Audio Fingerprinting for Speech Reconstruction and Recognition in Noisy Environments

by

Feng Liu
B.Sc., Beijing University of Posts and Telecommunications, 2009
M.Sc., Beijing University of Posts and Telecommunications, 2012

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Audio Fingerprinting for Speech Reconstruction and Recognition in Noisy Environments

by

Feng Liu
B.Sc., Beijing University of Posts and Telecommunications, 2009
M.Sc., Beijing University of Posts and Telecommunications, 2012

Supervisory Committee

Dr. George Tzanetakis, Supervisor
(Department of Computer Science)

Dr. Kui Wu, Departmental Member
(Department of Computer Science)

Supervisory Committee

Dr. George Tzanetakis, Supervisor
(Department of Computer Science)

Dr. Kui Wu, Departmental Member
(Department of Computer Science)

ABSTRACT

Audio fingerprinting is a highly specific content-based audio retrieval technique. Given a short audio fragment as query, an audio fingerprinting system can identify the particular file that contains the fragment in a large library potentially consisting of millions of audio files. In this thesis, we investigate the possibility and feasibility of applying audio fingerprinting to do speech recognition in noisy environments based on speech reconstruction. To reconstruct noisy speech, the speech is divided into small segments of equal length at first. Then, audio fingerprinting is used to find the most similar segment in a large dataset consisting of clean speech files. If the similarity is above a threshold, the noisy segment is replaced with the clean segment. At last, all the segments, after conditional replacement, are concatenated to form the reconstructed speech, which is sent to a traditional speech recognition system.

In the above procedure, a critical step is using audio fingerprinting to find the clean speech segment in a dataset. To test its performance, we build a landmark-based audio fingerprinting system. Experimental results show that this baseline system performs well in traditional applications, but its accuracy in this new application is not as good as we expected. Next, we propose three strategies to improve the system, resulting in better accuracy than the baseline system. Finally, we integrate the improved audio fingerprinting system into a traditional speech recognition system and evaluate the performance of the whole system.

Contents

List of Tables

List of Figures

List of Acronyms

FFT ..... Fast Fourier Transform

DFT ..... Discrete Fourier Transform

STFT ..... Short Time Fourier Transform

MFCC ..... Mel-Frequency Cepstrum Coefficient

MIR ..... Music Information Retrieval

HTK ..... Hidden Markov Model Toolkit

WER ..... Word Error Rate

SNR ..... Signal Noise Ratio

CQT ..... Constant-Q Transform

HMM ..... Hidden Markov Model

SVM ..... Support Vector Machine

ACKNOWLEDGEMENTS

I would like to thank:

Dr. George Tzanetakis, for supervising me in my research and supporting me during my graduate study at UVic.

Dr. Kui Wu and Dr. Peter Driessen for serving in my thesis examining committee.

Hang Li, for her accompaniment.

My friends, for their support and encouragement.

Cease to struggle and you cease to live
Thomas Carlyle

DEDICATION

To my parents,
and Hang.

Chapter 1

Introduction

Audio fingerprinting is a content-based audio retrieval technique. It is most commonly used in identifying the source of a piece of query audio content from a huge collection of audio files. Through extracting compact acoustic features, which are known as the audio fingerprint, this technique creates a database that stores only the fingerprint data of a large number of audio files. Later, when an unknown piece of audio is presented, its features are calculated using the same way and used to match against those features stored in the database. If the fingerprint of the query audio content matches a record in the database successfully, they are identified as the same audio content and the meta-data of that piece of audio is returned.

According to previous work done in this field, an ideal audio fingerprinting system should meet several requirements [12][29]. First of all, it needs to be robust against distortions such as additive noise, time stretch, lossy audio compression and interferences of other signals, since in real-world scenarios, query audio is frequently affected by these distortions. Secondly, it has to be scalable. The database should contain a large digital audio catalog that keeps growing in size. Thirdly, fingerprints should be compact and efficient to calculate, so as to minimize the size of the database and the transmission delay for remote services. Fourthly, the fingerprints should be highly specific so that a short query fragment will only match the corresponding document in a database consisting of millions of other audio files. And finally, the strategy to carry out database look-ups should be very efficient. All these five requirements need to be taken seriously when developing reliable large-scale audio fingerprinting applications.

Nowadays, there are plenty of practical applications based on audio fingerprinting. They can be classified into three categories [13]:

- Audio Content Monitoring and Tracking. In most countries, radio stations are required to pay royalties before they air a piece of music. Worrying whether royalties have been paid properly, some right holders want to monitor the potential radio channels that may illegally use their music.

- Added-Value Services. A good example is music recognition on mobile devices like smart phones. Imagine you are in a restaurant or a coffee house, and suddenly you hear a nice song but do not know its name. This is when audio fingerprinting can help you find more information about that song. There are already several popular music recognition applications on smart phones, like Shazam [53] and SoundHound [55].

- Integrity Verification Systems. In some scenarios, the integrity of audio files is required to be verified before they are actually used. Integrity means the audio files have not been changed or there is no much distortion. Another possible application is that companies want to check their advertisements are broadcasted with the required length and speed.

1.1 The Problem

Speech recognition is the process to convert speech signal to the corresponding sequence of words [21]. It has been implemented on mobile devices, computers or cloud [34]. Sometimes, it is also known as automatic speech recognition . A general speech recognition system is illustrated in Figure 1.1. The acoustic model describes the probabilistic relationship between audio signal and phonemes which are the basic units of speech. It is calculated from a training dataset consisting of speech files and their corresponding transcripts. The lexicon describes how the phonemes make up individual words and the language model defines the probability of different combinations of words. Given a speech waveform, the recognition algorithm collects probability information from these three sources and outputs the word string with the highest probability.

Recently, with the development of smart phones, wearable devices and virtual reality, the demand for robust speech recognition has increased greatly, requiring speech recognition to work in much more challenging circumstances. For example, a user may want to use Siri in his iPhone when he is driving a car or sitting in a restaurant, where interference sounds around the phone may distort the original

speech. A traditional speech recognition system will have a lot of problems in this scenario. As shown in Figure 1.2, the system is trained by clean speech, while later is fed with corrupted speech. This mismatch between the training and operating conditions will result in dramatic deterioration in the recognition rate of the speech recognition system.

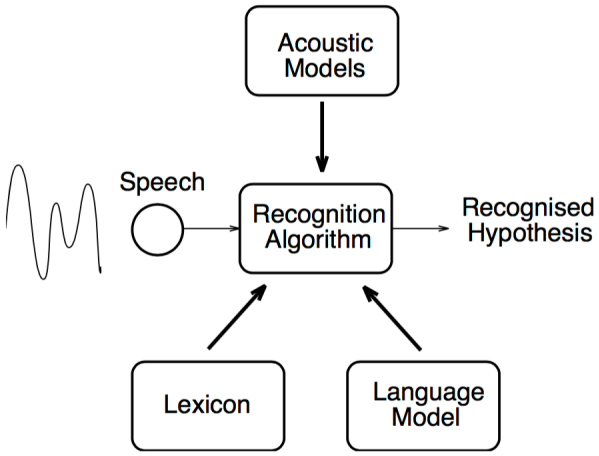

Figure 1.1: General speech recognition system [25]

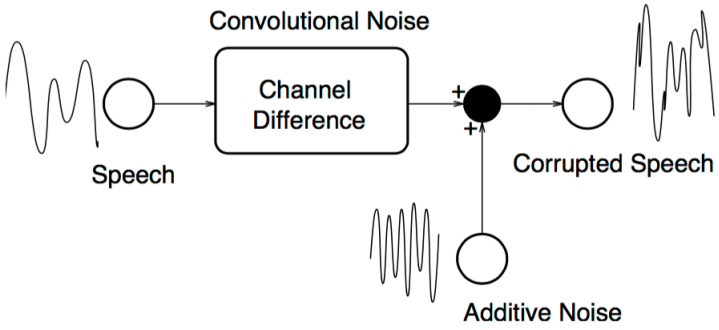

Figure 1.2: Simplified distortion framework [25]

In order to solve this problem, robust speech recognition strategies need to be designed. In the ideal case, the original speech should be recovered from the corrupted speech contaminated by various kinds of degradations such as additive noise, pitching, equalization, audio coder (such as GSM and MP3), to name a few. We know that a reliable audio fingerprinting system is robust against these distortions. So this naturally leads to the following question: can we integrate a traditional speech recognition system with a robust audio fingerprinting scheme to build a robust speech

recognition system applicable in noisy environments? This question leads to another two questions: How robust is the state-of-the-art audio fingerprinting system against these distortions? How to implement an audio system which is suitable for speech? In this thesis, we try to answer these questions.

1.2 Contributions of This Thesis

To the best of my knowledge, audio fingerprinting has never been used in robust speech recognition. It is a big challenge to combine two different techniques. The main contributions of this thesis are listed as follows:

- Detailed implementation of a landmark-based audio fingerprinting system is documented. This system is based on Dan Ellis' work [20], which implements the algorithm described in [62]. The prominent peaks on the spectrogram are extracted and formed into pairs as fingerprints, as the peaks are most likely to survive various types of noises and distortions.

- Thorough evaluation of the audio fingerprinting system for music signals under additive noise and various types of degradations is carried out. Before actually applying the audio fingerprinting system to robust speech recognition, a thorough evaluation is necessary. In this work, the audio fingerprinting system is tested with additive white noise, additive pub noise, live recording, radio broadcast, smartphone playback, smartphone recording, strong MP3 compression and vinyl.

- Experiments about speech reconstruction are carried out, focusing on a critical step, i.e., finding similar speech segments in a dataset of clean speech recordings to a noisy speech segment. The baseline landmark-based audio fingerprinting algorithm does not perform well in this step, so we propose three strategies to improve its performance, including pre-emphasis, robust landmark and morphological peak extraction.

- A novel speech recognition system is proposed and its possibility and feasibility is investigated. The system is based on audio fingerprinting. At first, an audio fingerprinting system is trained with the same dataset as the dataset of the following speech recognition system. Then, a corrupted speech is divided into

segments of fixed length. The segments are then processed by the audio fingerprinting system to locate similar clean segment in the database. If the similarity is above a threshold, the corrupted segment is replaced with a clean segment. After all the conditional replacements, the segments are concatenated together to get a reconstructed speech. Finally, this speech is sent to a traditional speech recognition system.

The proposed speech recognition system does not perform as well as we expected initially. The recognition rate of the proposed system cannot beat the baseline speech recognition system in noisy environments. However, we believe the investigation of this possibility as well as the simulation and analysis results are still valuable for future researchers.

1.3 Outline

The organization of this thesis is following:

Chapter 1 first describes the concept of audio fingerprinting. Speech recognition with its main challenges in noisy environments is then introduced as the problem we are going to solve in this thesis. Main contributions are listed with brief descriptions.

Chapter 2 introduces the background and previous work of audio fingerprinting and speech recognition. Firstly, basic acoustic processing of audio signal is introduced. Secondly, a general audio fingerprinting framework is presented. And finally, different ways to do robust speech recognition are summarized.

Chapter 3 shows the details to implement a baseline audio fingerprinting system and presents its evaluation results and analysis for music signals.

Chapter 4 presents experiments with speech reconstruction. Three strategies are proposed to improve the accuracy of a key step in speech reconstruction, i.e., finding similar clean speech segment in a dataset to a noisy speech segment.

Chapter 5 proposes a novel speech recognition system. Experiments are carried out to test its performance.

Chapter 6 summarizes this thesis and discusses the future work.

Chapter 2

Background and Related Work

In this chapter, we present the basic concepts and architecture of audio fingerprinting systems, and a summary of the related works done in speech recognition and speech enhancement in noisy environments. We begin with a brief introduction of the acoustic processing for audio signal. Then, a general audio fingerprinting framework is introduced. Most audio fingerprinting algorithms follow a similar architecture. In the end, we review previous work done in noise-robust speech recognition, mainly focusing on speech enhancement techniques.

2.1 Acoustic Processing

Acoustic processing is the basis of audio fingerprinting and speech recognition. The main steps of acoustic processing are: represent a sound wave to facilitate digital signal processing, get the distribution of frequencies from waveforms, and visualize an audio file.

2.1.1 Sound Wave

When we listen to a piece of audio, what our ears get is actually a series of changes of air pressure. The air pressure is generated by the speaker who makes air pass through the glottis and out the oral or nasal cavities [36]. To represent sound waves, we need to plot the changes of air pressure over time. For example, Figure 2.1 shows the waveform for the sentence "set white at B4 now" taken from the GRID[1] audiovisual

[1] `http://spandh.dcs.shef.ac.uk/gridcorpus/`

sentence corpus[2]. In this figure, we can easily distinguish waveforms for the vowels from most consonants in this sentence. The reason is that vowels are voiced and loud, leading to high amplitude in the waveform, while consonants are unvoiced and of low amplitude. Figure 2.2 shows the waveform for the vowel [ε] extracted from this sentence. Note that there are repeated pattens in the wave, which are related to the underlying frequency.

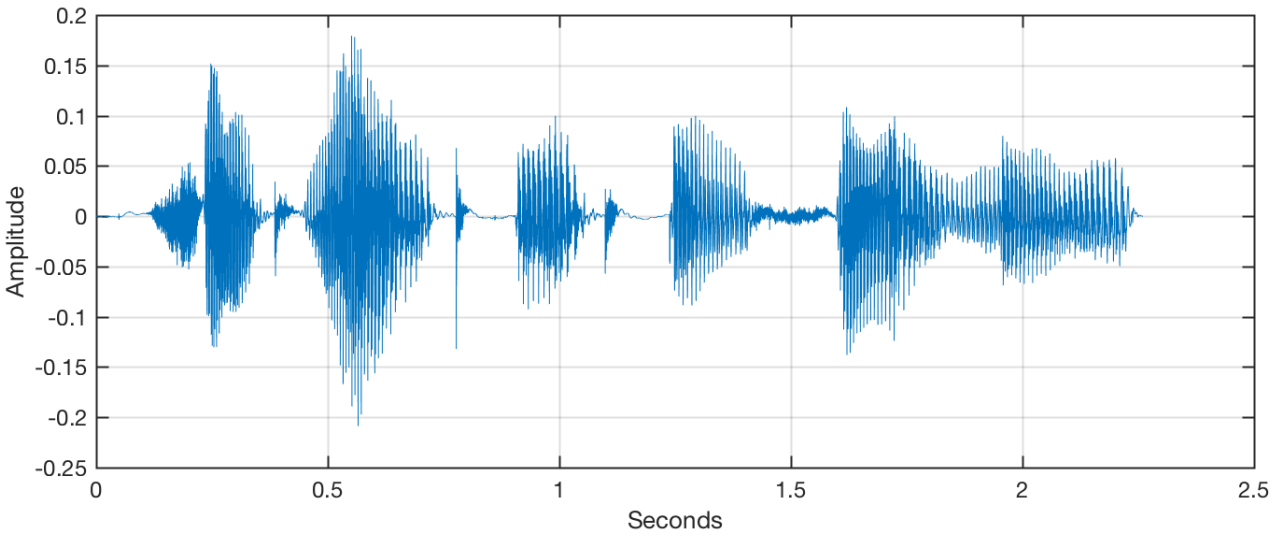

Figure 2.1: The waveform of the sentence "set white at B4 now"

Frequency and amplitude are two important characteristics of a sound wave. Frequency denotes how many times in a second a wave repeats itself. In Figure 2.2, we can find a wave with a special patten that repeats about 16 times in 0.11 seconds. So there is a frequency component of 16/0.11 (145) Hz in this vowel. Here "Hz" is a frequency unit. Amplitude is the strength of air pressure. Zero means the air pressure is normal, positive amplitude means the air pressure is stronger than normal one and negative amplitude means weaker air pressure [36]. From a perceptual perspective, frequency and amplitude are related to pitch and loudness respectively, although the relationship between them is not linear.

To process a sound wave, the first step is to digitize it using an analog-to-digital converter. Actually there are two stages here, sampling and quantization. Sampling is to measure the amplitude of a sound wave with a specified sampling rate, which is the number of samples taken in a second. According to Nyquist–Shannon sampling

[2]Corpus means a large set of speech audio files in linguistics.

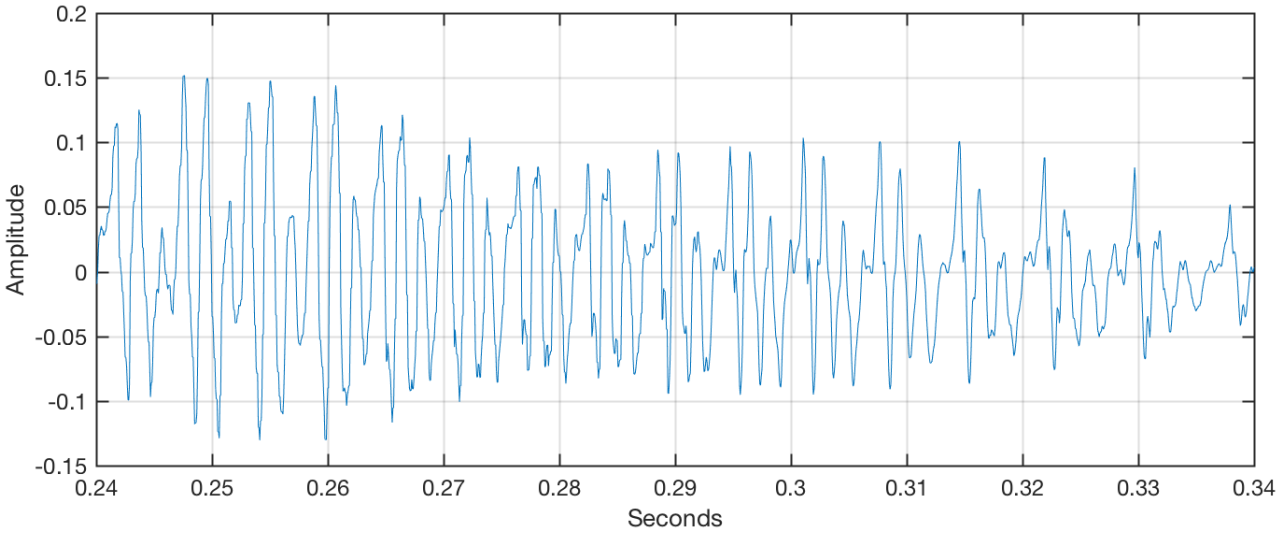

Figure 2.2: The waveform of [ε] extracted from Figure 2.1

theorem [28], the sampling rate should be at least two times the maximum frequency we want to capture. 8,000 Hz and 16,000 Hz are common sampling rate for speech signal, as the major energy of human voice is distributed between 300 Hz and 3,400 Hz [49]. After sampling, a sequence of amplitude measurements, which is real-valued numbers, is outputted. To save the sequence efficiently, we need quantization. In this stage, the real-valued numbers are converted to integers of 8 bits or 16 bits.

2.1.2 Spectrum

Processing sound waves in time domain could be very complicated, however, it turns out to be much simpler when the signal is converted to frequency domain. The mathematical operation that converts an acoustic signal between the time and frequency domains is called a transform. One example is the Fourier transform devised by the French mathematician Fourier in the 1820's, that can transform a time function into the sum of infinite sine waves, each of which represents a different frequency component.

In the context of acoustic signal processing, spectrum is a representation of all the frequency components of a sound wave in frequency domain. Its resolution depends on what transform is used, what the sampling rate is and how many samples we use to compute the spectrum.

The discrete Fourier Transform (DFT) is the most common way to perform Fourier

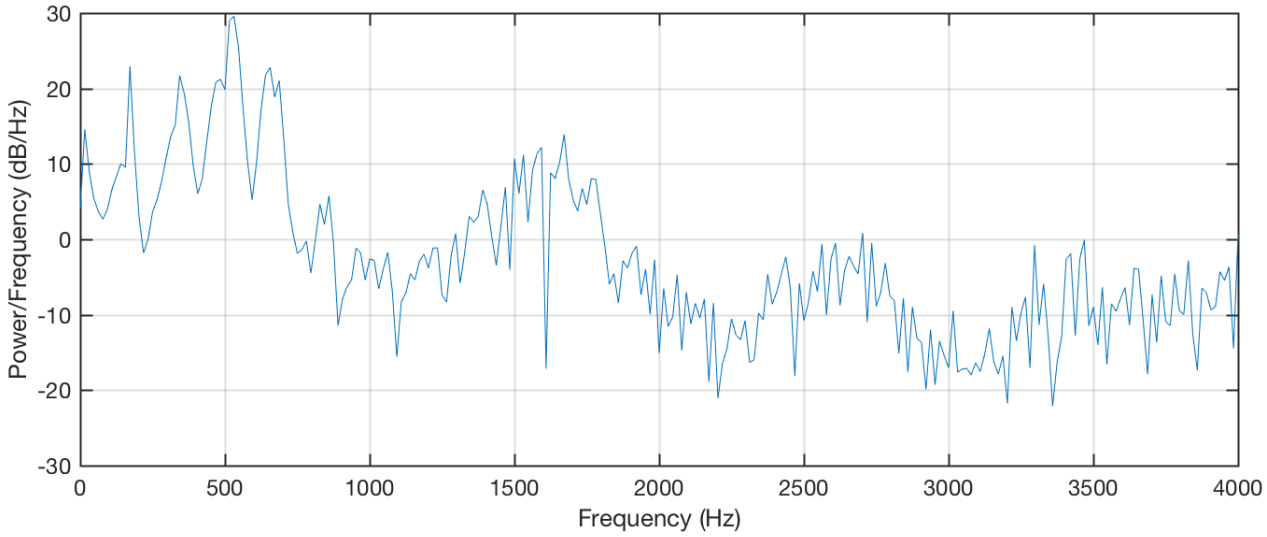

Figure 2.3: The FFT spectrum of the vowel [ɛ]

transform in real applications. DFT is calculated as follows [4]:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-2\pi ikn/N}, \ 0 \leq n < N, \ 0 \leq k < N.$$

Here x is the input sequence of sound wave and X is the frequency output. N is the number of samples we use to calculate.

