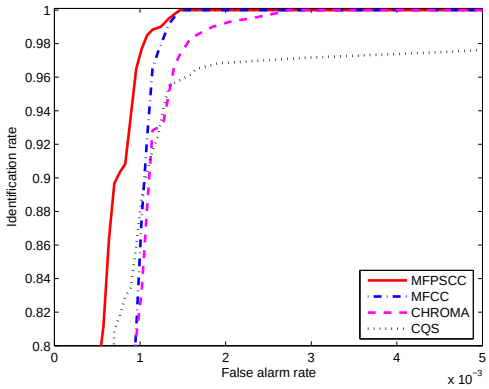
(c) 10dB



(d) 5dB



(e) 0dB



(f) -5dB

Figure 6.3: Receiver operating characteristic (ROC) comparison between spectral features under airport noise

The results prove that MFPSCC is more robust than the other three features because it combines magnitude spectrum and phase spectrum. Phase spectrum carries half of the information about the audio signal, as seen from Formula (3.2), making it useful in improving the robustness of acoustic features. Besides, MFCC is more robust to noise distortions than CHROMA and CQS. One possible reason lies in the filter-bank. MFCC uses Mel-frequency filter-bank which mimics the human auditory's response. However, filter-banks of both CHROMA and CQS are highly related to equal tempered scale in western music.

6.3 Evaluation on Similarity Measure

In the following two experiments, we evaluate the effectiveness and efficiency of the proposed similarity measure, PAS. The details of similarity measure are introduced in Section 5.2. The query set is 100 songs randomly selected from the database. The recognition accuracy is evaluated based on the average accuracy over 100 queries. In order to avoid direct computation between feature vectors and improve the efficiency, PCA is performed on the database and the query set to reduce the dimensionality of feature vectors into one dimension. For PAS, 10 nearest neighbors of each query pattern are retrieved by sequential scan or indexing.

Firstly, we compare the effectiveness and efficiency of PAS with Euclidean and DTW distance when channel distortion occurs. For each query, we randomly delete some frames. The overall deletion is from 1% to 10% of the query length, since we assume that the maximum data loss through the lossy transmission channel is 10% of the query. For DTW distance, there is a tradeoff between accuracy and efficiency with respect to the warping width r. When r increases, accuracy increases as well, but efficiency decreases. As the maximum data loss is 10% in the

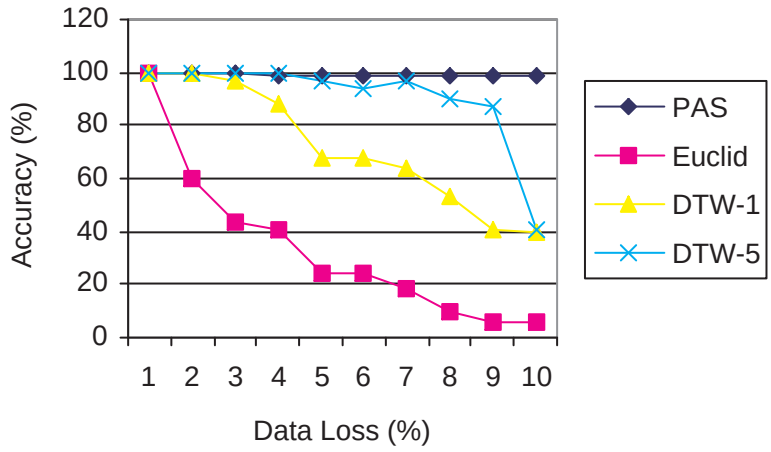


Figure 6.4: Recognition accuracy comparison of similarity measures under distortions due to lossy transmission channel

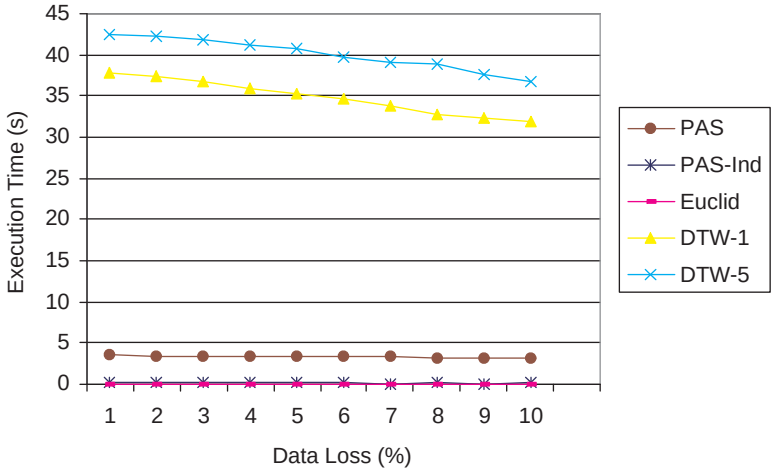


Figure 6.5: Efficiency comparison of similarity measures under distortions due to lossy transmission channel

