A Comparison-based Approach to Mispronunciation Detection

by

Ann Lee

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2012

Author . . . .
Department of Electrical Engineering and Computer Science
May 23, 2012

Certified by . . . .
James Glass
Senior Research Scientist
Thesis Supervisor

Accepted by . . . .
Leslie A. Kolodziejski
Chair of the Committee on Graduate Students

A Comparison-based Approach to Mispronunciation Detection

by

Ann Lee

Submitted to the Department of Electrical Engineering and Computer Science
on May 23, 2012, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

This thesis focuses on the problem of detecting word-level mispronunciations in nonnative speech. Conventional automatic speech recognition-based mispronunciation detection systems have the disadvantage of requiring a large amount of language-specific, annotated training data. Some systems even require a speech recognizer in the target language and another one in the students' native language. To reduce human labeling effort and for generalization across all languages, we propose a comparison-based framework which only requires word-level timing information from the native training data. With the assumption that the student is trying to enunciate the given script, dynamic time warping (DTW) is carried out between a student's utterance (nonnative speech) and a teacher's utterance (native speech), and we focus on detecting mis-alignment in the warping path and the distance matrix.

The first stage of the system locates word boundaries in the nonnative utterance. To handle the problem that nonnative speech often contains intra-word pauses, we run DTW with a silence model which can align the two utterances, detect and remove silences at the same time.

In order to segment each word into smaller, acoustically similar, units for a finer-grained analysis, we develop a phoneme-like unit segmentor which works by segmenting the self-similarity matrix into low-distance regions along the diagonal. Both phone-level and word-level features that describe the degree of mis-alignment between the two utterances are extracted, and the problem is formulated as a classification task. SVM classifiers are trained, and three voting schemes are considered for the cases where there are more than one matching reference utterance.

The system is evaluated on the Chinese University Chinese Learners of English (CU-CHLOE) corpus, and the TIMIT corpus is used as the native corpus. Experimental results have shown 1) the effectiveness of the silence model in guiding DTW to capture the word boundaries in nonnative speech more accurately, 2) the complimentary performance of the word-level and the phone-level features, and 3) the stable performance of the system with or without phonetic units labeling.

Thesis Supervisor: James Glass
Title: Senior Research Scientist

Acknowledgments

First of all, I would like to thank my advisor, Jim Glass, for the generous encouragements and inspiring research ideas he gave to me throughout this thesis work. His patience has guided me through the process of choosing a research topic. Without his insightful suggestions, it wouldn't be possible for me to finish this thesis. I would also like to thank Wade Shen for his valuable advice during our discussions.

Thanks Prof. Helen Meng from Chinese University of Hong Kong for providing the nonnative corpus. Thanks Mitch Peabody for his previous efforts on annotation collection, and Yaodong Zhang for his work which inspired the design of the core of the system. Also, thanks Carrie for spending her time proofreading this thesis. Many thanks to Hung-An, Yaodong, Jackie, Ekapol and Yushi for discussing with me and for all the great suggestions they gave. I would also like to thank all SLS members for creating such a wonderful working environment, especially my lovely former and present officemates.

It has been my second year living abroad now. Thanks Annie Chen for being my roommate when I first came here. My first experience of living on my own in a foreign country could be a mess without her help. Thanks all my dear friends here, Hsin-Jung, Yu-Han, Pan, Yin-Wen, and Paul. They've provided so much joy in my life. I'm also grateful to Chia-Wei and Chia-Ying in Taiwan, who have always been there online when I need someone to talk with, even though we are in different time zones.

Finally, special thanks to my parents, whose love and support have always been there encouraging me. Skyping with them has always been the best way for me to relieve stress. Thanks Chung-Heng Yeh for always being there sharing every tear and joy with me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of the Problem

Computer-Aided Language Learning (CALL) systems have gained popularity due to the flexibility they provide to empower students to learn at their own pace. Instead of physically sitting in a classroom and following pre-defined schedules, with the help of CALL systems, students can study the language they are interested in on their own, as long as they have access to a computer.

Pronunciation, vocabulary, and grammar are the three key factors to mastering a foreign language. Therefore, a good CALL system should be able to help students develop their capability in these three aspects. This thesis focuses on one specific area – Computer-Aided Pronunciation Training (CAPT), which is about detecting mispronounced words, especially in nonnative speech, and giving feedback. Automatic speech recognition (ASR) is the most intuitive solution when people first looked into the problem of building CALL systems, and recognizer-based CAPT has long been a research topic. However, when doing so, the limits of current ASR technology have also become a problem.

In this thesis, a comparison-based mispronunciation detection framework is proposed. Within our framework, a student's utterance is directly compared with a teacher's instead of going through a recognizer. The rest of this chapter describes the motivation and the assumptions of the proposed system, presents an overview of the whole system framework, and outlines the content of the remaining chapters.

1.2 Overview of the System

1.2.1 Motivation

In order to automatically evaluate a student's pronunciation, the student's speech has to be compared with some reference models, which can be one or more native speakers of the target language. As will be discussed in detail in Chapter 2, the conventional approach of detecting mispronunciation relies on good automatic speech recognition (ASR) techniques. However, a recognizer-based approach has several disadvantages. First, it requires a large amount of training data that is carefully transcribed at the phone-level in the target language, which is both time-consuming and expensive. Also, since native and nonnative speech differ in many aspects, a CAPT system usually also requires an acoustic model trained on nonnative speech, resulting in the need of well-labeled nonnative training data.