Figure 2.3 shows the spectrum of [ɛ] in Figure 2.2 calculated with Fast Fourier Transform (FFT), a method which can perform the DFT of a sequence rapidly and generate exactly the same result as evaluating the DFT definition directly. Normally magnitude of each frequency component is measured in decibels (dB). From this figure, we can find that there are two major frequency components at 500 Hz and 1700 Hz in this vowel, and some other weaker frequency components besides them. We can also find a strong frequency component around 150 Hz, which is consistent with our analysis in Section 2.1.1

The above major frequency components are called formants. They are characteristic resonant peaks in the spectrum of a voiced sound. Speech consists of voiced and unvoiced sounds, which are produced by the vowel and consonant portions of words respectively. Each vowel sound has its characteristic formants, as described in Table 2.1.

Table 2.1: Formant frequencies for common vowels in American English [47]

Phonetic Symbol	Example Word	F_1 (Hz)	F_2 (Hz)	F_3 (Hz)
/ow/	bought	570	840	2410
/oo/	boot	300	870	2240
/u/	foot	440	1020	2240
/a/	hot	730	1090	2440
/uh/	but	520	1190	2390
/er/	bird	490	1350	1690
/ae/	bat	660	1720	2410
/e/	bet	530	1840	2480
/i/	bit	390	1990	2550
/iy/	beet	270	2290	3010

2.1.3 Spectrogram

Spectrum provides information about frequency and amplitude of a signal in frequency domain. However, it does not take the time dimension into consideration which is also essential for acoustic signals. In this case, we use spectrogram, a visual representation of the spectrum of an acoustic signal that varies with time.

A spectrum displays frequency on the horizontal axis and amplitude on the vertical axis. In contrast, a spectrogram displays time on the horizontal axis and frequency on the vertical axis, while amplitude is indicated by the intensity of the color of the points in the figure.

Spectrogram represents how the spectrum of a sound wave changes over time. For digital sound signal, it is usually calculated using the Short Time Fourier Transform (STFT) as in Figure 2.4. Firstly, the digital time-domain samples are divided into overlapping frames, which is called the windowing process. Popular window functions includes rectangular window, Hamming window, Hanning window, etc.

$$\textit{Rectangular window} \quad w_n = \begin{cases} 1 & 0 \le n < W \\ 0 & \text{otherwise} \end{cases}$$

$$\textit{Hamming window} \quad w_n = \begin{cases} 0.54 - 0.46 \cdot \cos(\frac{2\pi n}{W}) & 0 \le n < W \\ 0 & \text{otherwise} \end{cases}$$

$$Hanning\ window \quad w_n = \begin{cases} 0.5 - 0.5 \cdot \cos(\frac{2\pi n}{W}) & 0 \leq n < W \\ 0 & \text{otherwise} \end{cases}$$

Rectangular window is rarely used because it will cause discontinuities between frames when we calculate spectrum. Then every frame goes through FFT transformation to get the corresponding spectrum. At last, every spectrum is considered as a column and they are concatenated along time. Figure 2.5 is the spectrogram of the sentence "set white at B4 now" and Figure 2.6 is its 3D view. The horizontal yellow bars in Figure 2.5 represent the formants of vowels in the sentence. For example, we can find three yellow bars between 0.2 and 0.4 seconds in this figure around 500 Hz, 1700 Hz and 2500 Hz, which correspond to the formants of [ɛ] in Table 2.1. Figure 2.6 can give a clearer visualization about these formants at the "mountain peaks".

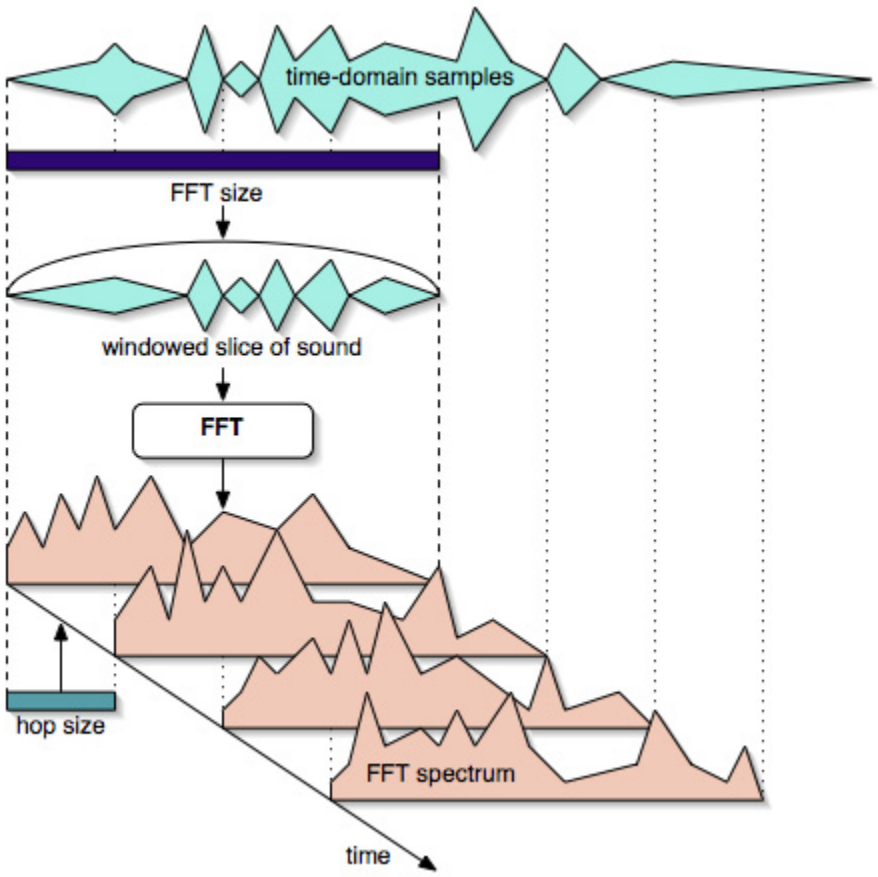

Figure 2.4: Diagram of the Short Time Fourier Transform [50]

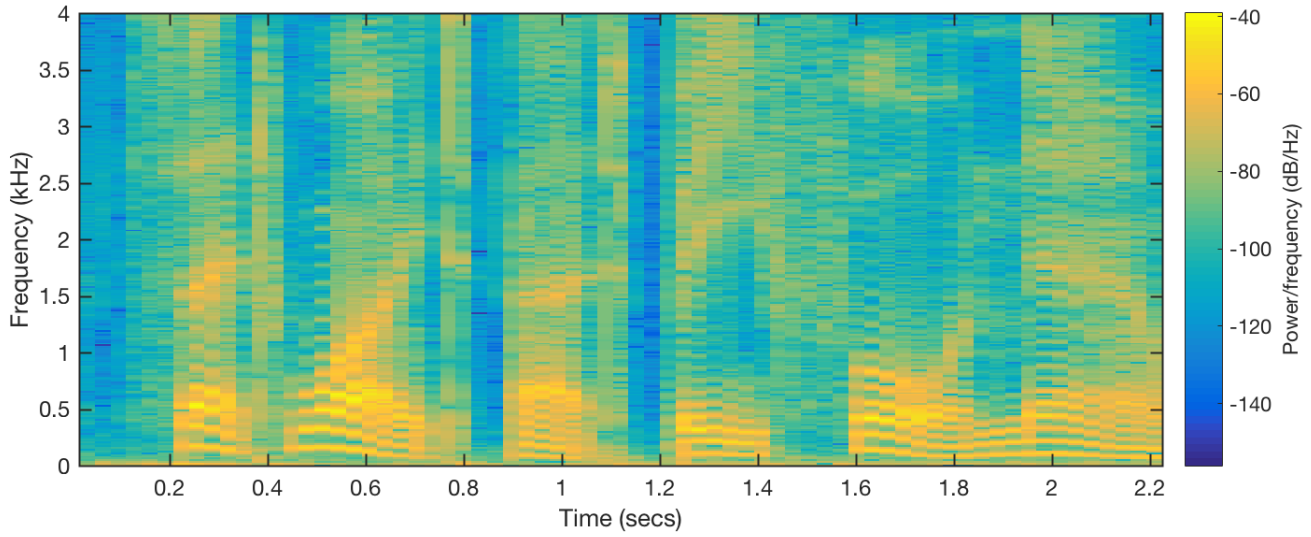

Figure 2.5: The 2D spectrogram of the sentence "set white at B4 now"

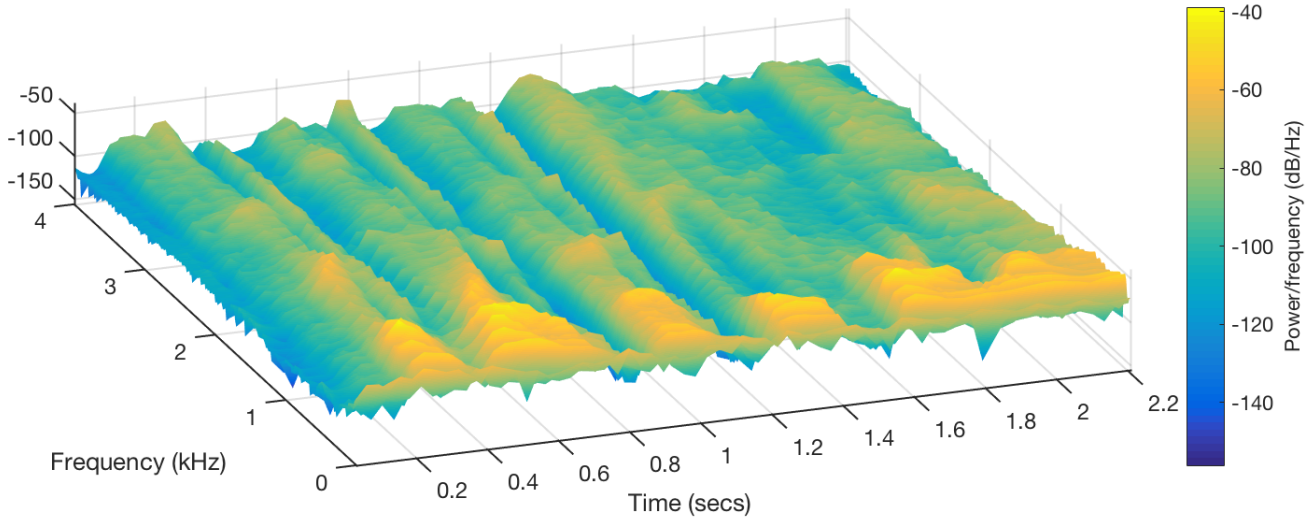

Figure 2.6: The 3D spectrogram of the sentence "set white at B4 now"

2.2 Audio Fingerprinting Framework

Nowadays there are a variety of audio fingerprinting schemes available, but most of them share the same general architecture [12]. As shown in Figure 2.7, there are two major parts: fingerprint extraction and fingerprint matching. The fingerprint extraction part computes a set of characteristics features from the input audio signal. These features are also called fingerprints. They might be extracted at uniform rate [30] or only around special zone on the spectrogram [62]. After fingerprint extraction, these fingerprints of the query sample are used by a matching algorithm to find the best match through searching a large database of fingerprints. In the fingerprint matching part, we compute the distance between the query fingerprint and other fingerprints in the database. The number of comparison is usually very high and the computation of distances could be expensive, so a good matching algorithm is critical. In the end, the hypothesis testing block computes a qualitative or quantitative measurement about the reliability of the searching results.

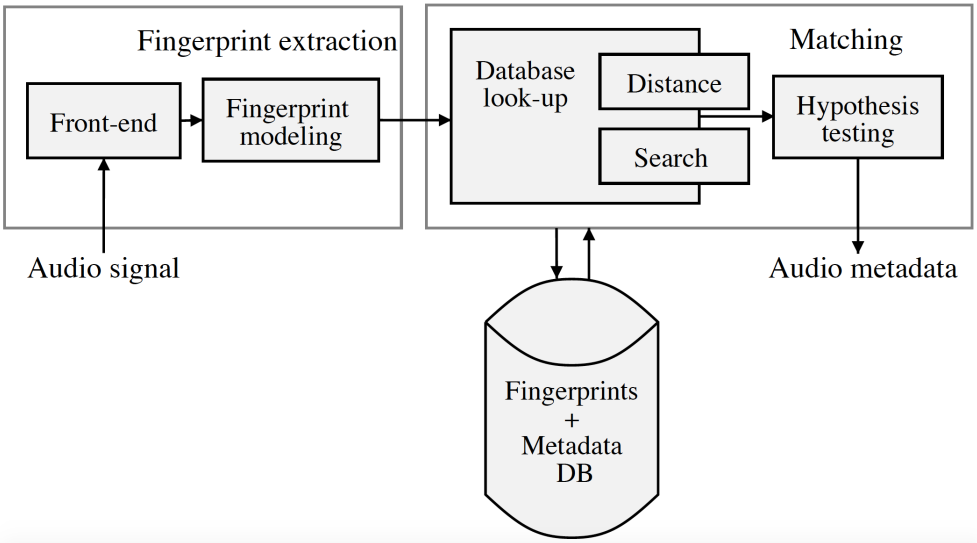

Figure 2.7: General framework for audio fingerprinting [12]

Let's look at this framework from another perspective. It has two working modes, training mode and operating mode. During training mode, reference tracks are fed into the fingerprint extraction part and fingerprints are extracted and stored in a database. When a query track is given, the system switches to operating mode. Fingerprints are extracted by the same means as the training mode and sent to the fingerprint matching part. In this step, fingerprints are compared to other fingerprints

in the database to find the particular document that has most fingerprints in common with the query sample.

2.2.1 Front-end

The front-end block of an audio fingerprinting system computes a set of characteristic features from the audio signal and sends them to the fingerprinting modeling block. These features should be robust to channel distortions and additive noise. Generally the front-end consists of five steps [12]:

1. Preprocessing. In this step, the audio signal is digitalized and quantized at first. Then, it is converted to mono signal by averaging two channels if necessary. Finally, it is resampled if the sampling rate is different with the target rate.

2. Framing. Framing means dividing the audio signal into frames of equal length by a window function (e.g. Hanning window). During this process, a large portion of the audio signal may be suppressed by the window function [33] because the value is very small near the boundaries of the window function. To compensate the loss of energy, the frames overlap.

3. Transformation. This step is designed to transform the set of frames to a new set of features, in order to reduce the redundancy. Most solutions choose standard transformation from time domain to frequency domain, like FFT. There are also some other transformations including the Discrete Cosine Transform [2], the Walsh-Hadamard Transform [58], the Modulated Complex Transform [43], the Singular Value Decomposition [59], etc.

4. Feature Extraction. After transformation, final acoustic features are extracted from the time-frequency representation. The main purpose is to reduce the dimensionality and increase the robustness to distortions. There are plenty of schemes proposed by researchers, such as Mel-Frequency Cepstrum Coefficients (MFCC) [14], Spectral Flatness Measure [3], "band representative vectors" [46], etc.

5. Post-processing. To capture the temporal variations of the audio signal, higher order time derivatives are required sometimes. For example, in [14], besides the MFCC features extracted in Step 4, the final feature vector also includes the derivatives and accelerations of the feature, as well as the derivatives and

accelerations of the energy. Although the derivative of the features will amplify noise [48], the distortions introduced can be reduced by use of a linear time invariant filter.

2.2.2 Fingerprint Modeling

The fingerprint modeling block computes the final fingerprint based on the sequence of feature vectors extracted by the front-end. Every frame generates a feature vector, so the initial sequence of feature vectors is too large to be used as fingerprint directly. In order to reduce the its size, a variety of methods have been proposed. In [52], Schwartzbard calculates a concise form of fingerprint from the means and variances of the 16 bank-filtered energies. In this way, a fingerprint of 512 bits represents 30 seconds of audio. In [16], Chen et al. use MPEG-7 Audio Signature descriptors to reduce the data. For m frames, if the scaling factor is df, the row number of the Weighted Audio Spectrum Flatness feature matrix will be $b = \lceil m/df \rceil$. In [30], Haitsma et al. generate sub-fingerprints over the energy differences along the time and the frequency axes and combine 256 subsequent sub-fingerprints as one fingerprint to represent one song.

2.2.3 Distance and Search

After fingerprints are extracted from the query audio, we need to search for similar fingerprints in the database. Here the similarity is the measure of how much alike two fingerprints are, and is described as a distance. Small distance indicates high degree of similarity, and vice versa. Popular similarity distance measures include the Euclidean distance [8], Manhattan distance [31], an error metric called "Exponential Pseudo Norm" [51], accumulated approximation error [3], etc. How to compute the distance largely depends on the design of the fingerprint.

Searching for the similar items in a large database is a non-trivial task, although it may be easy to find the exact same item. There are millions of fingerprints in the database, so it is unlikely to be efficient to compare them one by one. The general strategy is to design an index data structure to decrease the number of distance calculations. To further accelerate the searching procedure, some searching algorithms adopt multi-step searching strategy. In [31], Haitsma et al. design a two-phase search algorithm. Full fingerprint comparisons are only performed when they have been selected by a sub-fingerprint search. In [40], Lin et al. propose a matching system

consisting of three parts: “atomic” subsequence matching, long subsequence matching and sequence matching.

2.2.4 Hypothesis Testing

The final step is to decide whether there is a matching item in the database. If the similarity, which is based on the above distance, between the query fingerprint and other reference fingerprints in the database is above a threshold, the reference item will be returned as the matching result, otherwise the system thinks there is no matching item in the database. Based on the matching results, the performance of an audio fingerprinting system is measured as a fraction of the number of correct match out of all the queries that are used to test. Most systems report this recognition rate as their evaluation results [38],[62],[6],[35].

2.3 Speech Recognition

So far, a variety of algorithms have been proposed for speech recognition. The word error rate (WER) is close to zero in some laboratory environments where there is almost no noise and distortions. In September 2016, research scientists in Microsoft achieved a WER of 5.9% on an industrial benchmark [64], which has reached human parity. However, the presence of noise and other distortions will seriously degrade the performance of most existing speech recognition systems, so improvements are required before this technique can be widely used in our daily lives.

From a high level of perspective, the performance degradation of speech recognition in noisy environments results from the mismatch between the training and operating conditions. Figure 2.8 shows the performance of the baseline system in the 2nd CHiME Speech Separation and Recognition Challenge[3]. There is no noise suppression preprocessing in this system. The test data is noisy reverberated speech, and noisy training data is reverberated in the same environment and interfered by the same noise as the test data. We get the lowest WER with noisy training data, so we can say that the less the mismatch between the training data and the test data is, the better the performance is.