experiment, we set $r = 1\% * |query|$ and $r = 5\% * |query|$. For PAS, the pattern length is set to $1\% * |query|$. Figure 6.4 shows that PAS can maintain accuracy above 99% whereas Euclidean decreases in accuracy sharply as the amount of data loss increases. DTW-1 and DTW-5 represent DTW with r equals to 1% and 5% of the query length, respectively. Larger warping width achieves better accuracy but worse efficiency. However, both methods cannot beat PAS. Figure 6.5 compares the efficiency of these similarity measures. PAS obviously beats DTW in execution time. PAS-Ind builds VA-file indexing on query patterns, and the execution time approaches to that of the Euclidean distance. These experiments confirm that PAS is a suitable similarity measure when distortions exist in both time and amplitude axes, because it takes into account time gaps and amplitude differences in the short pattern matching.

Secondly, we compare the accuracy of PAS, Euclidean distance and DTW distance when parts of the audio are edited. Short pieces of human speech with lengths of 10% to 50% of the query length are added to each query audio. As shown in Figure 6.6, PAS can maintain high accuracy because it accumulates the similarity of short patterns of two audios along the matching path, while diminish the effect of unmatch. Based on the assumption that large portion of patterns remain the same after partly editing the audio, PAS well captures the similarities between the matching parts. However, as Euclidean distance accumulates amplitude differences, it is sensitive to amplitude distortions. Therefore, its accuracy decreases as the edited portion becomes larger. DTW shows the worst performance. DTW may mismatch the time axis to achieve minimum accumulative amplitude differences, although no distortion in time axis exists.

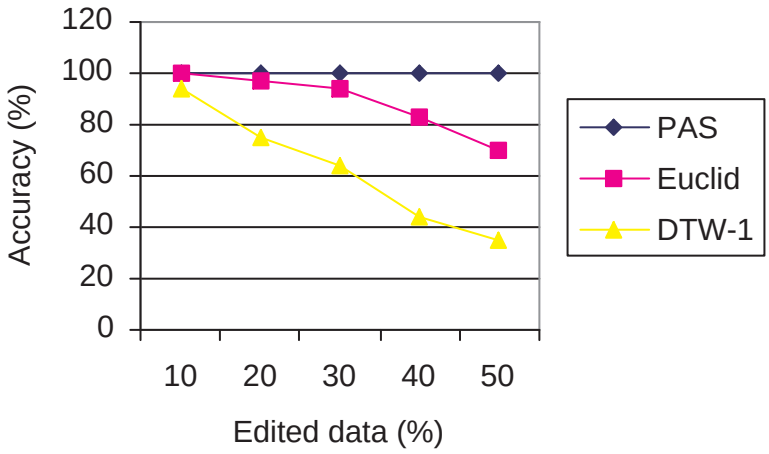


Figure 6.6: Recognition accuracy comparison of similarity measures under distortions due to source editing

6.4 Evaluation on Fingerprint Modeling

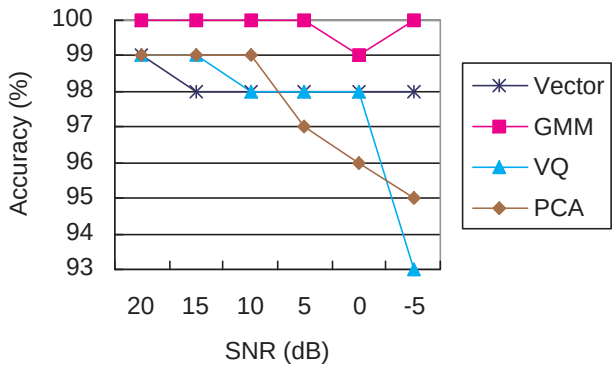
In this section, we evaluate the effect of GMM modeling, compared with VQ and PCA. The technical details are introduced in Section 4.2. The purpose of modeling is to gain robustness against distortions, reduce the disk space and the memory requirements, and be benefit for the subsequent matching process regarding convenience and efficiency. The query set is 100 songs the same as last section, with white noises of 6 different SNR levels added to each song. 64-component GMM is trained. The codebook size for VQ is also 64.