Besides the fact that preparing training data requires extensive human labeling efforts, these two disadvantages also lead to recognizer-based approaches being language-dependent. A new recognizer has to be built every time we want to build a CALL system for a different target language. What is worse is that the nonnative acoustic model has to take different native languages of the students into account. There are 6,909 languages according to the currently most extensive catalog of the world's languages [1]. However, commercially-available recognizers only feature around 50 languages that are frequently used in the world [2]. Therefore, there is definitely a need for investigating a non-recognizer-based, language-independent approach to building a CAPT system.

As a result, instead of using a recognizer, we turn to a comparison-based approach. The intuition is that if a student's utterance is very "close" to a teacher's in some sense, then he/she should be performing well. Inspired by the unsupervised pattern matching techniques described in the next chapter, we carry out dynamic time warping (DTW) to align two utterances and see if the student's looks similar to the teacher's. By locating poorly matching regions, we can thus detect mispronunciations. Our approach does not require any linguistic knowledge of the target language and the student's native language, and thus it should be language-independent. To avoid the problem of requiring too much human labeling effort, we also propose an unsupervised phone segmentor to segment each word into smaller units for a finer-grained analysis.

1.2.2 Assumptions

To envision the implementation of our framework into a CALL system, imagine a script-based system, which can be a reading game or a guided dialogue. Given a text sentence displayed on the screen, the first assumption would be that the student is trying his/her best to pronounced what he/she saw on the screen. If the student does not know how to read the given sentence, there would be a play button that could provide a reading example from a "teacher". If the student does not know how to read a certain word, he/she could also choose to play only the audio of that word. These functions are based on our second assumption – for every script in our teaching material, there is at least one recording from a native speaker of the target language, and we have word-level timing labels on the recording.

We believe these assumptions are fairly reasonable, since nowadays we can access a huge amount of audio data on the Internet easily. The content varies from news and public speeches to TV series, and all of them provide us with more than enough audio materials for language learning. Annotating word boundaries is a fairly simple task that every native speaker of the target language can do, unlike phone-level annotations, which are typically done by experts.

1.2.3 The Proposed Framework

Fig. 1-1 shows the flowchart of our system. Our system detects mispronunciation on a word level, so the first stage is to locate word boundaries in the student's utterance. Dynamic time warping (DTW) is a technique that is often used when aligning two sequences. Here we propose to incorporate a silence model when running DTW. In this way, we can align the two utterances, detect and remove silence in the student's utterance at the same time. The aligned path, together with the word boundaries of the teacher's utterance, can help us locate word boundaries in the student's utterance.

After segmenting the student's utterance into words, the second stage is to determine whether each word is mispronounced or not. Here we first propose an unsupervised phoneme-like unit segmentor that can further segment a word into smaller units. Some specially de-

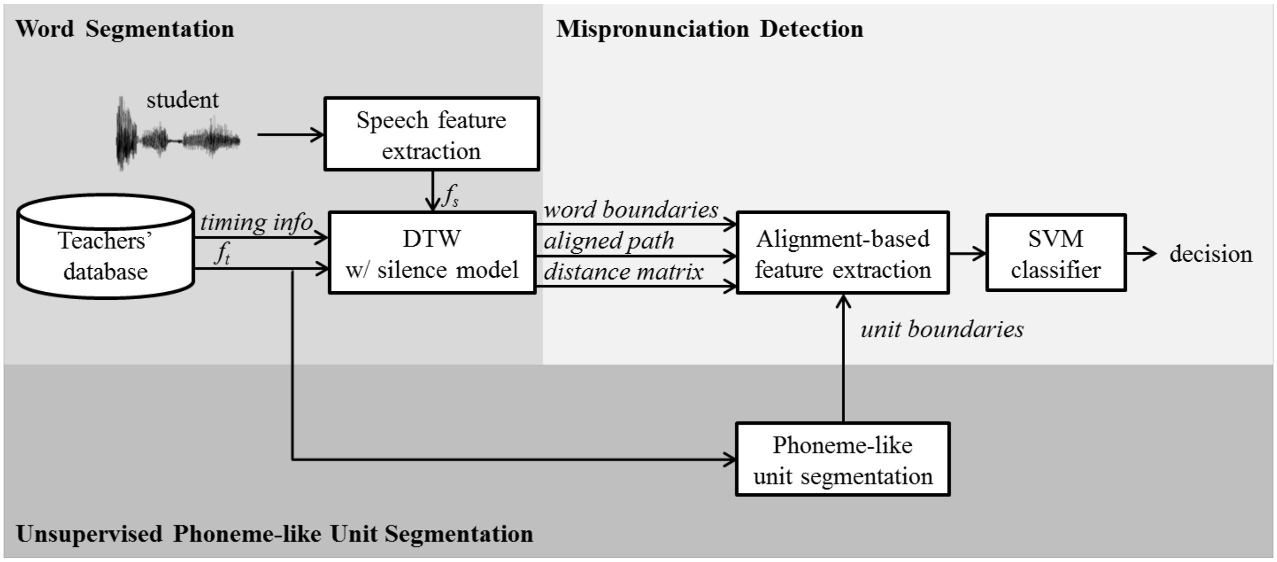


Figure 1-1: *System overview (with single reference speaker)*

signed features that describe the shape of the aligned path and the appearance of the distance matrix are extracted, either within a word or within those phone-like units. Given the features, together with the word-level labels, we form the problem of detecting mispronunciation as a classification problem. Support vector machine (SVM) classifiers are trained and used for prediction. If there are more than one reference speaker, i. e. more than one teacher's recording for a script, we further examine three voting schemes to combine the decisions from separate alignments.