To describe how to overcome the mismatch, we use the transformation f defined

[3] `http://spandh.dcs.shef.ac.uk/chime_challenge/chime2013/index.html`

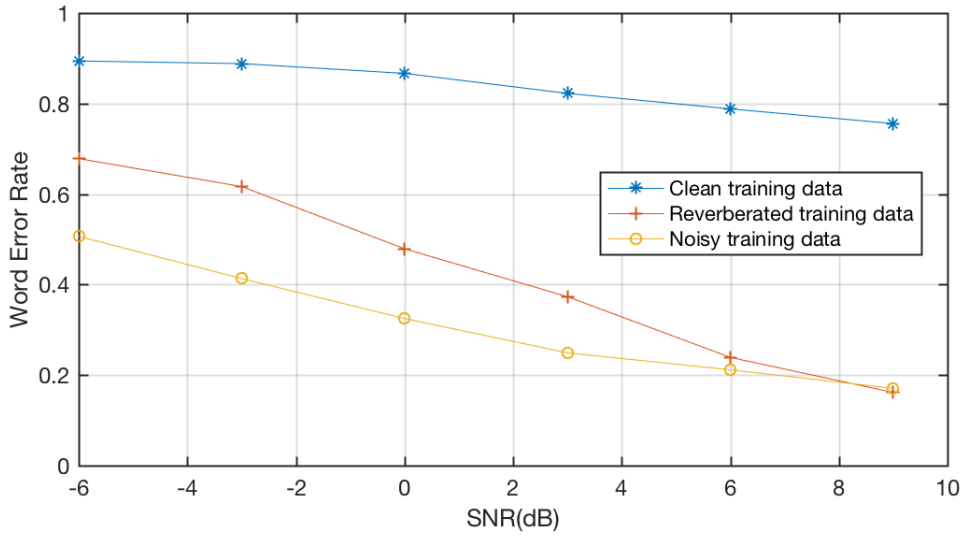

Figure 2.8: Word error rates for noisy, reverberated and clean training dataset [17]

in [27]:

$$q_\beta(s) = f(q_\alpha(s))$$

Here s is the model of a recognition unit (e.g. a phoneme or word) and $q_e(s)$ is some quantity defined on s in the environment e. The transformation f represents a mapping of quantities between two different environments α and β. A robust speech recognition system should have a optimized transformation minimizing the environment mismatch. Depending on the choice of α and β, there are two categories of transformations [27]:

- α is training environment and β is operating environment. This represents observation speech data transformation. The test speech data is transformed from a environment with distortions to the training environment before recognition.
- α is operating environment and β is training environment. This is speech model parameters transformation. Model parameters are adapted to match the operating environment with distortions.

Based on the above categorization, there are three basic ways to implement robust speech recognition [27]:

- Ignore the mismatch and do the same speech recognition for noisy and clean speech. To be robust, the system should be built with noise and distortions resistant features.

- Preprocess the input speech to reduce the noise and distortions. This way is also called speech enhancement.
- Adapt the parameters of speech models in order to match the noisy environment. One way is to use noisy speech to train the system.

This thesis focus on the second way, speech enhancement, which aims to reduce noise using various algorithms. A novel speech enhancement algorithm is proposed in this thesis. Specifically, we will use audio fingerprinting technique to preprocess the noisy speech, in order to recover the waveform of the clean speech embedded in noise.

2.4 Speech Enhancement

In the past decades, there have been plenty of speech enhancement algorithms proposed by researchers in scientific community. One way to classify them is based on how many channels are used, single-channel, dual-channel or multi-channel. Dual-channel and multi-channel enhancement end up with better performance than single-channel enhancement [23], but single-channel enhancement is still widely used to reduce additive noise because of its simple implementation and easy computation.

The spectral subtraction method is a classic single-channel speech enhancement technique. There are several assumptions in this method:

- The background noise is additive;
- The background noise environment is locally stationary;
- Most of the noise can be removed by subtracting magnitude spectra.

Based on these assumptions, Boll proposes a direct acoustic noise suppression method [9].

Same as common digital signal processing technique, the input signal is digitized and windowed to $y(n)$ at first, $0 < n \leq N$, N is the window size. This signal is composed of the actual speech signal $x(n)$ and the additive noise $w(n)$,

$$y(n) = x(n) + w(n), 0 < n \leq N$$

After N-point Fourier transform, we get

$$Y(k) = X(k) + W(k), 0 < k \leq N$$

where

$$y(n) \leftrightarrow Y(k), x(n) \leftrightarrow X(k), w(n) \leftrightarrow W(k)$$

$$Y(k) = \sum_{n=0}^{N-1} y(n) \cdot e^{-2\pi ikn/N}$$

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-2\pi ikn/N}$$

$$W(k) = \sum_{n=0}^{N-1} w(n) \cdot e^{-2\pi ikn/N}$$

So the spectral subtraction estimator is defined as

$$\begin{aligned} \hat{X}(k) &= [|Y(k)| - \mu(k)]e^{j\theta_y(k)} \\ &= H(k)Y(k) \end{aligned}$$

where

$$\mu(k) = E\{|W(k)|\}$$

$$H(k) = 1 - \frac{\mu(k)}{|X(k)|}$$

$|\mu(k)|$ is the average value of the spectrum during speech absence frames, $H(k)$ is called the spectral subtraction filter, $\theta_y(k)$ is the phase of the noisy signal. In this way, the spectrum of noise is removed from the input signal and we get relatively clean signal $\hat{X}(k)$. After Inverse Fast Fourier Transform (IFFT), the time-domain signal is derived.

The spectral error of this estimator is

$$\begin{aligned} \xi(k) &= \hat{X}(k) - X(k) \\ &= [|Y(k)| - \mu(k)]e^{j\theta_y(k)} - [Y(k) - W(k)] \\ &= Y(k) - \mu(k)e^{j\theta_y(k)} - Y(k) + W(k) \\ &= W(k) - \mu(k)e^{j\theta_y(k)} \end{aligned}$$

To reduce the above spectral error, several modifications are proposed in [9]. One of them is half-wave rectification. The main idea is to bias down the magnitude spectrum at each frequency bin by the corresponding noise bias. It is expressed as

$$|\hat{X}(k)|^2 = \begin{cases} |Y(k)|^2 - |\mu(k)|^2 & \text{if } |Y(k)|^2 - |\mu(k)|^2 > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

If the noisy signal power spectrum is less than the average noise power spectrum, the output is set to zero.

A slightly different approach in [7] is proposed to compensate for the spectral spikes in Eq.(2.1), which are also call "musical noise". The existence of "musical noise" is due to the differences between the actual noise frame and the noise estimator. In Eq.(2.1), the enhanced signal is set to zero when the actual value is negative. This new approach eliminates the "musical noise" and further reduces the background noise. It subtracts an overestimate of the noise power spectrum and prevents the resultant spectral components from going below a preset minimum level. The new spectral subtraction process is expressed as,

$$|\hat{X}(k)|^2 = \begin{cases} |Y(k)|^2 - \alpha \cdot |\mu(k)|^2 & \text{if } |Y(k)|^2 - |\mu(k)|^2 > \beta \cdot |\mu(k)|^2 \\ \beta \cdot |\mu(k)|^2 & \text{otherwise} \end{cases} \tag{2.2}$$

Here $\alpha \geq 0$ and $0 < \beta \ll 1$. α is the subtraction factor, which is a function of SNR. β is the spectral floor parameter.

In [37], a multi-band spectral subtraction method is proposed. This method is based on the idea that most real world noise is colored and does not affect the speech signal uniformly over the entire frequency range. The entire frequency range is divided into M bands that do not overlap. The spectral subtraction is performed in each band individually. The estimate of the clean speech is obtained by

$$|\hat{X}_i(k)|^2 = \begin{cases} |Y_i(k)|^2 - \alpha_i \cdot \delta_i \cdot |\mu_i(k)|^2 & \text{if } |Y_i(k)|^2 > \alpha_i \cdot \delta_i \cdot |\mu_i(k)|^2 \\ \beta_i \cdot |Y_i(k)|^2 & \text{otherwise} \end{cases} \tag{2.3}$$

where $b_i \leq k \leq e_i$, $0 < \beta \ll 1$. b_i and e_i is the beginning and ending frequency of the ith band. α_i is the over-subtraction factor of the ith band which is determined by the SNR of ith bank. δ_i is a tweaking factor for ith band, in order to customize the

noise spectral subtraction for each band.

Other than these above methods, there have been many other speech enhancement approaches [41] [54] [32] [61] [1] [45] based on Boll's original work [9]. Most, if not all, of them require that the noise is locally stationary and can be estimated from nearby speech absence frames. They are trying to subtract the spectrum of noise from the corrupted signal. However, in this thesis, we try to reconstruct the noisy signal by replacing it with clean signal, which will work even if these requirements are not met.

2.5 Summary

This chapter introduces background and related work of audio fingerprinting and speech recognition. We start with acoustic processing, which is a critical step in audio signal processing. It transforms waveforms of audio signal to time frequency representations, from which characteristic features are extracted for audio fingerprinting and speech recognition. Then, a general audio fingerprinting framework is introduced. Different audio fingerprint techniques are reviewed and their functional parts are mapped to corresponding blocks in the framework. Finally, we talk about speech recognition in noisy environments. As one of the robust speech recognition techniques, speech enhancement aims to improve speech quality by reducing noise and various degradations.

To investigate the possibility and feasibility of applying audio fingerprinting to speech recognition in noisy environments, a robust audio fingerprinting system is necessary. In next chapter, we present the details to implement a state-of-the-art audio fingerprinting system and evaluate it throughly.

Chapter 3

A Baseline Audio Fingerprinting System

These years audio fingerprinting has attract much research interest and a large amount of systems have been proposed. The main difference among them is that they have different ways to compute and model fingerprints [29], which decides the database structure and the matching algorithm. One category of fingerprints is composed of short sequences of frame-based feature vectors, like Bark-scale spectrograms, MFCC, etc. Another category of fingerprints consist of sparse sets of characteristic points, like characteristic wavelet coefficients and spectral peaks, etc.

Wang proposes a well known landmark-based audio fingerprinting system in [62], which is the basic algorithm of Shazam. It pairs spectrogram salient peaks to make up landmarks. These spectrogram peaks, which have highest amplitudes, are selected as the characteristic features since it is believed that they are most likely to survive noise and distortions. The system is also claimed to be computationally efficient, massively scalable and capable of quickly identifying a short segment of music out of a large database of over millions of tracks.

In this thesis, a landmark-based audio fingerprinting system is implemented based on the general framework in Chapter 2 and Ellis' work [22], in order to evaluate its performance and prepare for applying it to speech reconstruction. The block diagram of the system is shown in Figure 3.1. It consists of two stages. During the offline stage, fingerprints of a large number of reference tracks are extracted and stored in a hash table, which serves as a database. During the online stage, the system is presented with a query track. Fingerprints are extracted with the same way as

the offline stage at first. Then the fingerprints are matched against a large set of fingerprints in the database. At last, a ranked list of tracks, in the order of similarity, are returned. In addition, Shift and Unique block is used to overcome the potential time skew between the query track and the reference track. The offline stage and online stage are corresponding to training mode and operating mode of the system respectively.

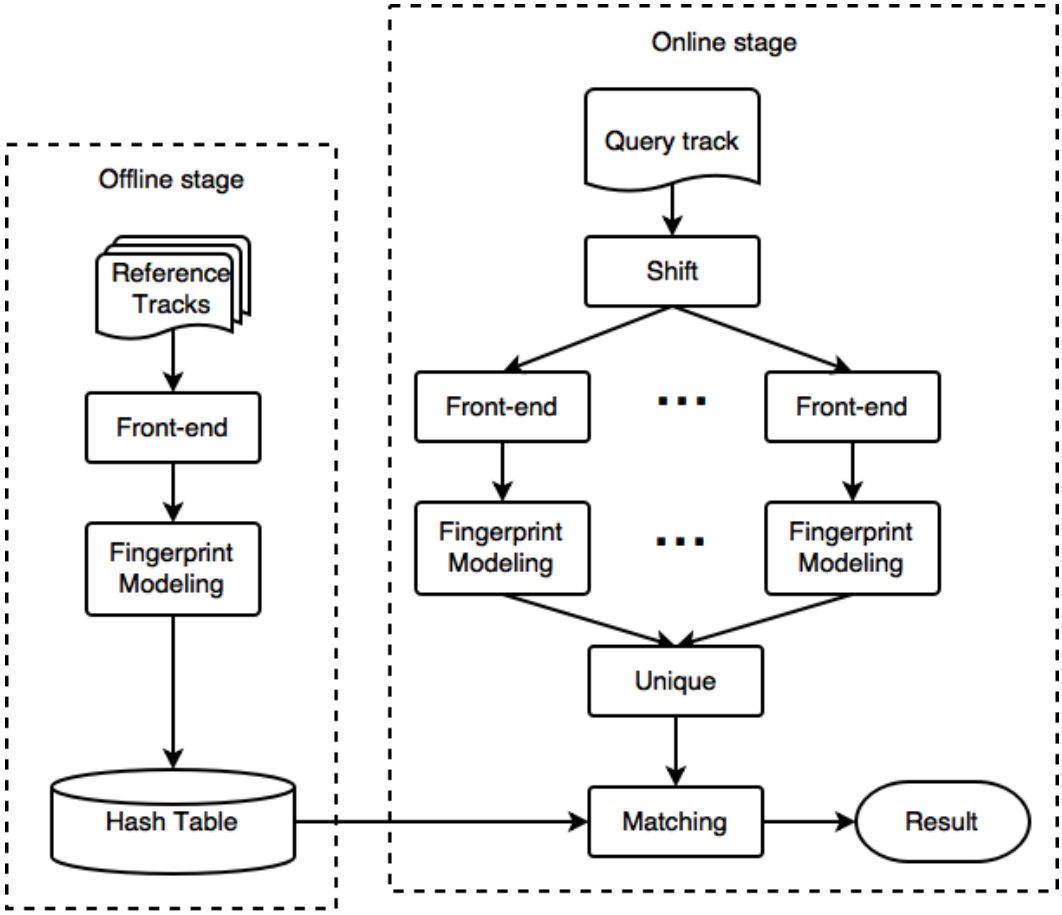

Figure 3.1: Structure of the landmark-based audio fingerprinting system

We will talk about each block of the system in detail in the following sections.

3.1 Front-end

The front-end block is responsible for extracting spectral peaks from audio files. There are three major steps: preprocessing, spectrogram computation and peak extraction.

3.1.1 Preprocessing

The main task for the preprocessing block is to convert the input audio signal to signal of single channel and target sampling rate. Assume the input audio signal is $s_{stereo}(c,n), c \in \{0,1\}, 0 \leq n < L$, c is the channel index and L is the number of samples in the input signal, the following procedures are taken in sequence:

- Convert signal $s_{stereo}(c,n)$ to be monaural.

$$s_{mono}(n) = \frac{s_{stereo}(0,n) + s_{stereo}(1,n)}{2}, 0 \leq n < L$$

- Resample the signal to target sampling rate. According to the Nyquist theorem [4], the target sampling rate should be two times the highest frequency we want to capture at least. For speech signal, as the meaningful frequency range is 0 to 4,000 Hz for human ears, the target sampling rate should be 8,000 Hz at least. Assume the original sampling rate is $f_{original}$ and the target sampling rate is f_{target},

$$s_{mono}(n), 0 \leq n < L \Rightarrow s(n), 0 \leq n < M$$

  Here M is the number of samples in the signal with target sampling rate and $\frac{L}{M} = \frac{f_{original}}{f_{target}}$.

3.1.2 Spectrogram Computation

To get the time frequency representation of the signal, we need to compute its spectrogram by STFT as described in Section 2.1.3. For this step, there are two important parameters, window size N_{win}, which often equals to FFT size N_{FFT}, and hop size N_{hop}. N_{FFT} decides the frequency resolution of the spectrogram, which is the distance between two frequency component in the spectrum. It is calculated as follows,

$$f_{res} = \frac{f_{target}}{N_{FFT}}$$

Hop size is different with window size to generate overlap between frames. The overlap is necessary because the window function in STFT is usually very small or even zero near the window boundaries. If there is no overlap, a large portion of the signal will be suppressed. Hop size N_{hop} depends on the choice of the window function. For

Hanning window, its value is typically half the window size.

$$N_{hop} = \frac{N_{win}}{2}$$

After computation, the spectrogram can be represented with a two-dimensional array,

$$S(f,t), 0 \leq t < N_{frame}, 0 \leq f < N_{bin}$$

where N_{frame} is the frame number and N_{bin} is the total bin number.

$$N_{frame} = \lceil \frac{L}{N_{hop}} \rceil, N_{bin} = \frac{N_{FFT}}{2}$$

Before peak extraction in the following step, the spectrogram requires further processing:

- Calculate the magnitude and ignore the phase information.

$$S(f,t) = |S(f,t)|, 0 \leq t < N_{frame}, 0 \leq f < N_{bin}$$

- Get the log-magnitude.

$$S(f,t) = log(S(f,t)), 0 \leq t < N_{frame}, 0 \leq f < N_{bin}$$

- Make the spectrogram zero-mean, in order to minimize the start-up transients for the following filter.

$$S(f,t) = S(f,t) - E(S), 0 \leq t < N_{frame}, 0 \leq f < N_{bin}$$

- Apply a high-pass filter. The filter equation is

$$y(n) = x(n) - x(n-1) + p \cdot y(n-1)$$

  where p is the pole of the filter. This is designed to remove slowly varying components and emphasize rapidly varying components. The frequency response of this filter with different p is shown in Figure 3.2. A value close to 1 greatly emphasizes rapid variations, ending up with more peaks.

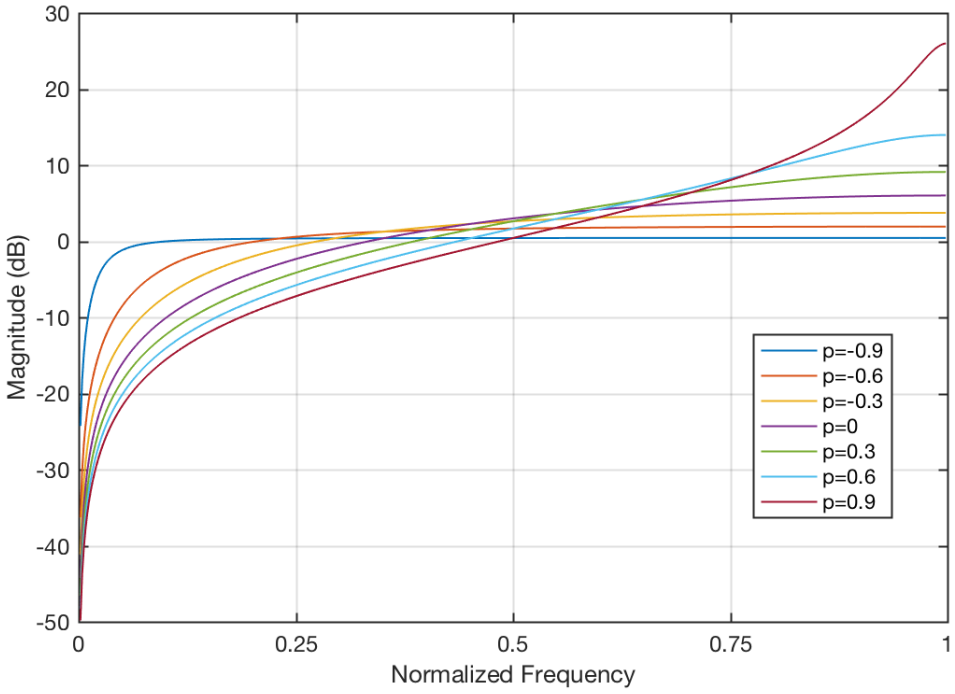

Figure 3.2: Frequency response of the high-pass filter

3.1.3 Gaussian Peak Extraction

Spectrogram peaks are extracted as characteristic features in this step, since they are robust against noise and distortions. A point in the spectrogram is considered as a peak if its amplitude is higher than its the neighbours in a region. Its coordinate is used to represent a peak and its amplitude is ignored. To find peaks that are salient along both frequency and time axes, 1-D Gaussian smoothing and decaying threshold are applied on them respectively.

1-D Gaussian smoothing is used to suppress non-salient maxima in a vector, which is corresponding to a column in the spectrogram. Its calculation is illustrated in Figure 3.3. For the input vector $\{x(n), 0 \leq n < N\}$ (black line), local maxima are extracted at first (black asterisk),

$$\{(a_i, l_i), 0 \leq i < I\}$$

where a_i and l_i is the amplitude and the time location of the ith maximum, I is the number of peaks in the vector. Then superimpose a Gaussian on each local maximum

(all the dotted lines). The Gaussian for maximum (a_i, l_i) is

$$G_i(n) = a_i \cdot e^{-\frac{(n-l_i)^2}{2\cdot\rho^2}}, 0 \leq n < N$$

The pointwise maxima of all the Gaussians is the Gaussian smoothing result, i.e., the envelope of all the Gaussians (red line). In the example of Figure 3.3, after Gaussian smoothing, 11 non-salient maxima are suppressed and the number of peaks is decreased from 17 to 6 (red circles).