The effect of fingerprint modeling methods are compared under different distance measures. In Figure 6.7 (a), Euclidean distance is used for feature vector sequence and PCA sequence, and Hamming distance is used for token sequence. In Figure 6.7 (b), DTW distance and Euclidean distance are used. In Figure 6.7 (c), PAS is used. The results show that GMM modeling gains robustness against

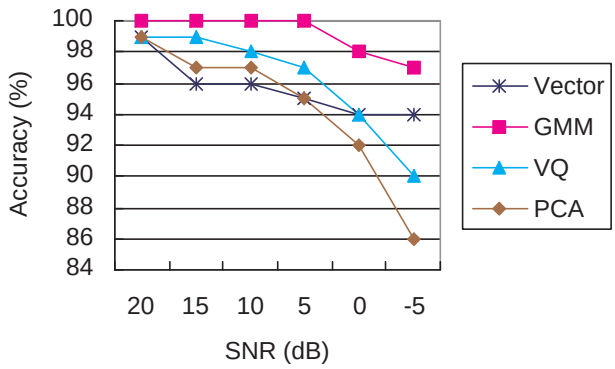
background noise distortions. It achieves accuracy improvement at every SNR level, compared with directly using feature vectors. It is because GMM globally models the feature vector space into clusters and performs a soft assignment of vectors to clusters. The effect of noise distortion is reduced in the processes of statistical modeling and soft assignment. VQ can achieve good accuracies when noise distortions are slight. However, its accuracies are lower than that of directly using feature vectors when noise distortions get severe. It is because VQ partitions the vector space into separate regions and performs a hard assignment of vectors to clusters. Vector with severe noise distortions is not likely to be classified into the same cell as the original vector. PCA makes the query even more sensitive to noise distortions since it models the vector space based on simple unimodal density. Therefore, the accuracies drop when SNR is below 5dB. The effect of GMM modeling is more obvious at -5dB, compared with VQ and PCA, confirming that GMM is more suitable to model audio fingerprints under severe noise distortions. Besides, compared with directly using feature vector, GMM modeling reduces the disk space of fingerprints database to 6% and the query process time to 14%, which will facilitate the matching process.

6.5 System Performance

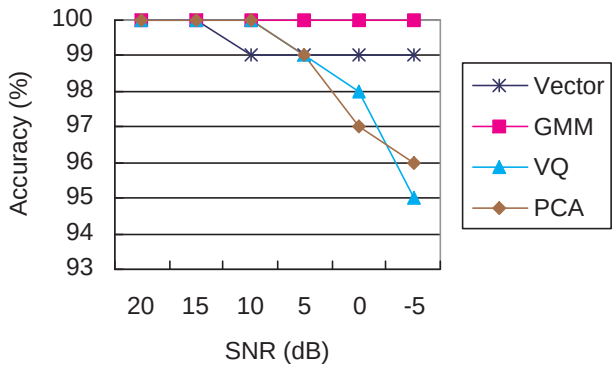
In this section, we present the performance of our system for queries of short audio clips under noise distortions. The queries are 100 5-second audio clips recorded with babble noise distortions of different SNR levels. The performance of our system is compared with that of AudioDNA. AudioDNA is introduced is Section 2.1.4. The differences between our method and AudioDNA are as follows: 1) product spectrum is used in our method while MFCC is used in AudioDNA, and 2) short



(a)



(b)



(c)

Figure 6.7: Accuracy comparison of fingerprint modeling methods

pattern matching of our method is based on k-RNN while AudioDNA is based on exact search of subsequence. Although AudioDNA also depends on short pattern matching, it finds exact matches to the query patterns, which can be very effective and efficient when queries are of little distortions. However, in real environment, background noise distortions are unavoidable. In this case, short patterns in query can hardly find exact matches.

Table 6.4 compares exact search, kNN search and k-RNN search in finding matching patterns ($k = 1$). 2000 patterns are extracted from the query set for each SNR level. The numbers shown in the table are pattern accuracy which stands for the percentage of patterns which can return true matching patterns. Under 20dB distortion, 80.10% of the patterns can find true match via 1-RNN, 71.15% via 1NN, but only 6.85% via exact search. It is because noise distortions make query patterns dissimilar to the clean patterns. When distortions get severe, fewer patterns can return true match via all methods.