1.2.4 Contributions

The main contribution of this thesis is the comparison-based mispronunciation detection framework that requires only word-level timing information on the native training data. We have approached this problem through three steps. First of all, we present how to accurately locate word boundaries in a nonnative utterance through an alignment with a native utterance. We extend basic DTW to incorporate a silence model, which allows the DTW process to be more flexible in choosing the warping targets.

Second, we propose an unsupervised phoneme-like unit segmentor that can divide an utterance into acoustically similar units. With these units, the system can work without phonetic unit human labeling.

Last but not least, we introduce a set of word-level and phone-level features and demonstrate their ability in detecting mispronunciations. These features are extracted based on the shape of the aligned path and the structure of the distance matrix. We perform a series of experiments to show that combining features at both levels can achieve the best performance.

1.3 Thesis Outline

The rest of this thesis is organized as follows:

Chapter 2 presents background and related work, including recognizer-based approaches to mispronunciation detection and pattern matching techniques. The native and nonnative corpora we use for experiments are also introduced.

Chapter 3 illustrates the first stage of our system – word segmentation. Experimental results showing how accurately DTW can capture the word boundaries in nonnative speech are reported and discussed.

Chapter 4 introduces a phoneme-like unit segmentor, which is used to segment each word into smaller acoustically similar units for more detailed analysis. The evaluation is done on the task of phonetic boundary detection.

Chapter 5 explains how we extract features for mispronunciation detection in detail. Experimental results showing the performance of the framework with respect to different amount of information are presented and discussed.

Chapter 6 concludes the thesis with a summary of the contributions and suggests possible future research directions.

Chapter 2

Background and Related Work

2.1 Pronunciation and Language Learning

Non-native speakers, especially adults, are easily affected by their native language (L1) when learning a target language (L2). Pronunciation involves many different dimensions. It is known that the error patterns a language learner makes are correlated with his/her level of competency [5, 12, 13]. As a student embarks on learning a new language, the most common errors are at the phonetic level, such as a substitution, insertion, or deletion of one or more phones. These errors are due to the unfamiliarity with phoneme inventories and the phonological rules of L2. The student might substitute an unfamiliar phoneme with a similar one that exists in L1. A famous example would be Japanese learners of English of beginning levels substituting /*s*/ for /*th*/, and /*l*/ for /*r*/ [13, 31]. Also, due to the lack of vocabulary, when seeing a word for the first time, the student might not know the correct rules to pronounce it. For example, a vowel in English has different pronunciations depending on its context.

As the student becomes more proficient, these kinds of errors may happen less frequently, and instead, prosody becomes an important issue. Lexical stress, tone, and time duration are some categories on the prosodic level. Previous work has shown that prosody has more impact on learners' intelligibility then the phonetic features do [3, 33]. However, the prosodic aspects of a language are sometimes hidden in details. Learning these details involves correctly perceiving the target language. Nonetheless, a student's L1 may limit his/her ability to become aware of certain prosodic features in L2. For example, for many language learners who have non-tonal native languages, it is difficult to distinguish the tones in tonal languages such as Mandarin Chinese or Cantonese even when perceiving, not to mention producing those tones [28, 34].

When designing a CALL system, people are more concerned about the *precision* of the system, rather than the *recall*, as it would discourage the student from learning if the system detects many errors that are in fact good pronunciations [5, 13]. In addition to correctly detecting the above errors, the feedback provided by the system is also critical. Multimodal feedback is popular, such as messages in text, audio playback, and animation of the lips or vocal tract, as they can improve the comprehensibility of the system and thus the learning efficiency.

2.2 Computer-Assisted Pronunciation Training (CAPT)

CAPT systems are a kind of CALL system that are specially designed for the purpose of pronunciation training. There are two types of evaluation: individual error detection and pronunciation assessment. The former is about detecting word or subword level pronunciation errors and providing feedback, while the latter is about scoring the overall fluency of a student. As the focus of this thesis is on word-level mispronunciation detection, here we only present previous work in this area.

2.2.1 Individual Error Detection

ASR technology can be applied to CAPT in many different ways. Kewley-Port et. al [21] used a speaker-independent, isolated-word, template-based recognizer to build a speech training system for children. The spectrum of a child's input speech is coded into a series of 16-bit binary vectors, and is compared to the stored templates by computing the percentage of matching bits relative to the total number of bits of each template. That number is used as the score for indicating how good the articulation is. Wohlert [37] also adopted a template-based speech recognizer for building a CAPT system for learning German. Dalby and Kewley-Port [10] further compared two types of recognizers: a template-based system which performs pattern matching and a HMM-based system which is based on nondeterministic stochastic modeling. Their conclusion is that an HMM-based recognizer performs better in terms of accuracy in identifying words, while a template-based recognizer works better in distinguishing between minimal pairs.

Witt and Young [36] have proposed a goodness of pronunciation (GOP) score, which can be interpreted as the duration normalized log of the posterior probability of a speaker saying a phone given acoustic observations. Phone dependent thresholds are set to judge whether each phone from the forced alignment is mispronounced or not. The 10 subjects for their experiment spoke a variety of L1s, which were Latin-American Spanish, Italian, Japanese and Korean. Their experimental results show that pronunciation scoring based on likelihood scores from a recognizer can achieve satisfactory performance. Also based on posterior probabilities, Franco et. al [15] trained three recognizers by using data of different levels of nativeness and used the ratio of the log-posterior probability-based scores from each recognizer to detect mispronunciation. In recent work, Peabody [28] proposed to anchor the vowel space of nonnative speakers according to that of native speakers before computing the likelihood scores.