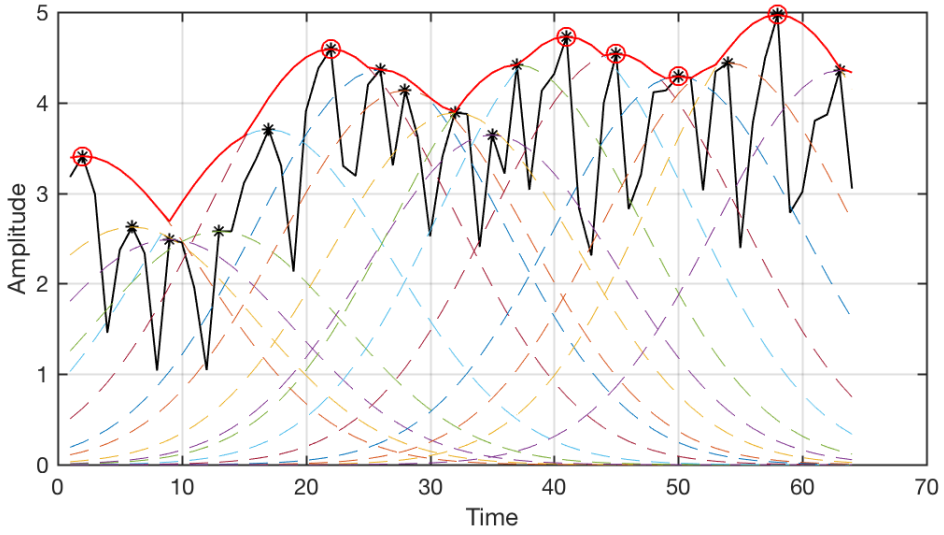

Figure 3.3: Gaussian smoothing

Decaying threshold means the threshold is decaying along time. Here threshold is not a value but a vector with the same length as a column in the spectrogram. Actually, there are two thresholds for one column, initial threshold and updated threshold. For a specific column, Gaussian smoothing is applied to it at first. Then, extract all the local maxima from the column. Only the maxima that are beyond the initial threshold is selected as salient peaks of the column. The updated threshold is calculated by finding the pointwise maximum of the column after Gaussian smoothing and the initial threshold. Then, this threshold is used to calculate the initial threshold for next column by multiplying it with a decaying factor a_{dec}. To get the initial threshold for the first column, extract pointwise maxima over the first F columns and apply Gaussian smoothing on it. A typical value for F is 10.

In summary, Gaussian peak extraction can be described as a forward pruning process. Starting with the first column of the spectrogram, we apply Gaussian smoothing

to a column and extract peaks that are beyond the initial threshold. Then, we calculate the updated threshold of current column and use it to compute the initial threshold for next column. Repeat this routine until we reach the last column of the spectrogram. All the peaks we have extracted in this process are the salient peaks we desire.

In this step, to control the number of salient peaks, there are three choices:

- Adjust the standard deviation in Gaussian smoothing. Larger deviation leads to fewer peaks.
- Modify the decaying factor. A value closer to 1 ends up with fewer peaks. In this baseline system, it is modified by changing the hashes density parameter $D_{training}$ or D_{test} depending on the system mode.

$$a_{dec} = 1 - 0.01 \cdot \frac{D}{35}$$

- Backward pruning. After we finish the forward pruning as we have described, backward pruning will help to further reduce the number of salient peaks.

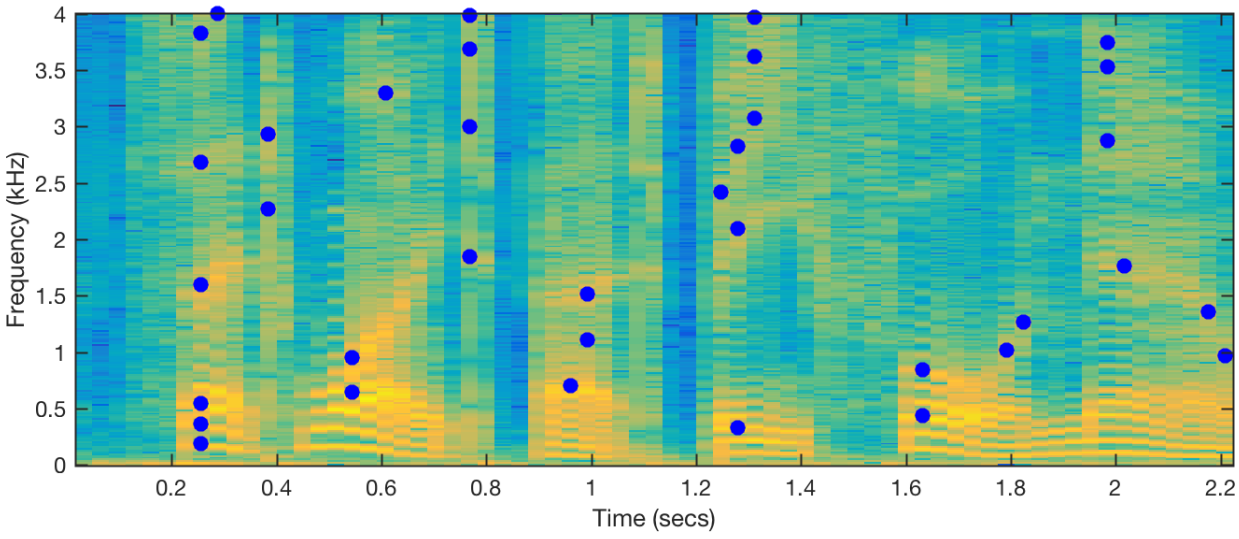

Figure 3.4: The peaks (blue points) extracted from the FFT spectrogram

After peak extraction, a complicated spectrogram is transformed to a compact sequence of coordinates as illustrated in Figure 3.4,

$$S(f,t), t < N_{frame}, 0 \leq f < N_{bin} \Rightarrow \{(f_n, t_n)\}, 0 \leq n < N_{peak}$$

where (f_n, t_n) is the coordinate of peak in the spectrogram and N_{peak} is the number of peaks in an audio track. The coordinate list is called "constellation map" since the peaks in the spectrogram look like many stars in the sky.

3.2 Fingerprint Modeling

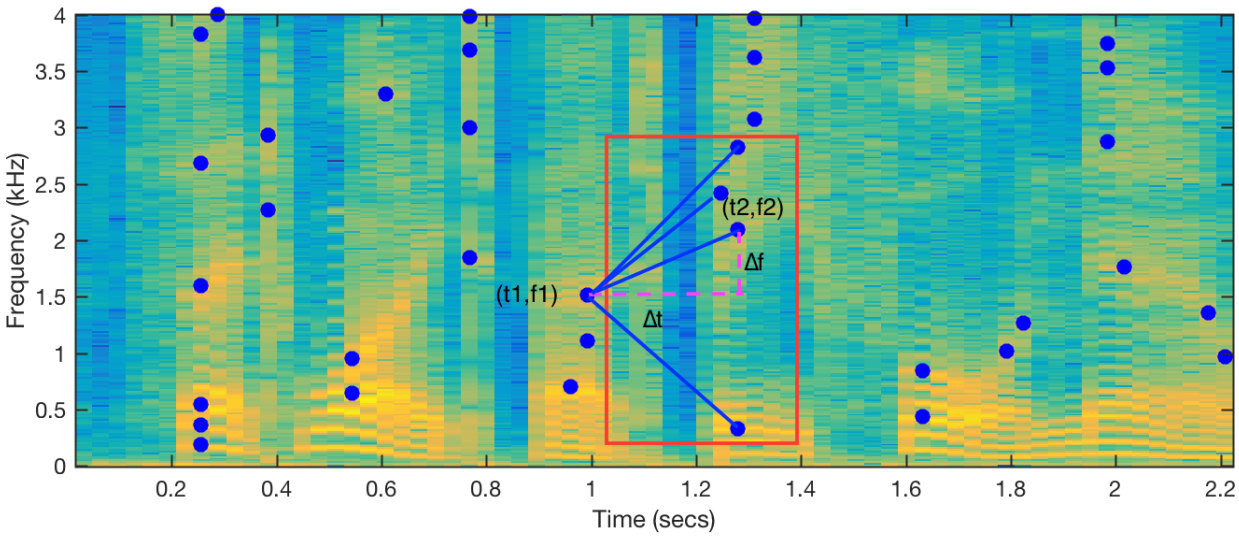

Figure 3.5: Landmark formation

Peaks are paired to form landmarks in order to accelerate the search process when matching, because the entropy of a pair of peaks is much higher than a single peak. As shown in Figure 3.5, every peak in the spectrogram is treated as an anchor point, e.g., (t_1, f_1), and there is a target zone (the area inside the red frame) associated with it. Every anchor point is sequentially paired with N_{fanout} points in the target zone in the descending order of distance. Every pair of peaks is represented with the time and frequency of the anchor point plus the time and frequency difference between the anchor point and the point in the target zone. For example, the pair of (t_1, f_1) and (t_2, f_2) can be represented with

$$t_1 : [f_1, \Delta f, \Delta t], \Delta f = f_2 - f_1, \Delta t = t_2 - t_1$$

This is also called (time offset:hash) pair. Assume f_1, Δf and Δt carry 10 bits of information each, a landmark yields 30 bits of information while a single point yields only 10 bits. High entropy of the landmark accelerates the search procedure greatly.

3.3 Hash Table

After the fingerprints of reference tracks are extracted, we need to save them in a database. In the baseline system, they are stored in a hash table along with their track identifications. For the landmark $t_1 : [f_1, \Delta f, \Delta t]$, the corresponding hash is

$$key = f_1 \cdot 2^{N_{\Delta f}+N_{\Delta t}} + \Delta f \cdot 2^{N_{\Delta t}} + \Delta t$$

$$value = ID \cdot 2^{N_{t_1}} + t_1$$

where N_{t_1}, N_{f_1}, $N_{\Delta t}$ and $N_{\Delta f}$ is the number of bits used to represent t_1,f_1,Δt and Δf, and ID is the track identification. So there are $2^{N_{f_1}+N_{\Delta f}+N_{\Delta t}}$ different keys in all.

Actually, the database is implemented using two arrays, a hash table plus a count table. An example is given in Figure 3.6, N equals to $2^{N_{f_1}+N_{\Delta f}+N_{\Delta t}}$ and M is the maximum bucket size. Hash Table is a two-dimensional array. Every column is a bucket to store all the hash values with same hash key. So there are $2^{N_{\Delta f_1}+N_{\Delta f}+N_{\Delta t}}$ columns in total. The bucket size is a parameter depending on the landmark density and the number of reference tracks. Count Table is a one-dimensional array and its size is also $2^{N_{\Delta f_1}+N_{\Delta f}+N_{\Delta t}}$. The value in this array indicates the number of items stored in the corresponding bucket.

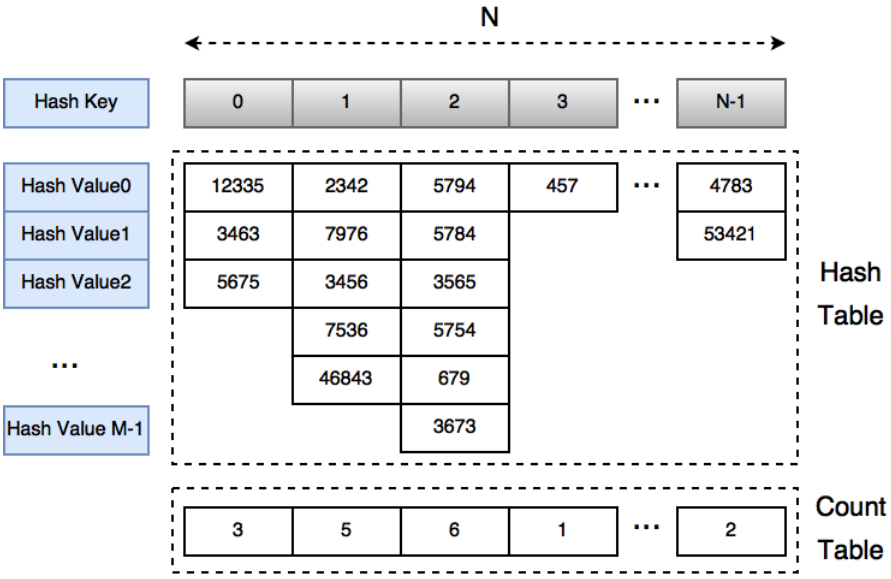

Figure 3.6: An example of the database composed of two tables

When one bucket in the hash table is full, random item in the bucket is replaced. This should be fine because a track will be represented by other hashes. On the other hand, too much hashes in one bucket means these hashes have low significance. Note that only a very small amount of buckets are allowed to be full, otherwise the performance will deteriorate. If this happens, a larger bucket size is required.

3.4 Shift and Unique

It is possible that there is time skew between the query track and the reference track, as shown in Figure 3.7. The time skew happens when the audio signal is windowed to frames and the frame boundaries of query track and reference track are not perfectly aligned. Large time skew may lead to different fingerprints for two same audio files, which is not desirable for a good fingerprint scheme.

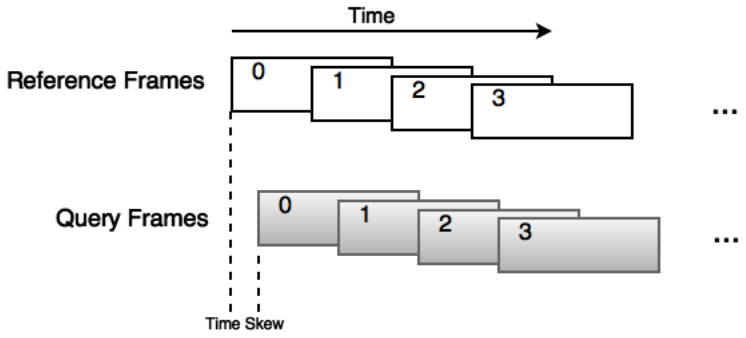

Figure 3.7: Time skew between query track frames and reference track frames

There are two solutions to mitigate this problem. The first one is to decrease the ratio of hop size to frame size for both reference track and query track, as the largest time skew is half the hop size. Usually the frame size is fixed, so what can we do is to decrease the hop size. One drawback of this solution is that the size of the database will increase and it takes more time to compute the fingerprints for a track because the number of frames will increase. The second solution is to extract landmarks several times at various time shifts and combine them next for the query track. This is a better solution since it only affects the landmark extraction of the query track and the size of the database will not change. This solution is adopted in this baseline system. An example of repeated extractions at 4 time shifts is given in Figure 3.8. With 4 different shift size (0, $N_{hop}/4$, $2N_{hop}/4$, $3N_{hop}/4$), we get 4 sets of landmarks.

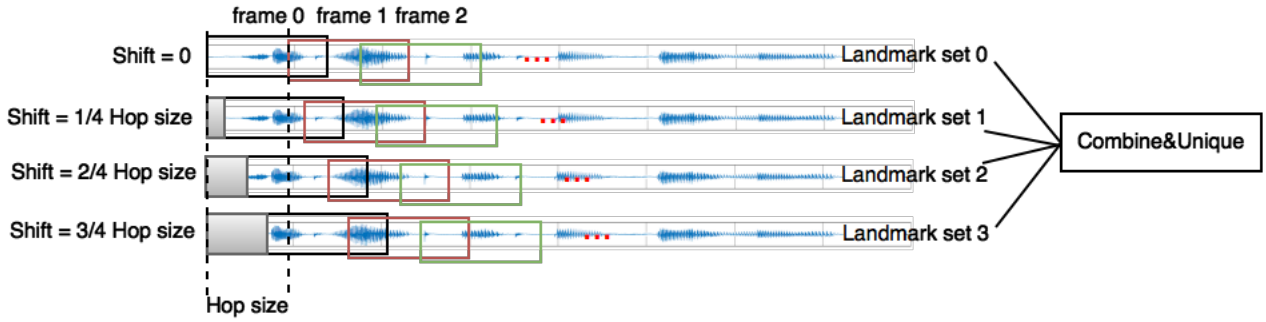

Figure 3.8: Repeated extractions at 4 time shifts

After repeated extractions at various time shifts, "unique" procedure is applied on the landmarks. These repeated extractions may generate same landmarks because the shift size difference between them is quite small. "Unique" procedure will combine all the sets of landmarks and remove the duplicates.

3.5 Matching

Matching is the essential part for the audio fingerprinting system. The basic idea is to find similar, if not exactly same, landmarks patten from the database to the query track. They are not exactly same because the query track may suffer noise and distortions on the transmission channel. In this section, the principle of matching algorithm is introduced at first. Then we will talk about how to implement this algorithm in hash table.

The main procedure of matching algorithm is to scan the database and find similar constellation maps. After fingerprint extraction, a query audio file is transformed to a list of landmarks, which is also a constellation map if landmarks are considered as peaks. The database actually consists of constellation maps of all the reference tracks. Put the constellation map of a reference track on a strip chart and the constellation map of the query track on a transparent piece of plastic. If we slide the piece of plastic over the strip chart of the reference track, at some point when the reference track is a matching track and the time offset is properly located, a significant number of points will coincide. This process is illustrated in Figure 3.9. The constellation map of the query track is sliding over the reference track from left to right. At every shift of the query track, we count the number of points that coincide, which is represented with a bin in the chart below. A significant bin in the chart indicates this is a matching

track and its shift location indicates the time offset between the query track and the matching reference track.

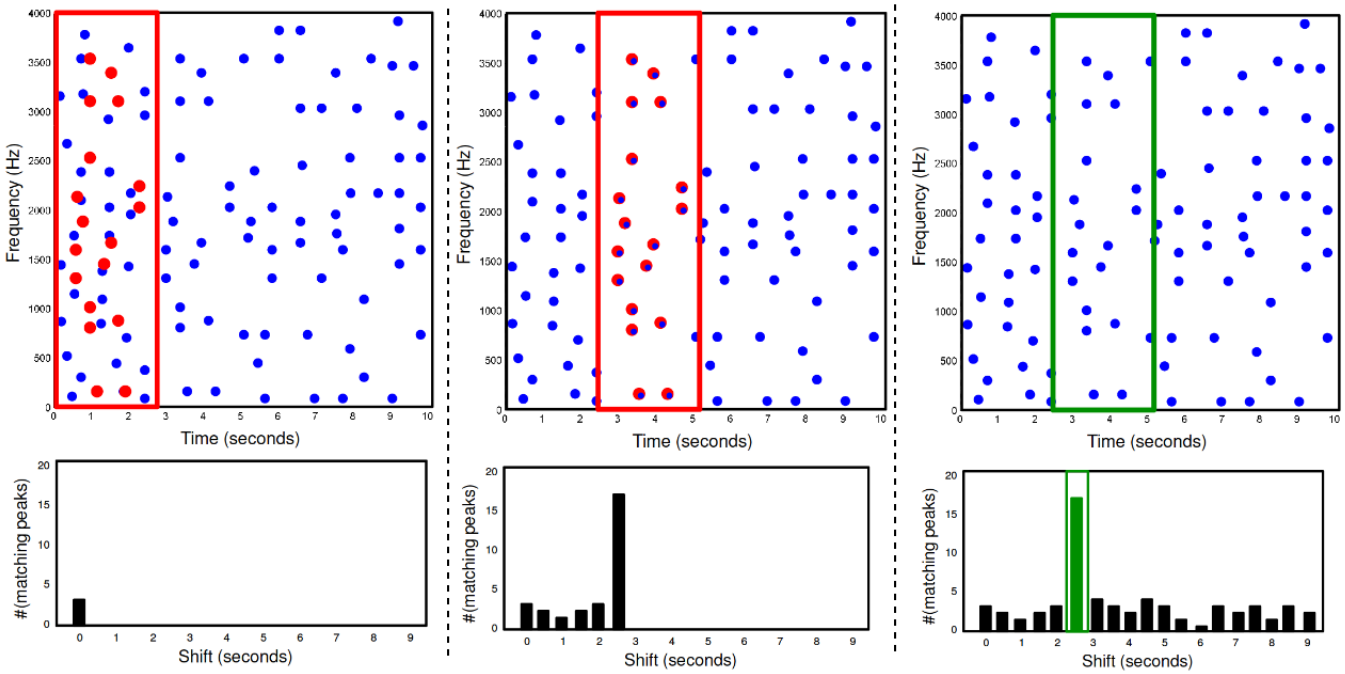

Figure 3.9: Illustration of sliding and matching [44]. Landmarks are treated as peaks in this figure.

The matching algorithm is implemented as follows:

- Extract all the hashes from the query track as described in Section 3.4. N_{query} is the number of hashes in the query track.

$$\{(t_n, h_n)\}, 0 \leq n < N_{query}$$

- For every hash h_n, fetch all the items in the corresponding bucket inside the hash table.

$$\{V_{n,i} = HashTable(i, h_n)\}, 0 \leq i < CountTable(h_n)$$

where $HashTable$ is the hash table and $CountTable$ is the count table we have created in Section 3.3.

- Retrieve track ID and time offset from $V_{n,i}$

$$V_{n,i} \Rightarrow (ID_{n,i}, T_{n,i})$$

So far, for every hash (t_n, h_n), there is a corresponding list, which is called reference list,

$$(t_n, h_n) \Rightarrow \{(ID_{n,i}, T_{n,i})\}, 0 \leq n < N_{query}, 0 \leq i < CountTable(h_n)$$

- Create a set composed of all the possible matching track ID by collecting all the track IDs in the above lists.

$$\{ID_k, 0 \leq k < K\}$$

- For every ID_k, scan the reference lists. If $ID_{n,i} == ID_k$, calculate the time difference $\delta t_k = T_{n,i} - t_n$. Then we compute a histogram of these δt_k. If there is a peak in the histogram and its value is above a threshold, a matching item is found.