Pattern accuracy will affect the recognition accuracy, which is confirmed in Table 6.5. Under 20dB distortion, 1-RNN can achieve 100% recognition accuracy because the corresponding pattern accuracy is high. All the matching patterns contribute to the similarity between the distorted version and the original song. 1NN obtains 99% recognition accuracy since some true matching patterns are missed. However, exact search can only obtain 63% recognition accuracy due to its low pattern accuracy. When distortions get more severe, recognition accuracies decrease for all methods.

Table 6.4: Pattern accuracy (in %) of different pattern search methods

Method	SNR(dB)				
	20	15	10	5	0
Exact	6.85	1.75	0.35	0.05	0.05
1NN	71.15	48.30	23.75	6.15	1.15
1-RNN	80.10	60.40	34.35	12.80	2.90

Table 6.5: Recognition accuracy (in %) of different pattern search methods

Method	SNR(dB)				
	20	15	10	5	0
Exact	63	28	6	1	1
1NN	99	96	88	55	19
1-RNN	100	98	94	66	26

In Table 6.4, although pattern accuracy of 1-RNN at 0dB SNR (2.9%) is higher than that of exact search at 15dB SNR (1.75%), the corresponding recognition accuracy of the former (26%) is smaller than that of the latter (28%), as shown in Table 6.5. It is because 1-RNN generates a large number of false matches to the patterns, making the distorted version more similar to different songs rather than to the original song. Besides, the similarity between the distorted version and the original song is already very low. In this case, a false positive is more likely to occur. Enlarging k in k-RNN can reduce the possibility of false positive, for it increases the similarity between the distorted version and the original song. Table 6.6 shows the effect of different k in k-RNN. When k gets larger, the accuracy increases as well. When SNR decreases, the effect of larger k in increasing accuracy becomes more obvious. One drawback of larger k is that it incurs more computational cost.

Table 6.6: Recognition accuracy (in %) of different k in k-RNN search

k	SNR(dB)				
	20	15	10	5	0
1	100	98	94	66	26
2	100	100	97	84	45
3	100	100	97	91	61

Figure 6.8 compares the recognition accuracy of AudioDNA with our method. The length of short pattern is set to 4 for both methods, because smaller pattern can achieve better performance for AudioDNA. The result shows that our method is better than AudioDNA under different levels of noise distortions. Our method

achieves 100% accuracy when SNR is 20dB, but AudioDNA only achieves 96% accuracy. As the noise distortion becomes more severe, our method can maintain good performance while AudioDNA degenerates. Two factors lead to such differences: 1) MFPSCC is more robust than MFCC, which is confirmed in Section 6.2.3, and 2) PAS with 1-RNN search strategy is better, as shown in previous experiments.

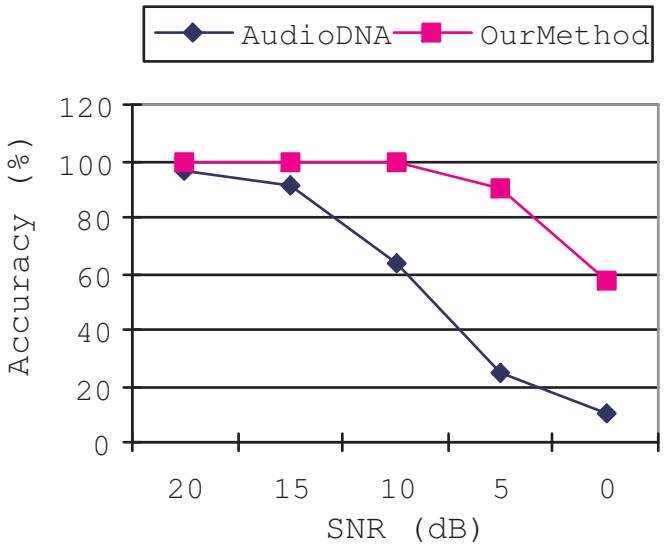


Figure 6.8: Accuracy comparison between our method and AudioDNA

Figure 6.9 shows the accuracy comparison for different query lengths. The queries are 100 audio clips of 5, 10, 15 and 20 seconds, respectively, recorded in a babble noise environment. Generally speaking, the system has good performance with respect to different query lengths when noise distortions are slight. As SNR decreases, longer clips get better performance.