Some approaches incorporated the knowledge of the students' L1 into consideration. They focused on predicting a set of possible errors and enhancing the recognizer to be able to recognize the wrong versions of pronunciations. These error patterns can be either hand-coded from linguistic knowledge or learned in a data-driven manner.

Meng et. al [24] proposed an extended pronunciation lexicon that incorporates possible phonetic confusions based on the theory of language transfer. Possible phonetic confusions were predicted by systematically comparing phonology between the two languages. They performed a thorough analysis of salient pronunciation error patterns that Cantonese speakers would have when learning English. Kim et. al [22] also carefully constructed phonological rules that account for the influence of Korean (L1) on English (L2). Harrison et. al [18] considered context-sensitive phonological rules rather than context-insensitive rules. Qian et. al [29] adopted the extended pronunciation lexicon and proposed a discriminatively-trained acoustic model that jointly minimizes mispronunciation and diagnosis errors to enhance the recognizer.

Most recently, Wang and Lee [35] further integrate GOP scores with error pattern detectors. Their experimental results show that the integrated framework performs better than using only one of them in detecting mispronunciation within a group of students from 36 different countries learning Mandarin Chinese.

2.3 Posteriorgram-based Pattern Matching in Speech

Dynamic time warping (DTW) is an algorithm that finds an optimal match between two sequences which may vary in time or speed. It has played an important role in early template-based speech recognition. The book written by Rabiner and Juang [31] has a complete discussion about issues such as what the distortion measures should be, and how to set time-normalization constraints. In early work, the distortion measure might be based on filter bank output [32] or LPC features [27]. More recently, posterior features have been applied to speech recognition, and have also been successfully applied to facilitate unsupervised spoken keyword detection. Below we introduce the definition of the posteriorgram, and present some previous work on posteriorgram-based pattern matching applications.

2.3.1 Posteriorgram Definition

Phonetic Posteriorgram

A posteriorgram is a vector of posterior probabilities over some predefined classes, and can be viewed as a compact representation of speech. Given a speech frame, its phonetic posteriorgram should be a $N \times 1$ vector, where N equals the number of phonetic classes, and each element of this vector is the posterior probability of the corresponding phonetic class given that speech frame. Posteriorgrams can be computed directly from likelihood scores for each phonetic class, or by running a phonetic recognizer to generate a lattice for decoding [19].

Gaussian Posteriorgram

In contrast to the supervised phonetic posteriorgram, a Gaussian posteriorgram (GP) can be computed from a set of Gaussian mixtures that are trained in an unsupervised manner [38].

For an utterance $S = (f_{s_1}, f_{s_2}, ..., f_{s_n})$, where n is the number of frames, the GP for the ith frame is defined as

$$gp_{f_{s_i}} = [P(C_1|f_{s_i}), P(C_2|f_{s_i}), ..., P(C_g|f_{s_i})], \tag{2.1}$$

where C_j is a component from a g-component Gaussian mixture model (GMM) which can be trained from a set of unlabeled speech. In other words, $gp_{f_{s_i}}$ is a g-dimensional vector of posterior probabilities.

2.3.2 Application in Speech

Application in Speech Recognition

Aradilla et. al [4] re-investigated the problem of template-based speech recognition by using posterior-based features as the templates and showed significant improvement over the standard template-based recognizers on a continuous digit recognition task. They took advantage of a multi-layer perceptron to estimate the phone posterior probabilities based on spectral-based features and used KL-divergence as the distance metric.

Application in Keyword Spotting and Spoken Term Discovery

Spoken keyword detection for spoken audio data aims at searching the audio data for a given keyword in audio without the need of a speech recognizer, and spoken term discovery is the task of finding repetitive patterns, which can be a word, a phrase, or a sentence, in audio data. The initial attempt by Hazen et al. [19] was a phonetic-posteriorgram-based, spoken keyword detection system. Given a keyword, it can be decoded into a set of phonetic posteriorgrams by phonetic recognizers. Dynamic time warping was carried out between the posteriorgram representation of the keyword and the posteriorgram of the stored utterances, and the alignment score was used to rank the detection results.

To train a phonetic recognizer, one must have a set of training data with phone-level labels. Zhang et. al [38] explored an unsupervised framework for spoken query detection by decoding the posteriorgrams from a GMM which can be trained on a set of unlabeled speech. Their following work [39] has shown that posteriorgrams decoded from Deep Boltzmann Machines can further improve the system performance. Besides the aligned scores, Muscariello et. al [26, 25] also investigated some image processing techniques to compare the self-similarity matrices (SSMs) of two words. By combining the DTW-based scores with the SSM-based scores, the performance on spoken term detection can be improved.

2.4 Corpora

In this thesis, we are proposing a comparison-based framework, and thus, we require a dataset in the target language (English) and a dataset in a nonnative language, which is Cantonese in our case. The two datasets must have utterances of the same content for us to carry out alignment, and this is also the reason why we choose the TIMIT corpus as the English dataset (target language), and the Chinese University Chinese Learners of English (CU-CHLOE) corpus [24] as the nonnative dataset, which contains a set of scripts that are the same as those in TIMIT. The following sub-sections describe the two corpora.