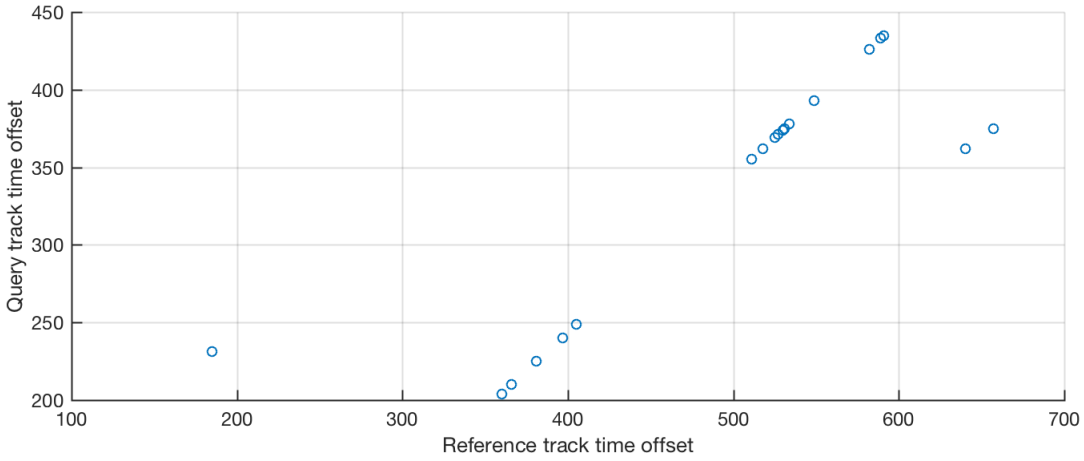

Figure 3.10: Scatterplot of matching hash time offsets, ($T_{n,i}$, t_n)

Figure 3.10 and Figure 3.11 shows a case where two tracks are matching. The scatterplot of matching hashes is usually very sparse because of the high specificity of the hash composed of pair of peaks. The appearance of a diagonal line indicates a match, which means there are a significant number of pairs of hashes that have

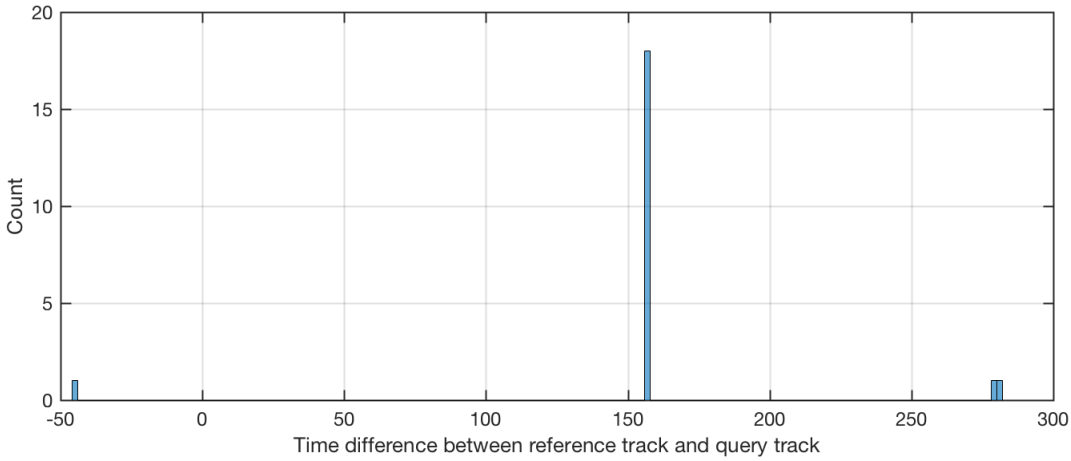

Figure 3.11: Histogram of differences of time offsets δt_k

same time offset differences. The peak bin in the histogram represents the number of points on the diagonal line, which means how many pairs of hashes are aligned in the reference track and query track. Its value is also a measure of similarity. In case when several matching tracks are found in the database, depending on the configuration, the output could be the one with the highest similarity or a list in the order of similarity from high to low.

Figure 3.12 shows the landmarks (blue) of a query track and the matching landmarks (red) of the correct reference track in the database. Because of additive noise, various distortions and time skew between query track and reference track, a lot of landmarks in the query track are not able to be found in the database. But the correct reference track can still be identified, due to the high specificity of landmarks.

3.6 Evaluation

We build a baseline audio fingerprinting system based on Ellis' work [22]. Before it is applied in noisy speech recognition, comprehensive evaluations about its performance are required. In this section, we will test the system under additive white noise, additive pub noise and different types of degradations.

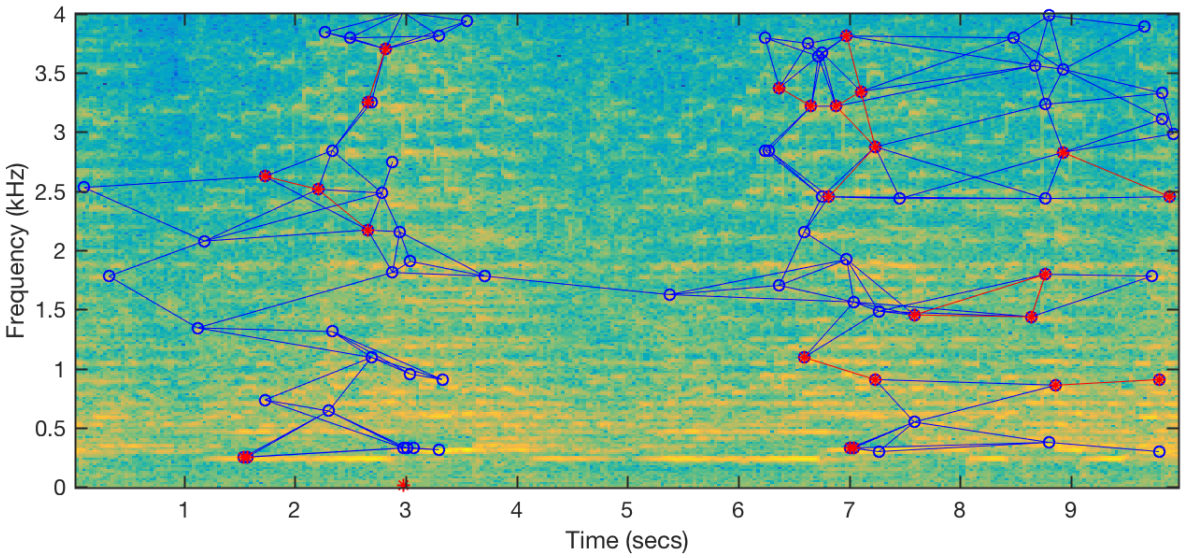

Figure 3.12: Matching landmarks

3.6.1 Training Dataset

GTZAN dataset[1] is used as training dataset in our experiments. It is created by G. Tzanetakis in [60] and then widely used in Music Information Retrieval (MIR). In this dataset, there are 1000 music audio excerpts classified into ten genres. The ten genres are Blues, Classical, Country, Disco, Hiphop, Jazz, Metal, Pop, Reggae and Rock. Every excerpt is 30 seconds long, sampled at 22050 Hz, 16-bit and monaural. The whole dataset is fed into the audio fingerprinting system and fingerprints are extracted from each track and then stored in the database along with metadata like the file ID.

3.6.2 Test Dataset

For each test case, there are 200 query tracks in its test dataset. Depending on the length requirement, every query track is 5, 10 or 15 seconds in length. They are taken from the middle of test track, which is randomly selected from the GTZAN dataset.

[1] `http://marsyasweb.appspot.com/download/data_sets/`

3.6.3 Audio Degradation Toolbox

Audio Degradation Toolbox[2] is a toolbox used to simulate various types of degradations. Using this toolbox, we test the baseline audio fingerprinting system under six real-world degradations, each of which consists of several basic degradation units as follows [42]:

- Live Recording. Apply Impulse Response of a large room and Add Noise.
- Radio Broadcast. Dynamic Range Compression to emulate the loudness of radio stations and Speed-up by 2%.
- Smartphone Playback. Apply Impulse Response of a smartphone speaker and Add Noise.
- Smartphone Recording. Apply Impulse Response of a smartphone microphone, Dynamic Range Compression to simulate the phone's auto-gain, Clipping and Add Noise.
- Strong MP3 Compression. MP3 Compression at 64 kbps.
- Vinyl. Apply Impulse Response of a common record player, Add Sound of player crackle, Wow Resample and Add Noise.

3.6.4 System Configuration

All the related parameters with their meanings and values are listed in Table 3.1. Note that we set different target hash density to train and test the system. Experiences show that larger density usually leads to better recognition rate within some limitation, but it also ends up with larger database and slower recognition speed. Setting a higher hash density only for the query track can help us get better recognition rate without these bad consequences.

3.6.5 Performance under Additive Noise

With the above configuration, the system performs well in environments with additive white and pub noise. Figure 3.13 and Figure 3.14 show the recognition rate when the system is tested with different query duration and SNR. During the test, the noise

[2] `https://code.soundsoftware.ac.uk/projects/audio-degradation-toolbox`

Parameter	**Meaning**	**Value**
f_{target}	Target sampling rate in Hz	8000
N_{win}	Window size	512
N_{hop}	Hop size	256
N_{FFT}	FFT size	512
p	The pole of the high-pass filter for spectrum	0.98
N_{peaks}	The maximum number of peaks per frame	5
f_{sd}	The spreading width applied to the masking skirt for each found peak	30
N_{bins}	Target zone height in bins	63
$N_{symbols}$	Target zone width in symbols	63
N_{fanout}	The maximum number of landmarks in a target zone	3
N_{t_1}	The number of bits used to represent t_1 in hash	14
N_{ID}	The number of bits used to represent track ID in hash	18
N_{f_1}	The number of bits used to represent f_1 in hash	8
$N_{\Delta t}$	The number of bits used to represent Δt in hash	6
$N_{\Delta f}$	The number of bits used to represent Δf in hash	6
N_{hash}	The number of buckets in the hash table	2^{20}
N_{bucket}	The bucket size in the hash table	100
$W_{matching}$	Width of matching bins	1
$T_{matching}$	Matching threshold	5
N_{report}	The number of matching items returned	1
$D_{training}$	The target density of hashes when we train the system	10
D_{test}	The target density of hashes when we test the system	20

Table 3.1: System configuration for audio fingerprinting performance test

is scaled to the desired SNR, then linearly added to the clean query track. The pub noise is recorded in a real noisy restaurant, which is part of the scene classification dataset as described in [26].

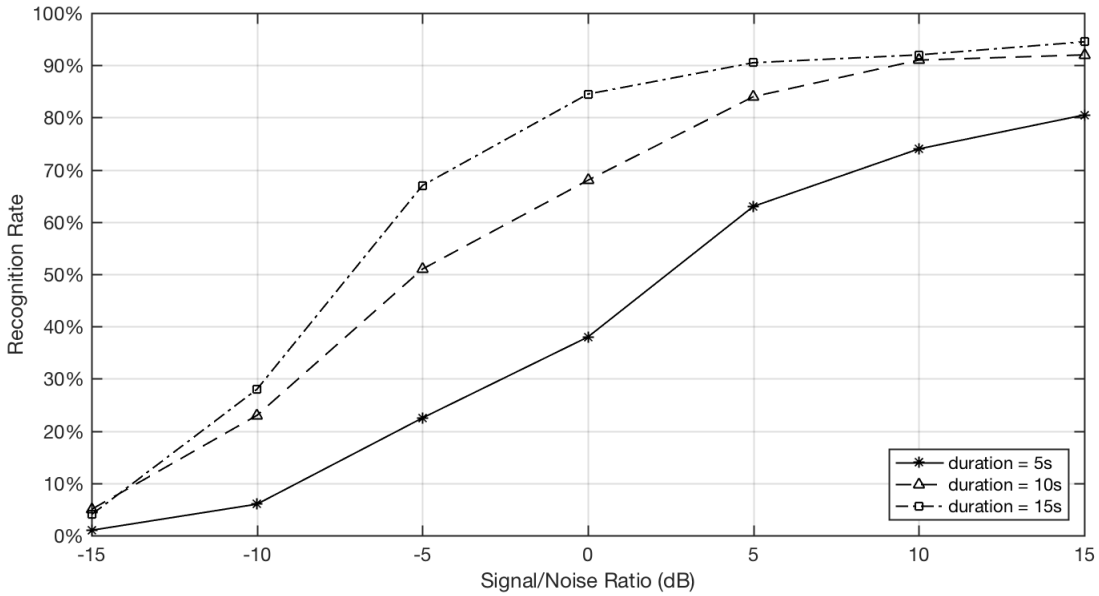

Figure 3.13: Recognition rate under white noise

Comparing Figure 3.13 and Figure 3.14, we can see that the performance is better when the system is tested in white noise than in pub noise. This is expected because white noise has same spectral intensity at different frequencies and pub noise is a combination of different sounds, which results in nonuniform spectral intensity. In addition, when the noise is unrelated with the clean audio, the spectrogram of a noisy audio is considered as a sum of spectrograms of the clean audio and the noise. So the pub noise will introduce more spurious peaks and also mask some salient peaks of the clean query audio.

Both Figure 3.13 and Figure 3.14 show that the increasing of SNR leads to better recognition rate. Higher SNR means less noise, which leads to fewer spurious peaks and more real peaks surviving in the spectrogram of the query track. In environment with pub noise, when the query length is 15 seconds, as the SNR increases from -15 dB to 15 dB, the recognition rate increases from 3.0% to 93.5%.

The two figures also shows longer query audio results in better performance. In Figure 3.14, when SNR is 0 dB, the recognition rate is 11%, 62% and 81% for query samples of 5, 10 and 15 seconds respectively. When we slide the constellation map

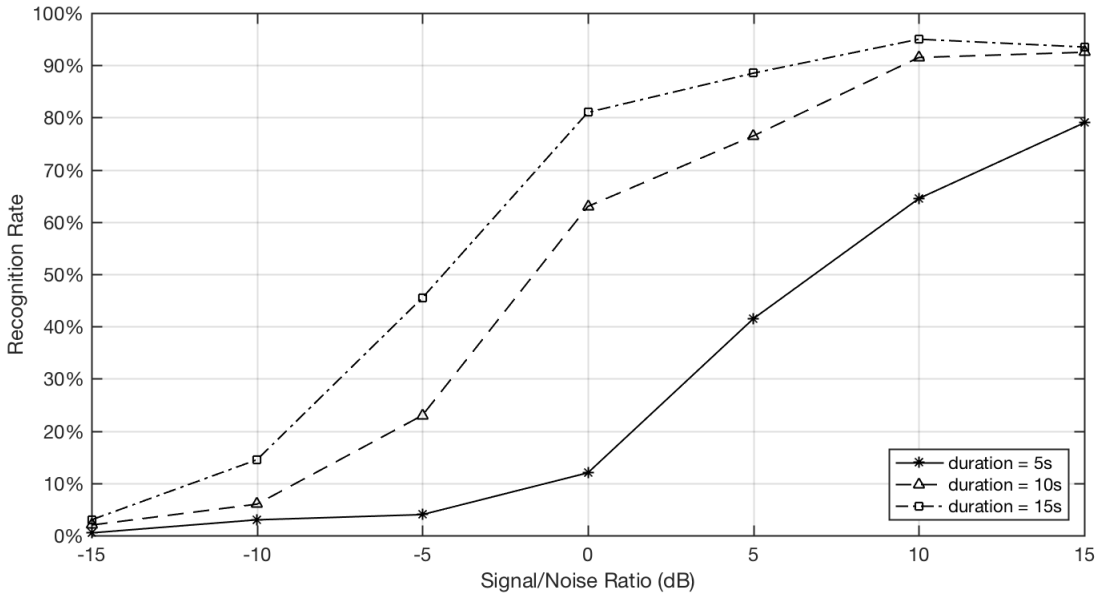

Figure 3.14: Recognition rate under pub noise

of the query track over the constellation map of a reference track, they will be more coinciding points if the query track is longer.

3.6.6 Performance under Degradations

With the help of Audio Degradation Toolbox, we test the system under degradations. From the Figure 3.15, we can see that the system is quite robust against various types of real-world degradations except Radio Broadcast and a longer query audio sample always helps improve the performance. When the query length is 15 seconds, the recognition rate is over 90% for Live Recording, Smartphone Playback, Smartphone Recording and String MP3 Compression. This recognition rate is almost same as testing with the clean query track without any degradations. However, the recognition rate for Vinyl is a little bit worse, which is about 85%. And Radio Broadcast is the worst case, whose recognition rate is only 10%. Having a closer look, we find that the reason is both of them have an unique degradation unit individually, Wow Resample in Vinyl and Speed-up in Radio. Wow Resample is similar to Speed-up, but its resampling rate is time-dependent, not constant. So it seems this baseline system is not robust enough to the degradation Speed-up. Specific test has been done to test the system's robustness to this special degradation in Section 3.6.7.

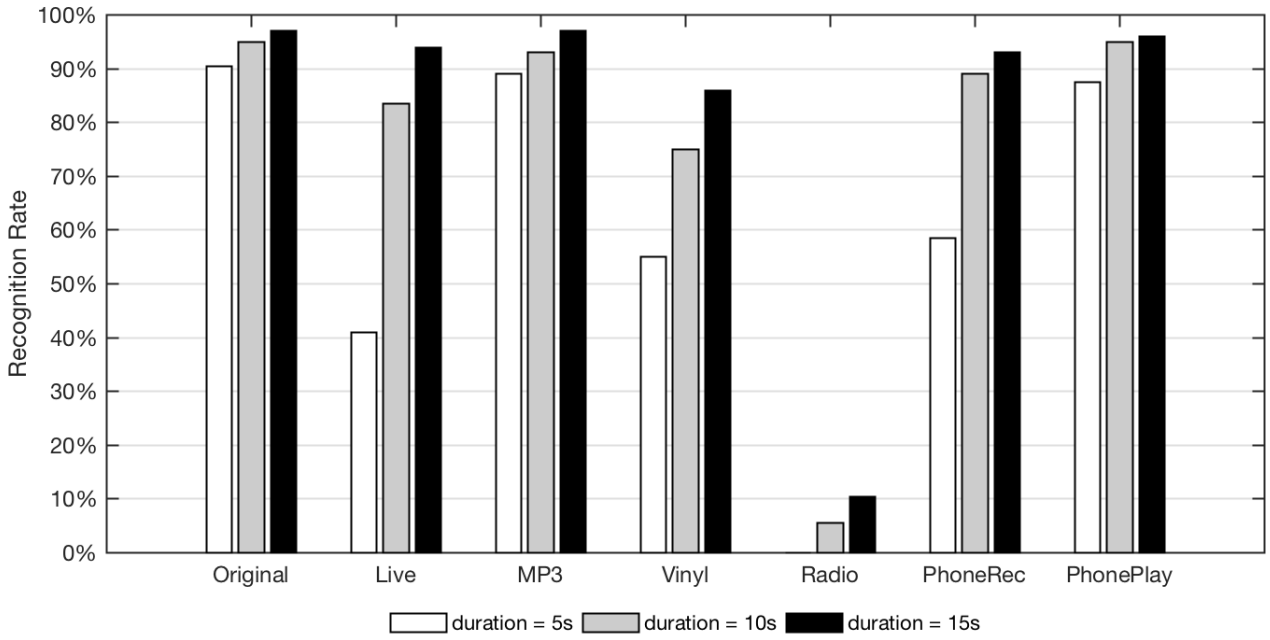

Figure 3.15: Recognition rate under different types of degradations

Note that the recognition rate is not 100% even for the original audio in Figure 3.15 even when its duration is 15 seconds, which is not expected. Looking into the test log and listening to the false positive results, we find that they are actually correct results. This happens due to the fault of GTZAN dataset. As stated in [57], there are repetitions in this dataset. For instance, in genre Disco, disco.00050.au, disco.00051.au and disco.00070.au are exactly same audio files. So if we take 15 seconds from disco.00050.au as query sample, it could be recognized as any of them. In spite of this fault, GTZAN dataset is still a good dataset for us to evaluate the system. Since the query samples are always taken from same subset of the GTZAN dataset, all the evaluations are affected in the same way.

3.6.7 Sensitivity to Speed-up

In the degradation unit Speed-up, the audio signal is expanded or compressed along the time axis, which will result in pitch shifting. Assume speed changing (speed-up or slow-down) only stretches the spectrogram and the patten of the peaks does not change, speed changing affects audio fingerprinting in three aspects:

- For a landmark $t_1 : [f_1, \Delta f, \Delta t]$, Δt and Δf are changed. With the configuration in Table 3.1, the maximum value for them is 63. If the speed changing is 2%, they may be changed by 1 ($63 \times 2\% = 1.26$);
- f_1 will be changed. Since the maximal value for f_1 is 255, 2% speed changing results in a maximum change of 5 ($255 \times 2\% = 5.1$);
- t_1 will also be changed. It will affect the filtering step when we match hashes since the time offset differences are changed. For a query audio of 15 seconds, 2% speed changing leads to a maximum change of 0.3 seconds for t_1. Since the unit on time axis is 0.032 second ($256/8000 = 0.032$), the maximum change for t_1 is almost 10 ($0.3/0.032 = 9.375$) after quantization. With such a big change, these landmarks will be filtered out when counting the coinciding landmarks even if they are actually matching with the reference landmarks.