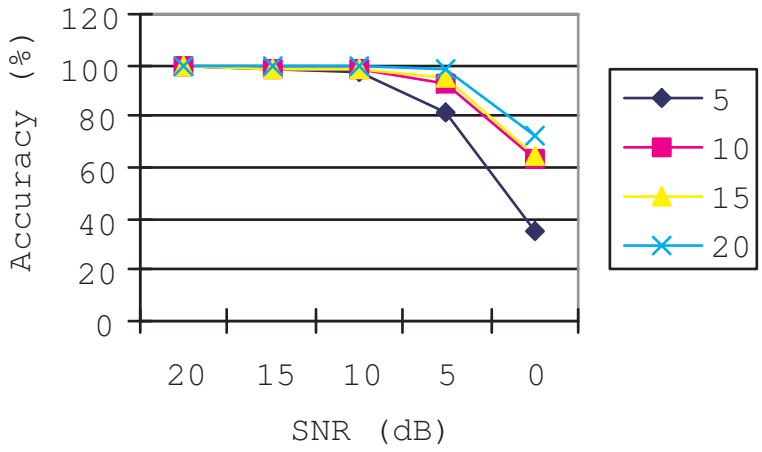


Figure 6.9: Accuracy comparison for queries of different lengths

6.6 Summary

This chapter presents our experimental results for evaluating the proposed methods for audio fingerprinting system.

The first set of experiments compare the robustness of spectral features under different kinds of noise conditions. Four spectral features are compared: Mel-frequency cepstral coefficients, chroma spectrum, constant Q spectrum and product spectrum. The results show that the product spectrum is more robust than the other three features in that it takes advantage of the phase spectrum. Product spectrum based feature has better ROC performance under different noise distortions, and achieves 92.09% overall identification rate with 0.1% false alarm rate. The results also demonstrate that feature normalization and short frame length have great effect in improving recognition accuracies.

The second experiment shows the advantages of PAS when queries are under

distortions due to lossy transmission channel and source editing. The results show the effectiveness and efficiency of PAS compared with Euclidean distance and DTW distance. It can achieve 99% accuracy when a query audio is distorted with 10% data loss, and 100% accuracy when 50% of a query audio is edited, while keeping computationally efficient.

The third experiment shows the advantages of GMM modeling that it gains robustness against noise distortions, reduces the disk space and the memory requirements and is benefit for the subsequent matching process regarding convenience and efficiency. Experimental results show the advantages of GMM modeling that it maintains high accuracy with respect to white noises of 6 different SNR levels from 20dB to -5dB, better than the performance when directly using feature vectors, or modeling with VQ and PCA. Besides, it reduces the disk space and memory requirements, and speeds up the matching process as well.

Finally, our system is compared with an existing work, AudioDNA. Our method is similar to AudioDNA except that the product spectrum based feature and the similarity measure PAS are used. Because AudioDNA is based on exact match of subsequence, its performance decreases as the noise distortions become more severe. As our method considers the effect of noise distortions, it achieves better performance. Experimental results show that our method is more resistent to noise distortions than AudioDNA. Our method can achieve 100% accuracy when queries are 5 seconds clips with 20dB babble noise distortions, but AudioDNA can only achieve 96%. When noise distortions become more severe, our method can maintain good accuracy whereas AudioDNA degenerates. Our method also shows good performance with queries of different lengths.

Chapter 7

Conclusion

As the amount of music data in multimedia databases increases rapidly, there are strong needs to investigate and develop content-based music information retrieval (CBMIR) systems in order to support effective and efficient analysis, retrieval and management for music data. Most of the current used music retrieval systems are based on metadata of music. It requires users to recall and specify metadata of music, which becomes a major restriction on users' queries. Therefore, CBMIR systems are essentially required.

Audio fingerprinting is a technology to identify some piece of unknown audio in a labeled audio database based on a compact set of features, called audio fingerprint, derived from the signal. It provides reliable and fast means for CBMIR because audio fingerprints which have similar function to that of human fingerprints are compact summarizations of the music wave files. A typical audio fingerprinting system contains two major components: fingerprint extraction and matching. The former extracts and models digital audio signals into concise audio fingerprints which are robust enough to identify unlabeled distorted versions of a song as the

same song stored in song database. The latter efficiently looks up the audio fingerprints against the database and judges whether there is a matching song in the database. Although current audio fingerprinting systems are different from each other in various aspect, the fundamental difference of these systems is the used acoustic features. In reality, music signals usually suffer from various distortions and modifications, such as mp3 compression, noise addition and so forth, therefore designing robust and efficient audio fingerprinting system which can resist effects of these distortions becomes crucial.