2.4.1 TIMIT

The TIMIT corpus [16] is a joint effort between Massachusetts Institute of Technology (MIT), Stanford Research Institute (SRI), Texas Instruments (TI), and the National Institute of Standards and Technology (NIST). It consists of continuous read speech from 630 speakers, 438 males and 192 females from 8 major dialect regions of American English, with each speaking 10 phonetically-rich sentences. All utterances are phonetically transcribed and the transcribed units are time-aligned.

There are three sets of prompts in TIMIT:

1. *dialect (SA)*:

   This set contains two sentences which are designed to expose the dialectal variants of the speakers, and thus were read by all 630 speakers.

2. *phonetically-compact (SX)*:

   This set consists of 450 sentences that are specially designed to have a good coverage of pairs of phonetic units. Each speaker read 5 of these sentences, and each prompt was spoken by 7 speakers.

3. *phonetically-diverse (SI)*:

   The 1,890 sentences in this set were selected from existing text sources with the goal of maximizing the variety of the context. Each speaker contributed 3 utterances to this set, and each sentence was only read by one speaker.

2.4.2 CU-CHLOE

The Chinese University Chinese Learners of English (CU-CHLOE) corpus is a specially-designed corpus of Cantonese speaking English collected at the Chinese University of Hong Kong (CUHK). There are 100 speakers (50 males and 50 females) in total, who are all university students. The speakers were selected based on the criteria that their native language is Cantonese, they have learned English for at least 10 years, and their English pronunciation is viewed as intermediate to good by professional teachers.

There are also three sets of prompts in the CU-CHLOE corpus:

1. *"The North Wind and the Sun"*:

   The first set is a story from the Aesop's Fable that is often used to exemplify languages in linguistic research. The passage was divided into 6 sentences after recording. Every speaker read this passage.

2. *specially designed materials*:

   There are three sets of materials designed by the English teachers in CUHK.

   - Phonemic Sounds (ps): There are 20 sentences in this subset.
   - Confusable Words (cw): There are 10 groups of confusable words. For example, *debt doubt dubious*, or *saga sage sagacious sagacity.*
   - Minimal Pairs (mp): There are 50 scripts including 128 pairs of words, such as *look luke*, *cart caught*, or *sew sore.*

   Each speaker recorded all the prompts in these three subsets.

3. *sentences from the TIMIT corpus*:

   All sentences from the *SA*, *SX* and the *SI* set in TIMIT are included.

All recordings were collected using close-talking microphones and were sampled at 16kHz. Overall, each speaker contributed 367 utterances to this corpus, so there are 36,700 recordings in total, consisting of 306,752 words. Of these utterances, 5,597 (36,874 words from all sets of prompts except the TIMIT recordings) were phonetically hand transcribed.

Previous work on this corpus focused on recognizer-based methods, such as building a lexicon that incorporates possible pronunciations of words and compares whether the recognizer's output is the same as the correct pronunciation [24], or anchoring the vowel space of nonnative speakers according to that of native speakers and adopting likelihood scores to train a classifier [28].

2.4.3 Datasets for Experiments

In order to carry out alignment, we use the part of the CU-CHOLE corpus that is based on TIMIT prompts for experiments. We divide the 50 male and 50 female speakers into 25 males and 25 females for training, and the rest for testing, so there is no speaker overlap between the training set and the test set.

In previous work by Peabody [28], annotation on word-level pronunciation correctness were collected through Amazon Mechanical Turk (AMT). Each turker (the person who performs a task on the AMT platform) was asked to listen to an audio recording and click on a word if it was mispronounced. There were three turkers labeling each utterance. Only the words whose labels got agreement among all three turkers are used, and only the utterances which have at least one mispronounced word are included in the dataset. Details about the labeling process can be found in [28].

In order to further decrease word overlap between the training and the test set, we chose the prompts in the *SI* set for training, and *SX* for testing. The native utterances come from the TIMIT corpus, but only reference speakers of the same sex as the student are used for alignment. In the end, there is only one matching reference utterance for each student's utterance in the training set, compared to 3.8 reference utterances on average in the test set. The details about the dataset are shown in Table 2.1.

	# utterances (# unique)	# words	# mispronounced words (ratio)
training	1196 (827)	9989	1523 (15.25%)
testing	1065 (348)	6894	1406 (20.39%)

Table 2.1: *Dataset*

2.5 Summary

In this chapter, we have introduced 1) the background on different types of pronunciation errors, 2) previous work on CAPT system, especially focusing on individual error detection, and on posteriorgram-based pattern matching approaches in speech, and 3) the two corpora that are used in this thesis. The posteriorgram-based pattern matching approaches motivated us to carry out DTW between a native utterance and a nonnative utterance. The difference to a spoken term detection task is that we have to not only consider the aligned scores, but also look into the aligned path for a more detailed analysis. In the following chapters, we will explain how we apply this idea to the task of mispronunciation detection.

Chapter 3

Word Segmentation

3.1 Motivation

Assume we are given a teacher's utterance with word-level timing labels. To locate word boundaries in a student's utterance, we can align the two utterances, and map the word boundaries in the teacher's utterance to the student's utterance through the aligned path. A common property of nonnative speech is that there can sometimes be a long pause between words, since the student may need time to consider how to pronounce a certain word. For a more accurate analysis of a word, those pauses have to be removed. One possible solution may be to apply a Voice Activity Detector (VAD) on the student's input utterance before subsequent analyses. However, building a good VAD is in itself a research problem, and it would introduce computational overhead to our system.