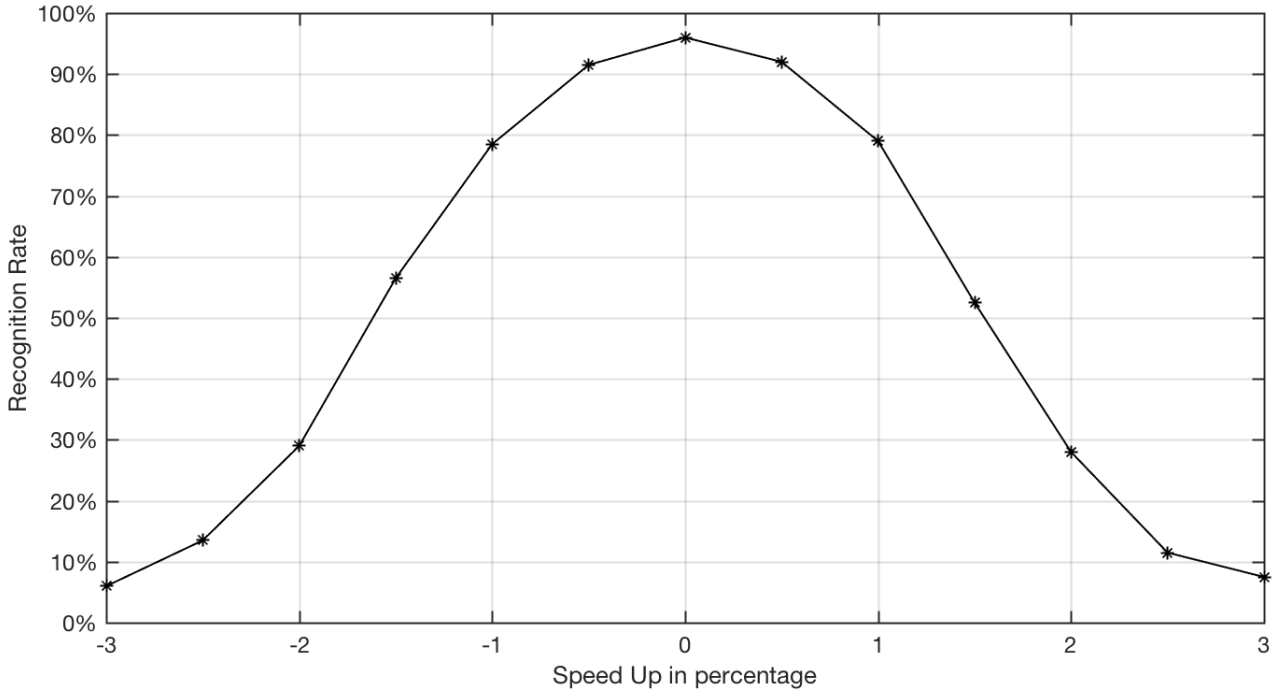

Figure 3.16: Sensitivity to speed-up

Figure 3.16 shows the sensitivity of the baseline system to the degradation Speed-up. Positive value on x axis means the audio file is compressed, while negative value means it is expanded. The recognition rate drops along with the increasing of speed

change. 1% speed changing is the limitation of the system, otherwise the result is not reliable.

3.7 Summary

These experiments show that the landmark based audio fingerprinting system is robust to additive noise and various degradations except pitch shifting. This robustness is due to its unique fingerprint scheme based on spectrogram peaks. These peaks can survive ambient noise and satisfy the property of linear superposition. However, when there is pitch shifting, the coordinate of peaks may change, ending up with different landmarks. So the system's high sensitivity to pitch shifting is expected.

In Chapter 4, we are going to apply this audio fingerprinting system to speech reconstruction.

Chapter 4

Experiments with Speech Reconstruction

In this chapter, we carry out experiments to reconstruct sentences using clean speeches from the same speaker. We will first talk about the motivations behind these experiments and describe the experimental dataset and the evaluation methodology. Then we propose three strategies to improve the performance of audio fingerprinting when it is used in speech reconstruction. The strategies include pre-emphasis, robust landmark scheme to pitch shifting and morphological peak extraction. At last, the results of speech reconstruction and the analysis are presented.

4.1 Motivation

Our new scheme of speech recognition is based on speech reconstruction, in which an essential step is to find similar fragment in clean dataset to the noisy query fragment using audio fingerprinting. The accuracy of audio fingerprinting to a large extent decides the quality of the reconstructed speech, so we need to test its performance in this new application scenario. In contrast to traditional application scenarios, this scenario poses four new challenges:

- The clean version of the query sample is not included in the training dataset. Clean version means there is no noise and distortions. For traditional audio fingerprinting, a query sample that has a matching item can always find its clean version in the database.

- The query sample is much shorter. Current commercial applications like Shazam

and SoundHound require the query length to be longer than 10 seconds. In our new application scenario, this length is impossible. With normal American talking speed one word occupies about 0.3 seconds in a sentence and 10 seconds will contain more than 30 words. It is very hard to find another sentence in the dataset that contains the same words in the same order. In our experiments, the query length will be shorter than 1 second.

- For two sentences consisting of same words generated by the same speaker, their spectrogram may be different. People will unconsciously change their frequencies to talk. Even if it is not distinguishable for human ear, there are much differences between the spectrograms. However, if the dataset is large enough, it is likely to contain some similar fragments.

- For two similar speech fragments from two different sentences, the distortions between the two fragments are much more complicated than distortions in near-duplicate detections problems. For example, they exhibit strong non-linear temporal distortion, which traditional audio fingerprinting algorithms typically do not handle well.

The above challenges make the performance of the baseline audio fingerprinting system deteriorate greatly. To improve the performance, we propose three strategies and present experimental results and analysis in this chapter.

4.2 Dataset

The experiments are based on GRID corpus[1]. It is a large multitalker audiovisual sentence corpus composed of high-quality audio recordings of 1000 sentences spoken by each of 34 talkers (16 females and 18 males). All the sentences consist of 6 words and follow the form $< command : 4 >< color : 4 >< preposition : 4 >< letter : 25 >< digit : 10 >< adverb : 4 >$. For example, "place white at L 3 now". The numbers in brackets indicate the number of choices for that word. All possible words are listed in Table 4.1. The words marked with asterisk are keywords when this corpus is used in speech recognition.

We choose corpus of one of the 34 talkers, the 20th talker, as the experimental dataset. It does not matter which talker we choose because we get similar results

[1] `http://spandh.dcs.shef.ac.uk/gridcorpus/`

Table 4.1: All possible words for GRID corpus[19]

command	color*	preposition	letter	digit*	adverb
bin	blue	at	A–Z	1–9, zero	again
lay	green	by	excluding W		now
place	red	in			please
set	white	with			soon

when experiments are carried out on corpus from different talkers. Our experiments are designed for speech reconstruction using clean speech from the same speaker, so corpus of one talker is enough. The dataset is represented with

$$\{SEN_i, 0 \leq i < 1000\}$$

4.3 Evaluation Methodology

We use the accuracy of the similar segment searching as our evaluation metric. Before we replace a query segment, we need to use audio fingerprinting to search the database for the most similar segment. The accuracy of the searching result to a large extent decides the performance of the whole system. The accuracy is defined as

$$Accuracy = \frac{Number\ of\ correct\ retrieved\ segments}{Number\ of\ query\ segments}$$

When a query segment is sent to an audio fingerprinting system, we cannot find the exactly same segment in the database, since the query segment is not included in the training dataset. If the retrieved segment is similar with the query segment, which means they convey same words, we count it as a correct retrieved segment.

Both the training dataset and test dataset come from the experimental dataset. Every sentence in the experimental dataset is divided into two segments equally, SEN_{i0} and SEN_{i1}. The first segments of all sentences make up of the test dataset.

$$\{SEN_{i0}, 0 \leq i < 1000\}$$

For every query segment, there is a corresponding training dataset, which includes

all the sentences in the experimental dataset except the sentence which the query segment comes from,

$$\{SEN_j, 0 \leq j < 1000 \; and \; j \neq i\}$$

The logic behind the design of training dataset and test dataset is quite similar to 1000-fold cross-validation.

Although the query segment itself is not in the training dataset, there are still many other similar segments. The dataset consists of 1000 sentences and there are only $4 \times 4 \times 4 = 16$ combinations for the first three words. Assume the first three words occupy the first half of a sentence, there are $1000/16 \approx 15$ similar segments for each combination in the experimental dataset.

Considering the first three words and the rest three words may not be perfectly equally distributed on the sentence, the first half of a sentence may contain part of or more than three words. In the following experiments, we consider only the first two words. When the query sentence, which query segment comes from, and the match result sentence have same first two words, we consider it as a correct result. This is a rough evaluation for a key step in the whole system.

4.4 Pre-emphasis

Pre-emphasis is a popular technique in speech recognition. It can strengthen the amplitude in the high frequencies. As shown in the spectrogram of speech signal in Figure 2.5, there are more energy in the lower frequencies than the higher frequencies. This phenomenon is caused by the nature of vocal cords. However, the third formants of vowels usually exist in high frequencies, which are useful for a system to distinguish different phones. Strengthening the energy in high frequencies makes spectrogram peaks of these formants more available for peak extraction block in audio fingerprinting system.

Normally the pre-emphasis is implemented with a first order high-pass filter. Its filter equation is

$$y(n) = x(n) - \alpha \cdot x(n-1), 0.9 \leq \alpha \leq 1.0$$

A typical value for α in speech recognition is 0.97. The effect of pre-emphasis is shown in Figure 4.1. After pre-emphasis, the energy is distributed more uniformly over all the frequencies while the positions of peaks remain unchanged.

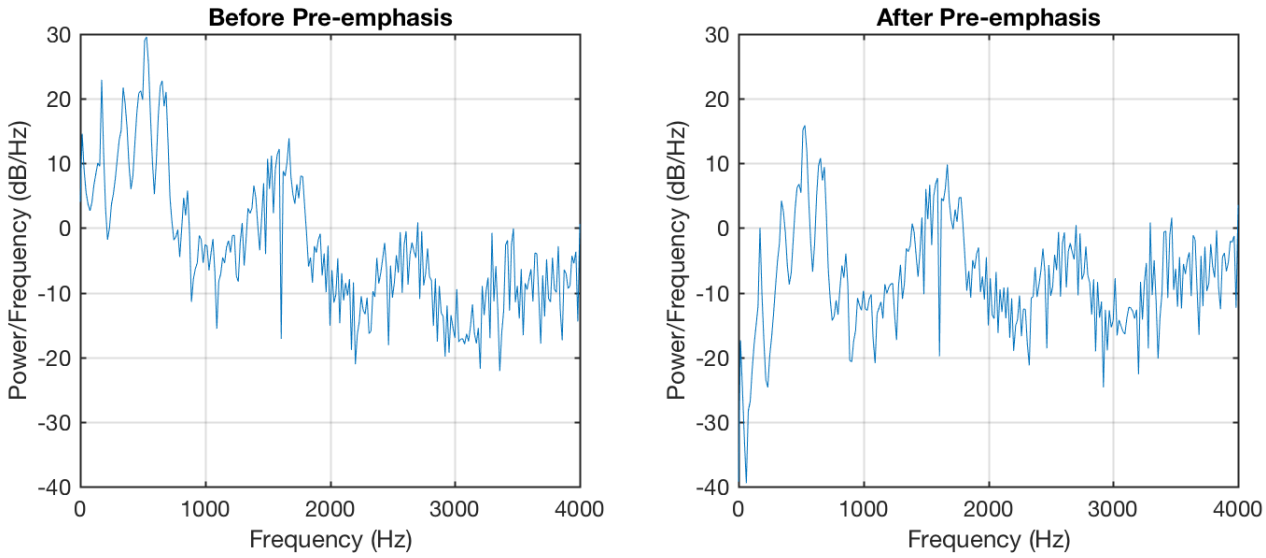

Figure 4.1: Spectrum of the vowel [ε] before pre-emphasis and after pre-emphasis

4.5 Robust Landmark Scheme to Pitch Shifting

In an audio fingerprinting system, pitch shifting is caused by expanding or compressing the query track in length in comparison with the matching reference track in the database. It will raise or lower all the frequency components with the same ratio, resulting in the change of coordinates of peaks. As shown in Figure 3.16, the limitation of the baseline audio fingerprinting system is 1% even if the query length is 15 seconds and the recognition rate drops rapidly beyond this limitation.

There is pitch shifting for similar speech segments in the database. In the GRID corpus, a talker does not always use the same speed to speak a sentence, even a word. Figure 4.2 shows the durations of the first two words "bin blue" spoke by the 20th talker. We can see that there are many different durations for the two words from 0.9 seconds to 1.8 seconds, which means there is pitch shifting among them. To improve the similar segment search accuracy, we need to improve the robustness of the system to pitch shifting.

Inspired by [24], a robust landmark scheme is proposed to overcome the pitch shifting. In this scheme, there are two major modifications, Constant-Q Transform (CQT) spectrogram and new landmark format. Recall that a landmark is represented with the following expression in the baseline system,

$$t_1 : [f_1, \Delta f, \Delta t], \Delta f = f_2 - f_1, \Delta t = t_2 - t_1$$

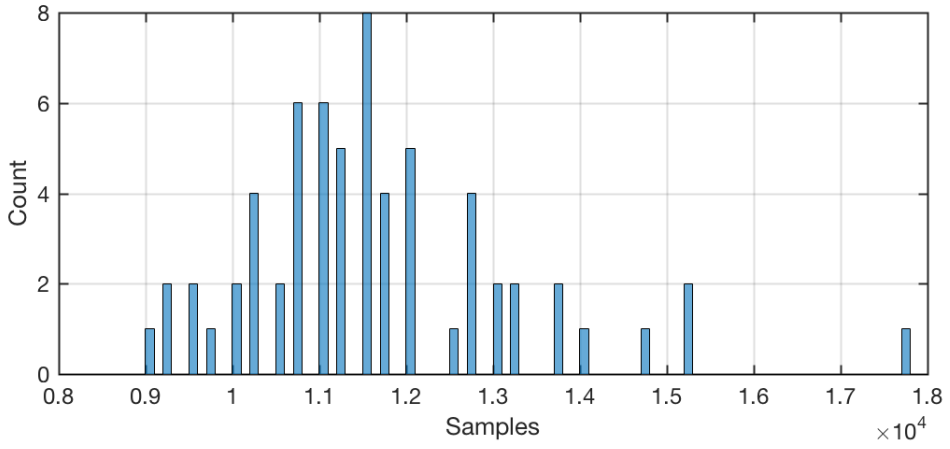

Figure 4.2: Histogram of the durations of "bin blue" spoke by the 20th talker in GRID corpus

When pitch shifting occurs, the vertical positions of all peaks will be multiplied with a factor k. So the landmark is changed to

$$t_1 : [k \cdot f_1, k \cdot \Delta f, \Delta t]$$

The changes will cause mismatch between the query track and the reference track. CQT spectrogram and new landmark format can mitigate the effect of pitch shifting by removing these changes.

CQT spectrogram is similar to FFT spectrogram but with a log frequency representation [10]. A good way to understand CQT transform is to think that there are a series of logarithmically spaced filters [63] as following,

$$\delta f_k = 2^{1/n} \cdot \delta f_{k-1} = \left(2^{1/n}\right)^k \cdot \delta f_{min}$$

Here n is the number of filters per octave, δf_k is the bandwidth of the k-th filter and δf_{min} is the smallest bandwidth. An efficient CQT algorithm is proposed in [11], which transforms a DFT transform to CQT transform. A comparison of FFT spectrogram and CQT spectrogram is provided in Figure 4.3. We can see that CQT spectrogram provides higher resolution between 0 Hz and 2,000 Hz where the first and second formants of vowels locate in. FFT spectrogram is computed with STFT directly. Its resolution is uniform over the entire frequency range and all frequency components are treated equally. However, CQT spectrogram is computed with a variant form of

STFT, where FFT spectrum is replaced with CQT spectrum. With a log frequency representation, CQT spectrogram can provide non-uniform frequency resolution for audio signal analysis.

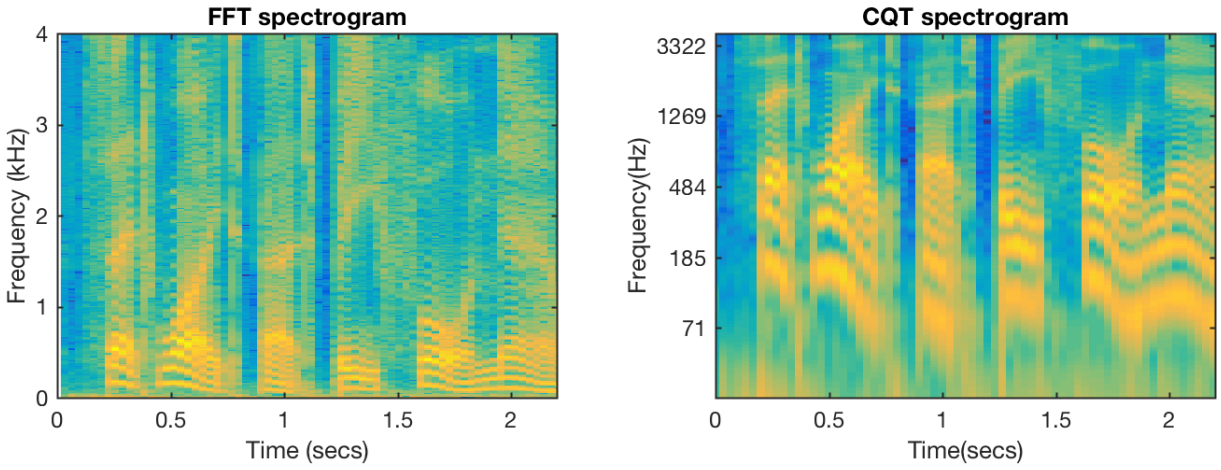

Figure 4.3: FFT spectrogram and CQT spectrogram

When we use CQT spectrogram, as the frequency bins are geometrically spaced, the position of peaks will only be shifted with a factor k'. In this case, the landmark is changed to

$$t_1 : [f_1 + k', \Delta f, \Delta t]$$

Since $\Delta f = (f_2 + k') - (f_1 + k') = f_2 - f_1$, the change of the second component in $[f_1, \Delta f, \Delta t]$ is removed.

The new landmark format can remove the change of the first component. The proposed format is as follows,

$$t_1 : [f_1 >> m, \Delta f, \Delta t]$$

The first component $(f_1 >> m)$ is a sub-resolved version of f_1. An advantage of this format is that the first component remains unchanged if $(k' >> m)$ equals to 0.

$$\begin{aligned}(f_1 + k') >> m &= (f_1 >> m) + (k' >> m) \\ &= (f_1 >> m)\end{aligned}$$

A typical value for m is 2, which makes the system robust to pitch shifting less than 2%. With the robust landmark scheme, carry out the experiment in Section 3.6.7

again and get the result shown in Figure 4.4. The recognition rate does not drop until the pitch shifting goes beyond 2%.

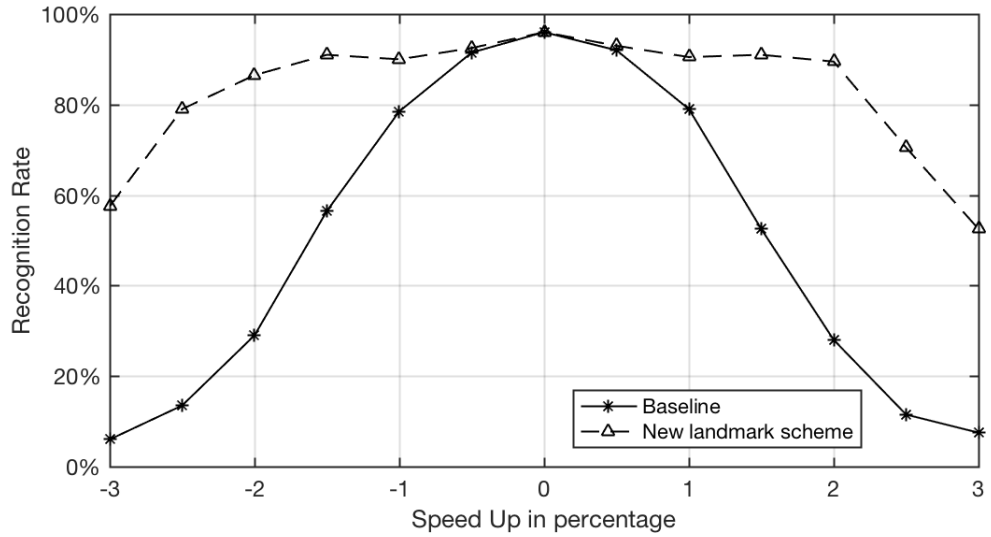

Figure 4.4: Recognition rate of audio fingerprinting with new landmark scheme under different pitch shifting (speed changing)

4.6 Morphological Peak Extraction

Another way to implement peak extraction is to use morphology, which is a set of image processing operations. Morphological operations process an input image and create an output image of the same size based on a structuring element. Each pixel of the output image is computed by comparing itself with its neighbors in a neighborhood defined by the structuring element. During a morphological operation, the structuring element will iterate over every pixel of the input image. As an example, a cross-shaped structuring element is shown in Figure 4.5. It is a boolean two-dimension array, in which only true pixels are used to compute the output pixel. The point marked with "Origin" is the pixel being processed. The value of this pixel on the output image depends on the comparison result between it and its four neighbours.