This thesis focuses on content-based music identification by efficient and robust audio fingerprinting. In particular, we focus on three important modules: feature extraction, fingerprint modeling and matching, which affect the accuracy and efficiency of the whole system. The contributions of this thesis are as follows:

Firstly, several typical spectral features are studied and compared in audio fingerprinting, including MFCC, chroma spectrum, constant Q spectrum, and product spectrum. Although these features have been used in many music/speech applications, their performance in audio fingerprinting are compared the first time. The former three features are derived only from magnitude spectrum. Both chroma spectrum and constant Q spectrum are designed for music signal because they express energy distribution related to music octave, making them superior in music signal analysis, such as key detection and chord recognition. MFCC uses Mel-frequency filter-bank which mimics the human auditory's response, making it robust to noise distortions. It has been widely used in speaker/speech recognition and music modeling. Product spectrum takes advantage of the phase spectrum by using the product of magnitude spectrum and group delay function. It shows effectiveness in robust speech recognition. Its effect in music signal is studied in our work. Experimental results show that product spectrum is more robust than the other

three features in that it utilizes the information of phase spectrum. Product spectrum based feature has better ROC performance under different noise distortions, and achieves 92.09% overall identification rate with 0.1% false alarm rate.

Secondly, a pattern accumulative similarity measure, PAS, is proposed, which better captures the similarity between music data under distortions due to lossy transmission channel, source editing, and background noise. These distortions may result in mismatches both in time and amplitude axes. Euclidean distance and DTW distance, both of which are often used for audio fingerprints sequences, have disadvantages in handling these mismatches. Euclidean distance is very sensitive to distortions in time axis and amplitude axis. DTW is sensitive to amplitude distortions as well, and computationally expensive. Based on short matching patterns, PAS accumulates the similarity of two audios along the matching path, while diminishes the effect of unmatch. As similar audios have more short segments that match each other than that of dissimilar audios, PAS is more suitable to measure similarities between audios with distortions. Experimental results show that PAS has improvement in effectiveness and efficiency compared with Euclidean distance and DTW distance.

Thirdly, GMM modeling is used to boost the robustness of audio fingerprints. GMM modeling generates robust and concise audio fingerprints, which reduces acoustic feature vectors into several types of tokens. First, a GMM is trained for the music database by using the EM algorithm, which better describes the distribution of acoustic feature space. Then, based on the trained GMM, the feature vectors of music database and test dataset are all converted into symbolic tokens (acoustic events). GMM has advantages over other modeling approaches. Experimental results show the advantages of GMM modeling that it maintains high accuracy with respect to white noises of 6 different SNR levels from 20dB to -5dB, better

than the performance when directly using feature vectors, or modeling with VQ and PCA.

Finally, our method is compared with an audio fingerprinting approach, AudioDNA. AudioDNA is designed for robust song detection in broadcast audio. It generates a sequence of acoustic events, called AudioDNA, by statistical modeling. Our method is similar to AudioDNA except that product spectrum based features and similarity measure PAS are used. Experimental results show that our method is more resistent to noise distortions than AudioDNA.

Our future work include integrating audio fingerprinting systems into p2p applications which ensures copyright protection on p2p network, and developing applications for audio streams monitoring.

Bibliography

[1] R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient similarity search in sequence databases. In *Proceedings of the 4th International Conference of Foundations of Data Organization and Algorithms (FODO)*, pages 69–84, 1993.

[2] E. Allamanche, J. Herre, B. Froba, and M. Cremer. Audioid: Towards content-based identification of audio material. In *Proc. 110th AES Convention*, Amsterdam, 2001.

[3] E. Allamanche, J. Herre, O. Hellmuth, F. B. Bernhard, and M. Cremer. Audioid: Towards content-based identification of audio material. In *100th AES Convention*, Amsterdam, Netherlands, May 2000.

[4] S. F. Altschul, W. Gish, W. Miller, E. Myers, and D. J. Lipman. Basic local alignment search tool. *J. Mol. Bil.*, 215:403–410, 1990.

[5] M.A. Bartsch and G.H. Wakefield. To catch a chorus: using chroma-based representations for audiothumbnailing. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 15–19, Mohonk, NY, 2001.