In this chapter, we propose to perform dynamic time warping (DTW) with a silence model, which not only compares each frame in a nonnative utterance with that of a native utterance, but also considers the distance to some frames of silence. In this way, we can align the two utterances and detect silence at the same time. Experimental results show that incorporating a silence model helps our system to identify word boundaries much more precisely.

3.2 Dynamic Time Warping with Silence Model

3.2.1 Distance Matrix

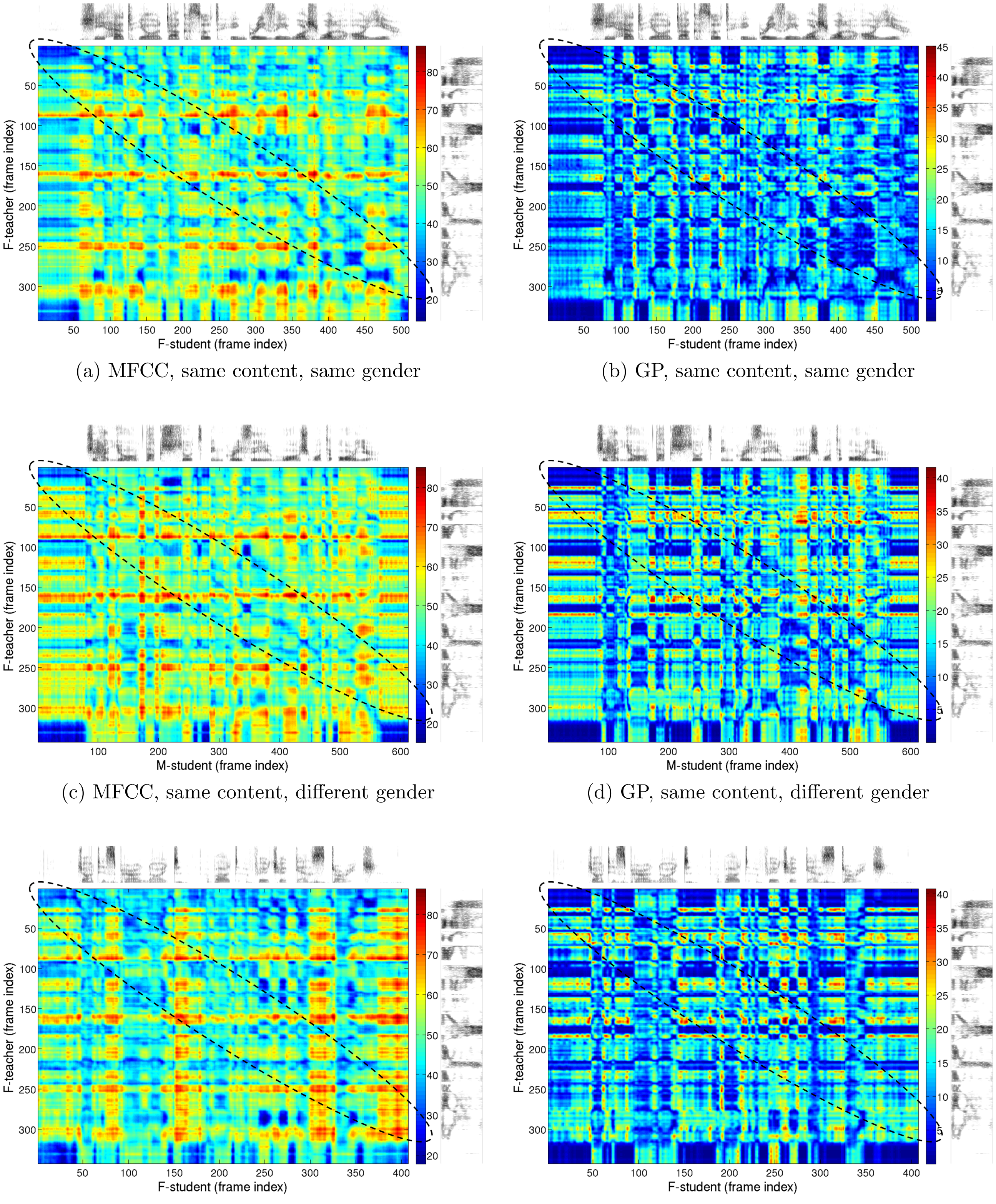
Given a teacher frame sequence $T = (f_{t_1}, f_{t_2}, ..., f_{t_n})$ and student frame sequence $S = (f_{s_1}, f_{s_2}, ..., f_{s_m})$, an $n \times m$ distance matrix, Φ_{ts}, can be built to record the distance between each possible frame pair from T and S. The definition of the distance matrix Φ_{ts} is

$$\Phi_{ts}(i, j) = D(f_{t_i}, f_{s_j}), \tag{3.1}$$

and $D(f_{t_i}, f_{s_j})$ denotes any possible distance metric between f_{t_i} and f_{s_j}. Here n is the total number of frames of the teacher's utterance and m the student's. The representation of a frame of speech, f_{t_i} or f_{s_j}, can be either a Mel-frequency cepstral coefficient (MFCC) (a 13-dim or 39-dim vector if with first and second-order derivatives) or a Gaussian posteriorgram (a g-dim vector if it is decoded from a g-mixture GMM). If we use MFCCs to represent the two utterances, $D(u, v)$ can be the Euclidean distance between two MFCC frames u and v. If we choose a Gaussian posteriorgram (GP) as the representation, $D(u, v)$ can be defined as $-\log(u \cdot v)$ [19, 38].