Dilation is one of the useful morphological operations for peaks extraction. After dilation, the value for the output pixel is the maximum value in the processing pixel's neighborhood specified by the structuring element. Take the cross-shaped structuring

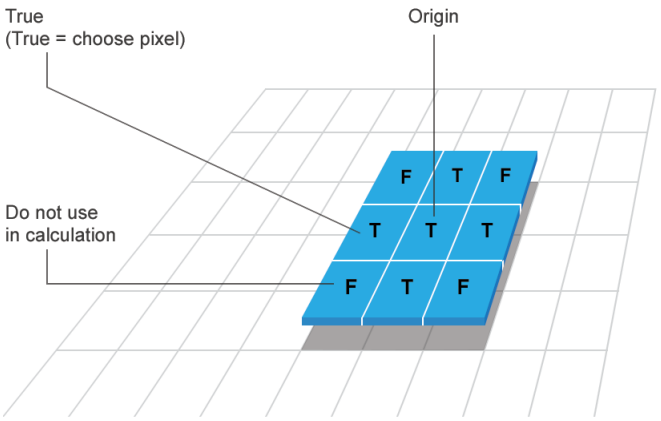

Figure 4.5: A cross-shaped '+' structuring element [56]

element for an example,

$$Dilation[S(f,t)] = max\Big(S(f,t), S(f+1,t), S(f-1,t), S(f,t+1), S(f,t-1)\Big)$$

Considering the spectrogram as an image, the procedure of morphological peak extraction is described as follows:

- Define a spherical structuring element E whose radius is r. Radius r is a parameter related with the density of peaks. A small radius results in large density, although the relationship is non-linear.
- Dilate the spectrogram $S(f,t)$ with E. The output image is $S_{dilation}(f,t)$, whose size is same as $S(f,t)$.
- Compare these two image by pixel, $S(f,t)$ and $S_{dilation}(f,t)$. For every pixel (f,t), if $S(f,t) == S_{dilation}(f,t)$, a new peak with coordinate (f,t) is found.
- Collect all the above coordinates and return them as the final result of peak extraction.

4.7 Results and Analysis

4.7.1 Parameters

One of the challenges in finding similar clean speech segment is that the query sample is very short. To satisfy this requirement, some parameters must be adjusted. Table

4.2 lists all the updated parameters in this scenario and new parameters for the three strategies. Parameters not listed in this table use the same setting as Table 3.1. f_{sd}, N_{fanout}, $D_{training}$ and D_{test} are adjusted to increase the density of hashes. Larger density is required because the query length is shorter. $N_{symbols}$ is updated to make the target zone width smaller than the length of the spectrogram of the query segment. r is a new parameter to make hash density consistent when morphological peak extraction is used. And m is a new parameter to control the system's sensitivity to pitch shifting.

Parameter	**Meaning**	**Value**
f_{sd}	The spreading width applied to the masking skirt for each found peak	8
N_{fanout}	The maximum number of landmarks in a target zone	6
$D_{training}$	The target density of hashes when we train the system	50
D_{test}	The target density of hashes when we test the system	50
$N_{symbols}$	Target zone width in symbols	7
r	The radius of the spherical structuring element	5
m	This shift value for f_1 when robust landmark scheme is used	2

Table 4.2: System configuration for audio fingerprinting in speech reconstruction

4.7.2 Clean Speech Reconstruction

Using these parameters, we test the system with various combinations of the three strategies we have described above in clean environments. Table 4.3 shows the experimental results, where 'YES' means this strategy is used and 'NO' means not used. The accuracy of the baseline system is 35.6%, which is much lower than its accuracy in traditional applications. This is caused by the new challenges in this application scenario as described in Section 4.1. Test 1, Test 2 and Test 3 are designed to test the improvement that the three strategies can bring individually. Their results show only robust landmark scheme improves the accuracy. Test 4, Test 5, Test 6 and Test 7 test different combinations of the three strategies. We can see that the highest accuracy is achieved when pre-emphasis and robust landmark scheme are applied. As the landmark based audio fingerprinting algorithm is proposed initially for music identification, pre-emphasis makes it more suitable for speech. It boosts the energy in high frequencies, which helps the system to extract peaks of the third and the

fourth formant of vowels. In addition, robust landmark scheme makes the system robust to pitch shifting. This is beneficial because people always speak with different speed, which results in pitch shifting. Morphological peak extraction does not help to improve the accuracy, even when the density of hashes is same as the baseline system (Gaussian peak extraction is used).

Test Case	**Strategies**			**Accuracy**
	Pre-emphasis	Robust Landmark Scheme	Morphological Peak Extraction	
Baseline	NO	NO	NO	35.6%
Test 1	YES	NO	NO	32.4%
Test 2	NO	YES	NO	44.4%
Test 3	NO	NO	YES	27.2%
Test 4	**YES**	**YES**	**NO**	**46.4%**
Test 5	YES	NO	YES	28.4%
Test 6	NO	YES	YES	33.6%
Test 7	YES	YES	YES	34.0%

Table 4.3: Results with different combination of strategies

The best accuracy in Table 4.3 is not very good, but still promising. If we randomly choose a track from the database, the accuracy is 6.25% (1/16). We improve the accuracy to 46.4% by combining the baseline audio fingerprinting system with pre-emphasis and robust landmark scheme.

4.7.3 Noisy Speech Reconstruction

In this part, we evaluate the accuracy of finding clean speech segment in noisy environments. The pub noise same as in Section 3.6.5 is added to the speech track before it is fed to the audio fingerprinting system.

We evaluate two audio fingerprinting systems, baseline system and improved system. Since in Section 4.7.2 we have found the optimal combination of the strategies is pre-emphasis and robust landmark scheme, improved system is built by applying these two strategies to the baseline system. Figure 4.6 shows the evaluation result. Blue line represents the accuracy of the baseline system, while the red line represents the accuracy of the improved system. As the SNR increases, their accuracies increase.

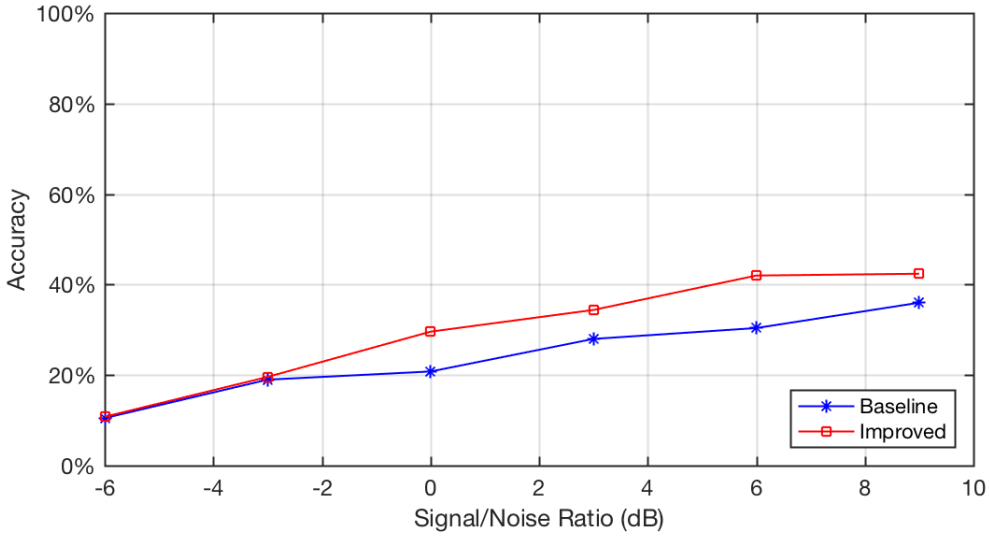

Figure 4.6: Accuracy under pub noise

This is consistent with the evaluation result of the baseline audio fingerprinting system in Section 3.6.5. At the same time, note that the accuracy of the improved system is always better than the baseline system at different SNR. This confirms that the improved system works better than the baseline system in noisy environments.

4.8 Summary

In this chapter, we carry out experiments to test the performance of audio fingerprinting in speech reconstruction. To overcome new challenges when audio fingerprinting is used to find similar speech segments in a large set of clean utterances for a noisy speech segment, we propose three strategies: pre-emphasis, robust landmark scheme and morphological peak extraction. Clean speech reconstruction experiments show that best performance is achieved when pre-emphasis and robust landmark scheme are applied, and noisy speech reconstruction experiments show this improved audio fingerprinting system also performs better than the baseline system in noisy environments.

After speech reconstruction, the reconstructed speech is fed to a traditional speech recognition system. To evaluate the performance of the whole system, from audio fingerprinting to speech reconstruction to speech recognition, more experiments are carried out in Chapter 5.

Chapter 5

Speech Recognition in Noisy Environments

In this chapter, we build a speech recognition system in noisy environments and test its performance. We will build a baseline speech recognition system at first. Then we will try to reconstruct clean speech from the noisy speech using audio fingerprinting and send the reconstructed speech to the baseline speech recognition system. At last, we will present the evaluation results and analysis.

5.1 Dataset

We use part of Track 1 in the 2nd CHiME Speech Separation and Recognition Challenge[1] as our dataset. This dataset consists of two parts, training dataset and test dataset. All of them are taken from the GRID corpus.

- Training dataset. It consists of 17000 clean utterances by taking 500 utterances from each of 34 talkers. It is used as training data for both the audio fingerprinting subsystem and the speech recognition subsystem.
- Test dataset. It consists of 600 noisy reverberated utterances. To generate this dataset, an initial dataset composed of 600 utterances is created by taking 17 or 18 utterances from each of 34 talkers. Then, these utterances are convolved with a set of of binaural room impulse responses to simulate speaker movements and reverberation in a family living room. At last, each utterance is mixed with

[1] `http://spandh.dcs.shef.ac.uk/chime_challenge/chime2013/index.html`

binaural recordings of genuine room noise made over a period of days in the same room at each of 6 ranges of SNR, which are -6 dB, -3 dB, 0 dB, 3 dB, 6 dB and 9 dB. More details about the generation of this dataset can be found in [18].

They are all 16 bit WAV files, stereo and sampled at 16kHz. Note that none of the utterance in the training dataset will appear in the test dataset again.

5.2 Baseline Speech Recognition System

We build a baseline speech recognition system based on the scripts[2] provided by V. Emmanuel for the 2nd CHiME Speech Separation and Recognition Challenge. The scripts take advantage of Hidden Markov Model Toolkit (HTK)[3] and are written for Linux/Unix platforms. This baseline system does not contain any noise suppression preprocessing. Using this system, we can do the following things:

- Train a speech recognition system from specified training dataset;
- Transcribe speech in the test dataset;
- Score the recognition results in terms of keyword recognition rate.

When we train the system, the speech signals are transformed to standard 39-dimensional MFCCs at first, including 12 cepstral coefficients, 12 delta cepstral coefficients, 12 double delta cepstral coefficients, 1 energy coefficient, 1 delta energy coefficient and 1 double delta energy coefficient. If the input speech signal is binaural, it will be converted to monaural signal by averaging the two channels.

Every word is modeled as whole-word Hidden Markov Model (HMM) with a left-to-right model topology. For every phone, there are two states and every state is modeled with 7 Gaussian mixtures. The number of states for all the words in the dictionary of the system is following:

- 4 states: one two three eight a b c d e f g h i j k l m n o p q r s t u v x y z at by in.
- 6 states: blue green red white bin lay place set with now please soon four five six nine.

[2] `http://spandh.dcs.shef.ac.uk/chime_challenge/chime2013/chime2_task1.html`
[3] `http://htk.eng.cam.ac.uk`

- 8 states: zero again.
- 10 states: seven.

When we transcribe speech, the following grammar, which is formulated in the language model, is used:

$$\$command\ \$color\ \$preposition\ \$letter\ \$digit\ \$adverb$$

For each above word type, the corresponding words are listed in Table 4.1.

The score of recognition results depends on two keywords, <color> and <digit>. The score of an utterance is the number of correct keywords, which is 0, 1 or 2. The total score is the average of the scores across all the utterances in the specified test dataset, which is calculated as a percentage.

$$score = \frac{\sum_{i=0}^{N-1} s_i}{2 \cdot N} \cdot 100\%$$

where s_i is the score for the ith utterance, $s_i \in \{0, 1, 2\}$, and N is the total number of utterances.

5.3 Application of Audio Fingerprinting

Figure 5.1 shows how audio fingerprinting is applied in noisy speech recognition. The main workflow is following:

1. The noisy speech is divided into segments of fixed length without overlap. Ideally the length of segment should be the average length of a phoneme in the dictionary, but it is not feasible for landmark based audio fingerprinting, as we cannot extract enough landmarks from such a short segment to uniquely represent it. Furthermore, the target zone is often longer than a phoneme. Typically the length of a segment is set to contain several words.

2. Each segment is fed to an audio fingerprinting system to find the most similar segment in the database. The audio fingerprinting system has been trained with a large set of clean speech.

3. If the number of hash hits between the query segment and the retrieved one is above a threshold, the query segment is replaced with the retrieved one.

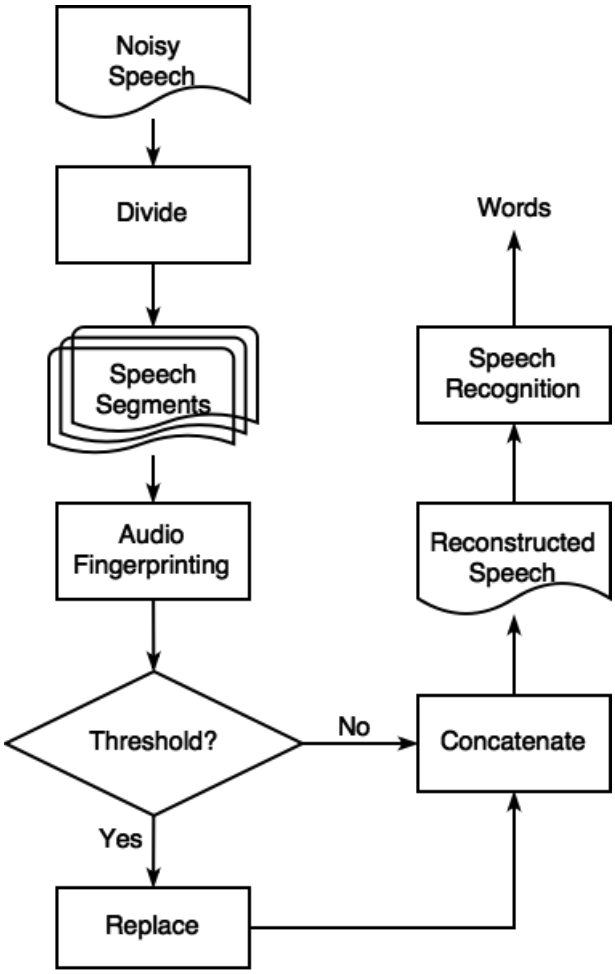

Figure 5.1: Application of audio fingerprinting in speech recognition

Otherwise it is kept as it was. The threshold is a parameter to minimize the false positive rate of the retrieved segment by the audio fingerprinting system.

4. Concatenate all the segments, no matter whether it has been replaced or not. So far we have reconstruct the noisy speech and some segments in the noisy speech have been replaced with clean segments.

5. The reconstructed speech is fed to a traditional speech recognition system. The speech recognition system has been training with the same set of clean speech as the audio fingerprinting system.

5.4 Results and Analysis

To evaluate the proposed speech recognition scheme, a system is built as described in Section 5.3. The training dataset presented in Section 5.3 is used to train both the audio fingerprinting system and the speech recognition system. The former one is implemented with the improved audio fingerprinting system and the latter one is just the baseline speech recognition system. When an noisy utterance is fed to the system, it is divided into two segments equally at first, so there are approximately three words in each segment. Then the audio fingerprinting system will try to find similar clean segment for each query segment in the clean dataset. Noisy speech segment in the utterance is replaced with clean segment if the number of hash hits between them is above a threshold. After conditional replacement, the two segments of the noisy utterance are concatenated together and sent to the speech recognition system.

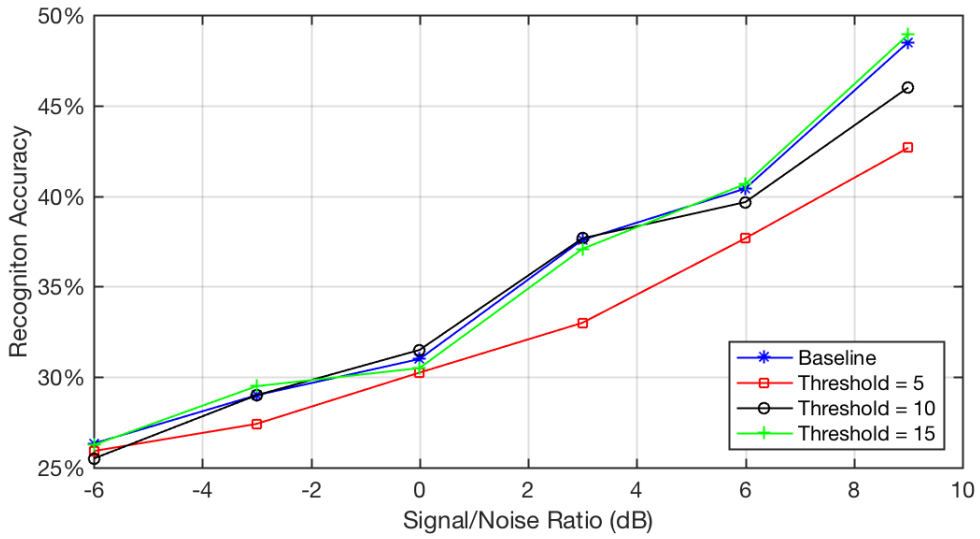

Figure 5.2: Recognition accuracy with different similarity thresholds

Figure 5.2 presents the recognition accuracy with different similarity thresholds. The blue line represents the recognition accuracy of the baseline speech recognition system without speech reconstruction. As the SNR increases, its accuracy increases. This is expected because when SNR is higher, there is less mismatch between the speech signal used to train the acoustic model and the input speech signal. The best accuracy 48.5% is achieved when SNR is 9 dB. Although SNR is relatively high in this case, the test utterance still suffers strong reverberation. That is why the best accuracy is much lower than 100%. When audio fingerprinting is applied and

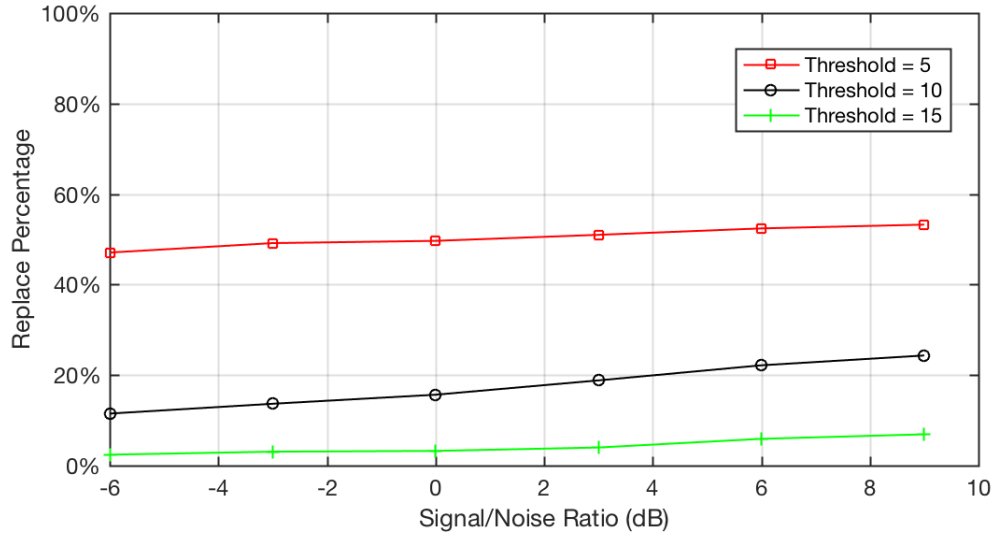

Figure 5.3: Replace percentage with different similarity thresholds

the threshold is 5, represented by the red line, the recognition accuracy deteriorates. This is caused by the speech segment replacement. From Figure 5.3, we can see the replacement percentage is approximately 50% and almost remains constant at different SNRs when the threshold is 5, which means 50% of the noisy segments is replaced with clean segments. However, as shown in Figure 4.6, the accuracy of finding correct clean segments is quite low, which is 42.4% even when SNR is 9 dB. When the threshold is 10 or 15, the replacement percentage is relatively low and the recognition accuracy is almost same as the baseline system. This shows that, using current audio fingerprinting system, more replacements lead to lower recognition accuracy. To make the proposed system perform better than the baseline system, we need to further improve the audio fingerprinting system to find the correct clean segment. More investigations about the relation between the accuracy of the audio fingerprinting system and the speech recognition rate of the whole system are done in next section.