[6] E. Batlle, J. Masip, and E. Guaus. Automatic song identification in noisy broadcast audio. In *IASTED International Conference on Singal and Image Processing*, Hawaii, 2002.

[7] J. C. Brown. Calculation of a constant q spectral transform. *J. Acoust. Soc. Am*, 89:425–434, Jan. 1990.

[8] C. Burges, J. Platt, and S. Jana. Extracting noise-robust features from audio data. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florida, USA, 2002.

[9] C. Burges, J. Platt, and S. Jana. Distortion discriminant analysis for audio fingerprinting. *IEEE Trans. on Speech and Audio Processing*, 11:165–174, May 2003.

[10] P. Cano, E. Batlle, T. Kalker, and J. Haitsma. A review of audio fingerprinting. *Journal of VLSI Signal Processing Systems*, 41:271–284, November 2005.

[11] P. Cano, E. Batlle, H. Mayer, and H. Neuschmied. Robust sound modeling for song detection in broadcast audio. In *Proc. AES 112th Int. Conv*, Munich, Germany, May 2002.

[12] M. Casey and M. Slaney. The importance of sequences in musical similarity. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France, May 2006.

[13] W. Chai. Semantic segmentation and summarization of music. In *IEEE Signal Processing Magazine, Special Issue on Semantic Retrieval of Multimedia*, 2006.

[14] K. Chakrabarti and S. Mehrotra. Local dimensionality reduction: A new approach to indexing high dimensional spaces. In *VLDB*, pages 89–100, 2000.

[15] Y. Chen, M. A. Nascimento, B. C. Ooi, and A. K. H. Tung. Spade: On shape-based pattern detection in streaming time series. In *ICDE*, pages 786–795, 2007.

[16] R. B. Dannenberg and N. Hu. Polyphonic audio matching for score following and intelligent audio editors. In *Proceedings of the 2003 International Computer Music Conference*, pages 27–34, 2003.

[17] G. Das, D. Gunopulos, and H. Mannila. Finding similar time series. In *Principles of Data Mining and Knowledge Discovery*, pages 88–100, 1997.

[18] S. Davis and P. Mermelstein. Experiments in syllable-based recognition of continuous speech. *IEEE Transcactions on Acoustics, Speech and Signal Processing*, 28:357–366, Aug 1980.

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

[20] L. Deng, J. Droppo, and A. Acero. Enhancement of log mel power spectra of speech using a phase-sensitive model of the acoustic environment and sequential estimation of the corrupting noise. *IEEE Trans. on Speech and Audio Processing*, 12(2):133–143, 2004.

[21] S. Dixon and G. Widmer. Match: A music alignment tool chest. In *ISMIR*, pages 492–497, 2005.

[22] Etantrum. http://www.freshmeat.net/projects/songprint/.

[23] J. Foote. Visualizing music and audio using self-similarity. In *ACM Multimedia (1)*, page 77C80, 1999.

[24] J. Goldstein, J. C. Platt, and C. J. C. Burges. Indexing high dimensional rectangles for fast multimedia identification. *Microsoft Research Tech. Report MSR-TR-2003-38*, 2003.

[25] M. Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Transactions on Audio, Speech and Language Processing*, 14:1783–1794, Sept. 2006.

[26] J. M. Grey. Multidimensional perceptual scaling of musical timbres. *Journal of the Acoustical Society of America*, 61:1270–1277, 1977.

[27] J. Haitsma and T. Kalker. A highly robust audio fingerprinting system. In *Proc. ISMIR*, 2002.

[28] D. B. Heras, J. C. Cabaleiro, V. B. Perez, P. Costas, and F. F. Rivera. Principal component analysis on vector computers. In *Proceedings of Vector and Parallel Processing*, pages 416–428, 1996.

[29] J. Herre, E. Allamanche, and O. Hellumth. Robust matching of audio signals using spectral flatness features. In *Proc. IEEE Workshop Applications Signal Processing Audio Acoustics*, pages 127–130, 2001.

[30] P. G. Hoel, S. C. Port, and C. J. Stone. *Introduction to Statistical Theory.* Houghton Mifflin, New York, 1971.

[31] X. D. Huang, A. Acero, and H.-W. Hon. *spoken language processing, a guide to theory, algorithm, and system development.* Prentice-Hall, 2001.

[32] H. Jiang. Confidence measures for speech recognition: A survey. *Speech Communication*, 45:455–470, 2005.