Fig. 3-1 shows six examples of distance matrices. We can tell that in both the MFCC-based and GP-based distance matrices between two utterances of the same content and from the same gender (Fig. 3-1a and 3-1b), there generally is a dark blue path (i. e. a low-distance path) starting from the top left corner and ending the bottom right corner along a quasi-diagonal. This observation motivates the idea aligning the two utterances for direct comparison. Things change a little when we align two utterances of the same content but produced by different genders (Fig. 3-1c and 3-1d), where the path becomes less clear because of the difference in vocal tracts. However, there is obviously no such path in the distance matrices of different content (Fig. 3-1e and 3-1f).

For the comparison between MFCC-based and GP-based distance matrices, obviously, there are more dark blue regions (i. e. low-distance regions) in GP-based distance matrices. Ideally, each mixture in the GMM should correspond to one phonetic unit in the training data. However, since the GMM was trained in an unsupervised fashion, each resulting



(a) MFCC, same content, same gender

(b) GP, same content, same gender

(c) MFCC, same content, different gender

(d) GP, same content, different gender

(e) MFCC, different content, same gender

(f) GP, different content, same gender

Figure 3-1: *Examples of distance matrices and the corresponding spectrograms. (MFCC: using MFCC to represent speech, GP: using Gaussian posteriorgram to represent speech; F: female, M: male) The colorbar next to each matrix shows the mapping between the color and the distance values. Low distances tend to be in blue, and high distances tend to be in red.*

mixture actually binds several acoustically similar units together. Therefore, when decoding, two phonetically similar frames may have a similar probability distribution over mixtures even though they are different phones, resulting in low distance between them. We can conclude that using MFCC can help us avoid confusion between different phonetic units. However, as stated in Chapter 2, many posteriorgram-based approaches have been proven to be useful in many pattern-matching-based applications. In these work, MFCCs have been shown to be sensitive to differences in vocal tracts, while the mixtures in GMM capture more difference in the acoustic characteristics of phonetic units. Therefore, at this stage, we cannot determine for sure which representation is better for our application. We therefore consider both MFCC-based and GP-based distance matrices when carrying out experiments.

3.2.2 Dynamic Time Warping (DTW)

Given a distance matrix, Φ_{ts}, we carry out global dynamic time warping (global DTW) to search for the "best" path starting from $\Phi_{ts}(1,1)$ and ending at $\Phi_{ts}(n,m)$, and the "best" path is the one along which the accumulated distance is the minimum. By doing so, the resulting path should be an "optimal" alignment between T and S.

The formal definition of the global DTW we carry out is as follows. Let $\boldsymbol{\psi}$ be a sequence of l index pairs, $((\psi_{11},\psi_{12}),(\psi_{21},\psi_{22}),...,(\psi_{l1},\psi_{l2}))$. $\boldsymbol{\psi}$ is a feasible path on an $n \times m$ distance matrix Φ if it satisfies the following constraints:

1. $\psi_{11} = \psi_{12} = 1$, $\psi_{l1} = n$ and $\psi_{l2} = m$
2. $\psi_{i+1\,1} - \psi_{i1} \leq 1$ and $\psi_{i+1\,2} - \psi_{i2} \leq 1, \forall i = 1,2,...,l-1$

The first constraint is due to $\boldsymbol{\psi}$ being a global path, and the second one comes from the fact that we are aligning two sequences in time. Let $\boldsymbol{\Psi}$ be a set of all feasible $\boldsymbol{\psi}$'s. As a result, the best alignment between T and S will be

$$\boldsymbol{\psi}^*_{ts} = \operatorname*{argmin}_{\boldsymbol{\psi}\in\boldsymbol{\Psi}} \sum_{k=1}^{l} \Phi_{ts}(\psi_{k1},\psi_{k2}). \tag{3.2}$$

The problem of searching for the best path can be formulated as a dynamic programming problem. We define an $n \times m$ cumulative distance matrix C_{ts}, where $C_{ts}(i,j)$ records the minimum accumulated distance along a legal path from $(1, 1)$ to (i, j). To find $\boldsymbol{\psi}^*_{ts}$, the first pass would be to compute each element in $\Phi^\dagger_{ts}$ from the following

$$C_{ts}(i,j) = \begin{cases} \Phi_{ts}(i,j), & \text{if } i = j = 1 \\ C_{ts}(i, j-1) + \Phi_{ts}(i,j), & \text{if } i = 1, 1 < j \leq m \\ C_{ts}(i-1, j) + \Phi_{ts}(i,j), & \text{if } j = 1, 1 < i \leq n. \\ \min(C_{ts}(i-1,j), C_{ts}(i,j-1), C_{ts}(i-1,j-1)) + \Phi_{ts}(i,j), & \text{otherwise} \end{cases} \tag{3.3}$$

When computing, we keep track of which direction, i. e. $(i-1, j), (i, j-1)$ or $(i-1, j-1)$, leads to a minimum cumulative distance along a path ending at position (i, j). Then, the second pass backtraces from (n, m) for the recorded indices.