5.5 Further Experiment

In previous section, the evaluation result shows the current audio fingerprinting system can not actually improve the speech recognition rate, due to its low accuracy in finding the correct speech segment for a noisy segment. However, if the accuracy of

the audio fingerprinting system is increased, we expect to achieve better performance than the baseline speech recognition system. To test this hypothesis, a synthetic experiment is carried out.

In this experiment, we control the clean speech segment used to replace the noisy one. In other words, we control the accuracy of the audio fingerprinting system. In addition, the replacement percentage, how many noisy segments will be replaced, is same as it is when the threshold is 5. By setting different accuracies for the audio fingerprinting system, we can derive the relation between the speech recognition rate and the accuracy of the audio fingerprinting system.

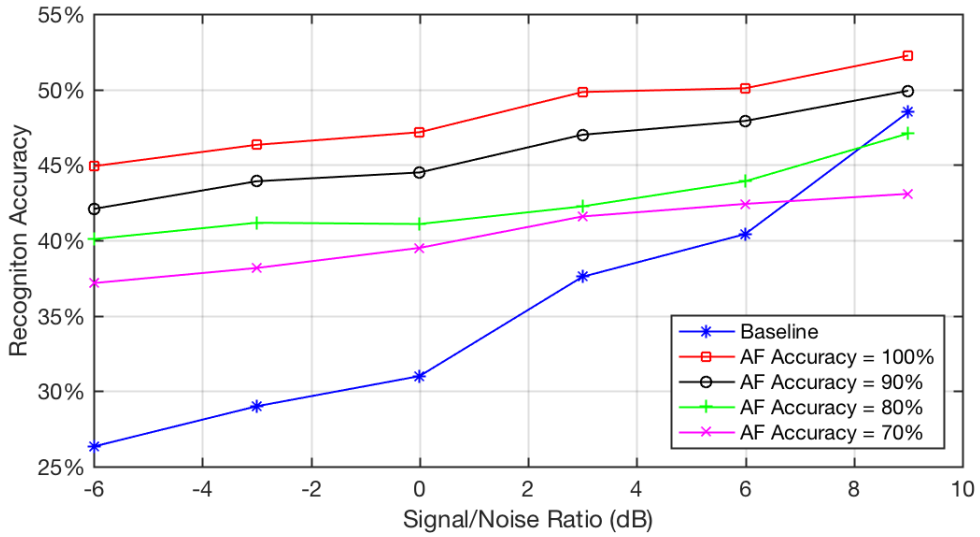

Figure 5.4: Synthetic experiment about speech recognition accuracy. AF Accuracy means the accuracy of the audio fingerprinting system in finding the correct speech segment for a noisy segment.

Experiment result is shown in Figure 5.4. Baseline recognition accuracy (blue line) is achieved when we test an alone speech recognition system as in Section 5.2. When audio fingerprinting is applied to the baseline system and we set 100% as its accuracy regardless of the SNR, we get the ceiling of the recognition accuracy (red line). Obviously, the performance is better than the baseline system, although the recognition accuracy is still much lower than 100%. There are two reasons. One is that the replacement percentage is approximately 50% when threshold is 5, which means only half of the noisy segments are replaced with clean ones. The other one is that even if a noisy segment is replace with a clean one, we cannot guarantee it will be recognized correctly due to the simple architecture of the speech recognition system.

After testing with another three audio fingerprinting accuracy (70%, 80% and 90%), we conclude that if the accuracy of an audio fingerprinting system is higher than 85% when the SNR is 9 dB and it is robust to noise, we can apply it to speech recognition and get better recognition accuracy.

5.6 Summary

Due to the low accuracy of the current audio fingerprinting system in finding correct clean speech segment, the proposed speech recognition system does not beat the baseline system in noisy environments. However, synthetic experiments show that the recognition accuracy will be improved if we can further improve the accuracy of the audio fingerprinting system. At some point, the proposed speech recognition system can achieve better performance than the baseline system. In Chapter 6, several possible ways are proposed to improve the audio fingerprinting system.

Chapter 6

Conclusions and Future Work

Among all the music information retrieval strategies based on audio content, audio fingerprinting has received most interest from both the academic and industrial areas. Many audio fingerprinting systems have been proposed, following different audio fingerprint computation and comparison algorithms, and they all fulfill some common requirements, including discrimination power, robustness to noise and distortions, compactness, computational simplicity, high search speed and good scalability. They have been widely used in broadcast monitoring, connected audio, filtering technology for file sharing and automatic music library organization [30], but it has never been used in speech recognition. In this thesis, we investigated the possibility and feasibility of applying audio fingerprinting to speech reconstruction, so as to improve speech recognition in noisy environments.

To evaluate the performance of one of the state-of-the-art audio fingerprinting algorithms, we build a landmark based audio fingerprinting system based on Ellis' work [22] and documented the detailed implementation in this work. The basic operation of this algorithm is to extract salient peaks from the spectrogram of the audio track and form them to pairs, which are also called landmarks. All the landmarks of the reference audio tracks are stored in a database implemented with a hash table. To identify a query track, we convert it to query landmarks and search the database for all the reference tracks that share similar landmarks pattern. The evaluation results show that this audio fingerprinting algorithm is robust against additive noise and various types of degradations. One drawback of this algorithm is that it is sensitive to speed changing of the query track, which leads to pitch shifting. 1% pitch shifting is the limitation of the system for reliable matching results.

Three strategies are proposed to improve the baseline audio fingerprinting algo-

rithm, in order to overcome the new challenges in the new application scenario of speech reconstruction . The first strategy is pre-emphasis, which can boost the energy in high frequencies, so as to make the peaks of the third and fourth formant of vowels more available for the following fingerprint extraction. The second strategy is robust landmark scheme to pitch shifting. In this scheme, CQT spectrogram is calculated instead of traditional FFT spectrogram and a new landmark representation is adopted. The last strategy is morphological peak extraction, a technique borrowed from image processing. The first two strategies improve the accuracy of clean speech segment searching from 35.6% to 46.4%, while the third strategy is not helpful.

A speech recognition platform is built to evaluate the whole system in which audio fingerprinting is integrated. As the accuracy of finding correct clean speech segment for a noisy speech segment is not high enough, the integration of audio fingerprinting does not actually improve the recognition rate of the speech recognition system.

In summary, we conclude that:

1. The baseline landmark-based audio fingerprinting scheme is robust against additive noise and various distortions except pitch shifting.

2. Robust landmark with CQT spectrogram and new landmark representation can improve the robustness of the audio fingerprinting system to pitch shifting.

3. Pre-emphasis and robust landmark scheme improve the accuracy of finding similar segment in the clean dataset for a noisy speech segment. Morphological peak extraction is not effective.

4. Further improvements are required for the accuracy of clean speech segment searching before it can be adopted in speech reconstruction and recognition in noisy environments.

This research work proposed a novel idea to do speech reconstruction and recognition in noisy environments and provided a preliminary investigation about its possibility and feasibility, but there are still much more works needed to be done:

1. Adaptive landmark density for query track. Although the landmark density can be set differently depending on the audio fingerprinting working mode, which is training mode or operating mode, the density is always fixed for all query tracks. One possible way to improve the recognition rate is to adapt the density based on the estimation of SNR of the query track. The density can be set in

inverse relation to the SNR. Higher hash density is configured for query track whose SNR is lower. This is a way of trading computational complexity and search speed for accuracy.

2. Landmark with strength. Rather than considering only the locations of peaks, we can incorporate the landmark with the strength of peaks. In this way, we can put more importance on more strong peaks, which are more likely to survive in noisy environments.

3. Larger amount of hashes. Instead of forming landmarks by combining two peaks, generate more informative hashes with higher entropy by combining three or more peaks. In this way, there will be less items in each bucket of the hash table. This can help to accelerate the search procedure in the database.

4. A database management system to store the landmarks of reference tracks. This can help to avoid collision and replacement when there are too many items in some buckets. In addition, to increase memory access efficiency, we can cache landmarks of frequently queried reference tracks and put the others in memory.

5. More robust regression techniques to decide whether a match has been found. Currently we need to detect a diagonal line within the scatterplot of time offsets. This is too rigid for audio fingerprinting when it is used in speech reconstruction. Support vector machine (SVM) is a potential solution for this problem.

6. Dynamic speech segment length. When we divide the noisy speech into short segments, the length can be dynamic and depend on SNR, vocabulary size, clean training dataset size, etc.

7. Other audio fingerprinting schemes. There are many more audio fingerprinting schemes that have been proposed by researchers. For instance, compared to the landmark based scheme, Ke's scheme [39] provides better recognition rate when the query track is short [15] and Baluja's scheme [5] provides more discriminative fingerprints. These schemes are good candidates to improve the accuracy of finding correct similar segment in the clean dataset.

Hopefully this thesis can open up a new application scenario for audio fingerprinting and provide directions for future research work.

Bibliography

[1] MA Abd El-Fattah, Moawad Ibrahim Dessouky, Salah M Diab, and Fathi El-Sayed Abd El-Samie. Speech enhancement using an adaptive wiener filtering approach. *Progress In Electromagnetics Research M*, 4:167–184, 2008.

[2] Nasir Ahmed, T_ Natarajan, and Kamisetty R Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

[3] Eric Allamanche, Jürgen Herre, Oliver Hellmuth, Bernhard Fröba, Thrsten Kastner, and Markus Cremer. Content-based identification of audio material using mpeg-7 low level description. In *ISMIR*, 2001.

[4] Andreas Antoniou. *Digital signal processing*. McGraw-Hill Toronto, Canada:, 2006.

[5] Shumeet Baluja and Michele Covell. Content fingerprinting using wavelets. *Proc. of European Conference on Visual Media Production (CVMP)*, 2006.

[6] Shumeet Baluja and Michele Covell. Audio fingerprinting: Combining computer vision & data stream processing. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II–213. IEEE, 2007.

[7] Michael Berouti, Richard Schwartz, and John Makhoul. Enhancement of speech corrupted by acoustic noise. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'79.*, volume 4, pages 208–211. IEEE, 1979.

[8] Thomas L Blum, Douglas F Keislar, James A Wheaton, and Erling H Wold. Method and article of manufacture for content-based analysis, storage, retrieval, and segmentation of audio information, June 29 1999. US Patent 5,918,223.

[9] Steven Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on acoustics, speech, and signal processing*, 27(2):113–120, 1979.

[10] Judith C Brown. Calculation of a constant q spectral transform. *The Journal of the Acoustical Society of America*, 89(1):425–434, 1991.

[11] Judith C Brown and Miller S Puckette. An efficient algorithm for the calculation of a constant q transform. *The Journal of the Acoustical Society of America*, 92(5):2698–2701, 1992.

[12] Pedro Cano, E Batle, Ton Kalker, and Jaap Haitsma. A review of algorithms for audio fingerprinting. In *Multimedia Signal Processing, 2002 IEEE Workshop on*, pages 169–173. IEEE, 2002.

[13] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma. A review of audio fingerprinting. *Journal of VLSI signal processing systems for signal, image and video technology*, 41(3):271–284, 2005.

[14] Pedro Cano, Eloi Batlle, Harald Mayer, and Helmut Neuschmied. Robust sound modeling for song detection in broadcast audio. *Proc. AES 112th Int. Conv*, pages 1–7, 2002.

[15] Vijay Chandrasekhar, Matt Sharifi, and David A Ross. Survey and evaluation of audio fingerprinting schemes for mobile query-by-example applications. In *ISMIR*, volume 20, pages 801–806, 2011.

[16] Jianping Chen and Tiejun Huang. A robust feature extraction algorithm for audio fingerprinting. In *Pacific-Rim Conference on Multimedia*, pages 887–890. Springer, 2008.

[17] CHiME. Small vocabulary track. `http://spandh.dcs.shef.ac.uk/chime_challenge/chime2013/chime2_task1.html`, 2013. [Online; accessed 31-January-2017].

[18] Heidi Christensen, Jon Barker, Ning Ma, and Phil D Green. The chime corpus: A resource and a challenge for computational hearing in multisource environments. In *Interspeech*, pages 1918–1921. Citeseer, 2010.

[19] Martin Cooke, Jon Barker, Stuart Cunningham, and Xu Shao. An audio-visual corpus for speech perception and automatic speech recognition. *The Journal of the Acoustical Society of America*, 120(5):2421–2424, 2006.

[20] D.Ellis. Robust landmark-based audio fingerprinting. `http://www.ee.columbia.edu/ln/rosa/matlab/fingerprint/`, 2009.

[21] Li Deng and Douglas O'Shaughnessy. *Speech processing: a dynamic and optimization-oriented approach*. CRC Press, 2003.

[22] Dan Ellis. Robust landmark-based audio fingerprinting. *web resource, available: http://labrosa. ee. columbia. edu/matlab/fingerprint*, 2009.

[23] Yariv Ephraim, Hanoch Lev-Ari, and William JJ Roberts. A brief survey of speech enhancement. *The Electronic Handbook*, 2, 2005.

[24] Sébastien Fenet, Gaël Richard, Yves Grenier, et al. A scalable audio fingerprint method with robustness to pitch-shifting. In *ISMIR*, pages 121–126, 2011.

[25] Mark John Francis Gales. *Model-based techniques for noise robust speech recognition*. PhD thesis, University of Cambridge Cambridge, 1995.

[26] D Giannoulis, E Benetos, D Stowell, and MD Plumbley. Ieee aasp challenge on detection and classification of acoustic scenes and events-public dataset for scene classification task. *Queen Mary University of London*, 2012.

[27] Yifan Gong. Speech recognition in noisy environments: A survey. *Speech communication*, 16(3):261–291, 1995.

[28] Ulf Grenander. Probability and statistics the harald cramer volume, 1959.

[29] Peter Grosche, Meinard Müller, and Joan Serrà. Audio content-based music retrieval. In *Dagstuhl Follow-Ups*, volume 3. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2012.

[30] Jaap Haitsma and Ton Kalker. A highly robust audio fingerprinting system. In *Ismir*, volume 2002, pages 107–115, 2002.

[31] Jaap Haitsma, Ton Kalker, and Job Oostveen. Robust audio hashing for content identification. In *International Workshop on Content-Based Multimedia Indexing*, volume 4, pages 117–124. Citeseer, 2001.

[32] Chuang He and George Zweig. Adaptive two-band spectral subtraction with multi-window spectral estimation. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 2, pages 793–796. IEEE, 1999.

[33] Gerhard Heinzel, Albrecht Rüdiger, and Roland Schilling. Spectrum and spectral density estimation by the discrete fourier transform (dft), including a comprehensive list of window functions and some new at-top windows, 2002.

[34] Xuedong Huang, Alex Acero, Hsiao-Wuen Hon, and Raj Foreword By-Reddy. *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice hall PTR, 2001.

[35] Dalwon Jang, Chang Dong Yoo, Sunil Lee, Sungwoong Kim, and Ton Kalker. Pairwise boosted audio fingerprint. *IEEE transactions on information forensics and security*, 4(4):995, 2009.

[36] Daniel Jurafsky and H James. *Speech and language processing an introduction to natural language processing, computational linguistics, and speech*. Pearson Education, 2000.

[37] Sunil Kamath and Philipos Loizou. A multi-band spectral subtraction method for enhancing speech corrupted by colored noise. In *ICASSP*, volume 4, pages 44164–44164. Citeseer, 2002.

[38] Thorsten Kastner, Eric Allamanche, Jurgen Herre, Oliver Hellmuth, Markus Cremer, and Holger Grossmann. Mpeg-7 scalable robust audio fingerprinting. In *Audio Engineering Society Convention 112*. Audio Engineering Society, 2002.

[39] Yan Ke, Derek Hoiem, and Rahul Sukthankar. Computer vision for music identification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 597–604. IEEE, 2005.

[40] Rake& Agrawal King-lp Lin and Harpreet S Sawhney Kyuseok Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Proceeding of the 21th International Conference on Very Large Data Bases*, pages 490–501. Citeseer, 1995.

[41] Philip Lockwood and Jérôme Boudy. Experiments with a nonlinear spectral subtractor (nss), hidden markov models and the projection, for robust speech recognition in cars. *Speech communication*, 11(2-3):215–228, 1992.

[42] Matthias Mauch and Sebastian Ewert. The audio degradation toolbox and its application to robustness evaluation. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR 2013)*, Curitiba, Brazil, 2013. accepted.

[43] M Kivanç Mıhçak and Ramarathnam Venkatesan. A perceptual audio hashing algorithm: a tool for robust audio identification and information hiding. In *International Workshop on Information Hiding*, pages 51–65. Springer, 2001.

[44] Meinard Müller and Joan Serrà. Tutorial: Audio content-based music retrieval. `http://www.iiia.csic.es/~jserra/downloads/2011_MuellerSerra_MusicRetrieval_Tutorial-ISMIR_handouts-6.pdf`. [Online; accessed 8-Feb-2017].

[45] S Ogata and Tetsuya Shimamura. Reinforced spectral subtraction method to enhance speech signal. In *TENCON 2001. Proceedings of IEEE Region 10 International Conference on Electrical and Electronic Technology*, volume 1, pages 242–245. IEEE, 2001.

[46] Constantin Papaodysseus, George Roussopoulos, Dimitrios Fragoulis, Athanasios Panagopoulos, and Constantin Alexiou. A new approach to the automatic recognition of musical recordings. *Journal of the Audio Engineering Society*, 49(1/2):23–35, 2001.

[47] Gordon E Peterson and Harold L Barney. Control methods used in a study of the vowels. *The Journal of the acoustical society of America*, 24(2):175–184, 1952.

[48] Joseph W Picone. Signal modeling techniques in speech recognition. *Proceedings of the IEEE*, 81(9):1215–1247, 1993.

[49] Lawrence R Rabiner and Ronald W Schafer. *Digital processing of speech signals*. Prentice Hall, 1978.

[50] Cort Lippe Richard Dudas. The phase vocoder part i. https://cycling74.com/2006/11/02/the-phase-vocoder, 2006. [Online; accessed 28-January-2017].

[51] Gábor Richly, Lbzlo Varga, F Kovács, and G Hosszú. Short-term sound stream characterization for reliable, real-time occurrence monitoring of given sound-prints. In *Electrotechnical Conference, 2000. MELECON 2000. 10th Mediterranean*, volume 2, pages 526–528. IEEE, 2000.

[52] Aaron Schwartzbard. Songprint. `http://www.freecode.com/projects/songprint`, 2000. [Online; accessed 30-January-2017].

[53] Shazam music recognition service. `http://www.shazam.com/`.

[54] Yann Soon, Soo Nge Koh, and Chai Kiat Yeo. Selective magnitude subtraction for speech enhancement. In *High Performance Computing in the Asia-Pacific Region, 2000. Proceedings. The Fourth International Conference/Exhibition on*, volume 2, pages 692–695. IEEE, 2000.

[55] Soundhound. `http://www.soundhound.com/`.

[56] Structuring elements. `https://www.mathworks.com/help/images/structuring-elements.html`. [Online; accessed 6-Feb-2017].

[57] Bob L Sturm. The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use. *arXiv preprint arXiv:1306.1461*, 2013.

[58] SR Subramanya, Rahul Simha, Bhagirath Narahari, and Abdou Youssef. Transform-based indexing of audio data for multimedia databases. In *Multimedia Computing and Systems' 97. Proceedings., IEEE International Conference on*, pages 211–218. IEEE, 1997.

[59] S Theodoridis and K Koutroumbas. Pattern recognition, academic press. *New York*, 1999.

[60] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.

[61] Nathalie Virag. Single channel speech enhancement based on masking properties of the human auditory system. *IEEE Transactions on speech and audio processing*, 7(2):126–137, 1999.

[62] Avery Wang et al. An industrial strength audio search algorithm. In *ISMIR*, volume 2003, pages 7–13. Washington, DC, 2003.

[63] Wikipedia. Constant-q transform — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Constant-Q_transform`, 2017. [Online; accessed 28-January-2017].

[64] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Achieving human parity in conversational speech recognition. *arXiv preprint arXiv:1610.05256*, 2016.