[33] W. J. Kent. Blat-the blast-like alignment tool. *Genome Research*, 12:656–664, 2002.

[34] K. Lee. Automatic chord recognition using enhanced pitch class profile. In *Proceedings of International Computer Music Conference (ICMC)*, 2006.

[35] H. Lin, Z. Ou, and X. Xiao. Generalized time-series active search with kullback-leibler distance for audio fingerprinting. *Signal Processing Letters*, 13:465–468, Aug. 2006.

[36] D. J. Lipman and W. R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227:1435–1441, March 1985.

[37] B. Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the First International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, Massachusetts, 2000.

[38] Shazam Ltd. http://www.shazam.com.

[39] L. Lu, M.Wang, and H. Zhang. Repeating pattern discovery and structure analysis from acoustic music data. In *6th ACM SIGMM International Workshop on Multimedia Information Retrieval*, Edinburgh, 2004.

[40] Musicbrainz. ftp://ftp.musicbrainz.org/pub/musicbrainz/.

[41] A. V. Oppenheim and R. W. Schafer. *Digital Signal Processing*. NJ:Prentice-Hall, Englewood Cliffs, 1975.

[42] K. K. Paliwal and L. Alsteris. Usefulness of phase spectrum in human speech perception. *Proc. Eurospeech*, pages 2117–2120, 2003.

[43] C. Patterson. Homology in classical and molecular biology. *Molecular Biology and Evolution*, 5:603–625, 1988.

[44] S. Pauws. Musical key extraction from audio. In *Proc. 5th Int. Conf. Music Information Retrieval (ISMIR)*, pages 96–99, 2004.

[45] J. W. Picone. Signal modeling techniques in speech recognition. *Proc. IEEE*, 81(9), 1993.

[46] L. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.

[47] A. Ramalingam and S. Krishnan. Gaussian mixture modeling using short time fourier transform features for audio fingerprinting. In *Proc. International Conference on Multimedia and Expo (ICME)*, pages 1146–1149, Amsterdam, Netherlands, July 2005.

[48] J.S. Seo, M. Jin, S. Lee, D. Jang, S. Lee, and C.D. Yoo. Audio fingerprinting based on normalized spectral subband moments. *IEEE Signal Processing Letters*, 13:209– 212, April 2006.

[49] R. Shepard. Circularity in judgement of relative pitch. *Journal of the acoustical society of america*, 36:2346–2353, 1964.

[50] S. S. Stevens and J. Volkman. The relation of pitch to frequency. *American Journal of Psychology*, 53:329–353, 1940.

[51] S. Sukittanon, L. Atlas, and J. Pitton. Modulation scale analysis for content identification. *IEEE Trans. Signal Process*, 52:3023–3035, Oct. 2004.

[52] Y. Tanaka, K. Iwamoto, and K. Uehara. Discovery of time-series motif from multi-dimensional data based on mdl principle. *Machine Learning*, 58:269–300, February 2005.

[53] G. Tzanetaki and P. Cook. Music analysis and retrieval systems for audio signals. *Journal of the American Society for Information Science and Technology*, 55:1077–1083, 2004.

[54] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, July 2002.

[55] V. Venkatachalam, L. Cazzanti, N. Dhillon, and M. Wells. Automatic identification of sound recordings. *Signal Processing Magazine*, 21:92–99, Mar. 2004.

[56] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. J. Keogh. Indexing multi-dimensional time-series with support for multiple distance measures. In *KDD*, pages 216–225, 2003.

[57] G. H. Wakefield. Mathematical representation of joint time-chroma distributions. In *SPIE*, Denver, Colorado, 1999.

[58] A. Wang. An industrial strength audio search algorithm. In *Proc. 4th Int. Conf. Music Information Retrieval (ISMIR)*, 2003.

[59] E. Weinstein and P. Moreno. Music identificaiton with weighted finite-state transducers. In *ICASSP*, Hawaii, 2007.

[60] D. William and E. Brown. *Theoretical foundations of music*. Wadsworth, Belmont, California, USA, 1978.

[61] D. Zhu and K. K. Paliwal. Product of power spectrum and group delay function for speech recognition. *Proc. ICASSP*, 2004.

[62] Y. Zhu and M. S. Kankanhalli. Precise pitch profile feature extraction from musical audio for key detection. *IEEE Transactions on Multimedia*, 8:575–584, 2006.