3.2.3 DTW with Silence Model

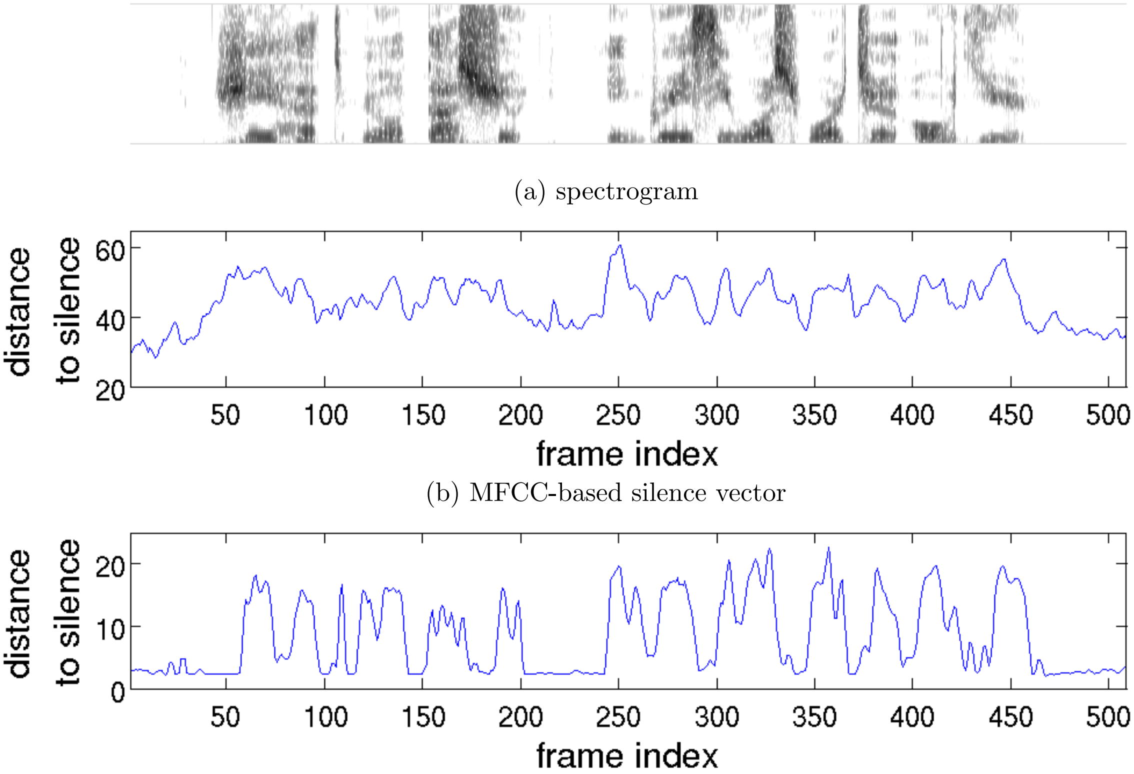
In a comparison-based framework, to determine whether a frame in the student's utterance is silence or not, we look at whether it is "close" to silence. In other words, we can also calculate its distance to silence.

We define a $1 \times m$ silence vector, ϕ_{sil}, which records the average distance between each frame in S and r silence frames in the beginning of T. ϕ_{sil} can be computed as

$$\phi_{sil}(j) = \frac{1}{r} \sum_{k=1}^{r} D(f_{t_k}, f_{s_j}) = \frac{1}{r} \sum_{k=1}^{r} \Phi_{ts}(k, j). \tag{3.4}$$

Fig. 3-2 shows two examples of silence vectors, one obtained from an MFCC representation, and one from a GP, and the corresponding spectrograms. From the spectrogram we can see that there are three long pauses in the utterance, one at the beginning from frame 1 to frame 44, one at the end from frame 461 to frame 509, and one intra-word pause from frame 216 to frame 245. In the silence vectors, these regions do have relatively low average distance to the first 3 silence frames from a reference utterance compared to that of other frames.

To incorporate ϕ_{sil}, we consider a modified $n \times m$ distance matrix, Φ'_{ts}. Let B_t be a set



(a) spectrogram

(b) MFCC-based silence vector

(c) GP-based silence vector

Figure 3-2: *The spectrogram (in (a)) and the corresponding silence vectors ϕ_{sil}, (b): MFCC-based and (c): GP-based, with $r = 3$*

of indices of word boundaries in T. Then, each element in Φ'_{ts} can be computed as

$$\Phi'_{ts}(i,j) = \begin{cases} \min(\Phi_{ts}(i,j), \phi_{sil}(j)), & \text{if } i \in B_t \\ \Phi_{ts}(i,j), & \text{otherwise} \end{cases} \tag{3.5}$$

At word boundaries of the native utterance, Φ'_{ts} would be $\phi_{sil}(j)$ if it is smaller than $\Phi_{ts}(i,j)$, i.e. s_j is closer to silence. If there is a segment of pause in S, say $(f_{s_i}, f_{s_{i+1}}, \dots f_{s_j})$, there would be a low-distance horizontal "silence" band in Φ'_{ts}, as the value of each element in that band comes from ϕ_{sil}. The ordinary DTW can be carried out on Φ'_{ts} to search for the best path. While backtracing, if the path passes through elements in Φ'_{ts} that were from ϕ_{sil}, we could determine that the frames those elements correspond to are pauses.

Another way to think of running DTW on this modified distance matrix, Φ'_{ts}, is to allow the path to jump between the original distance matrix Φ_{ts} and the silence vector ϕ_{sil}, i.e.

consider the possibility of s_j being a frame of silence, at word boundaries in T when running the original DTW. Therefore, we can reformulate the first pass of the original DTW. To simplify the notation, we further define

$$\phi_s(i,j) = \begin{cases} \infty, & i \notin B_t \\ \phi_{sil}(j), & i \in B_t \end{cases} \tag{3.6}$$

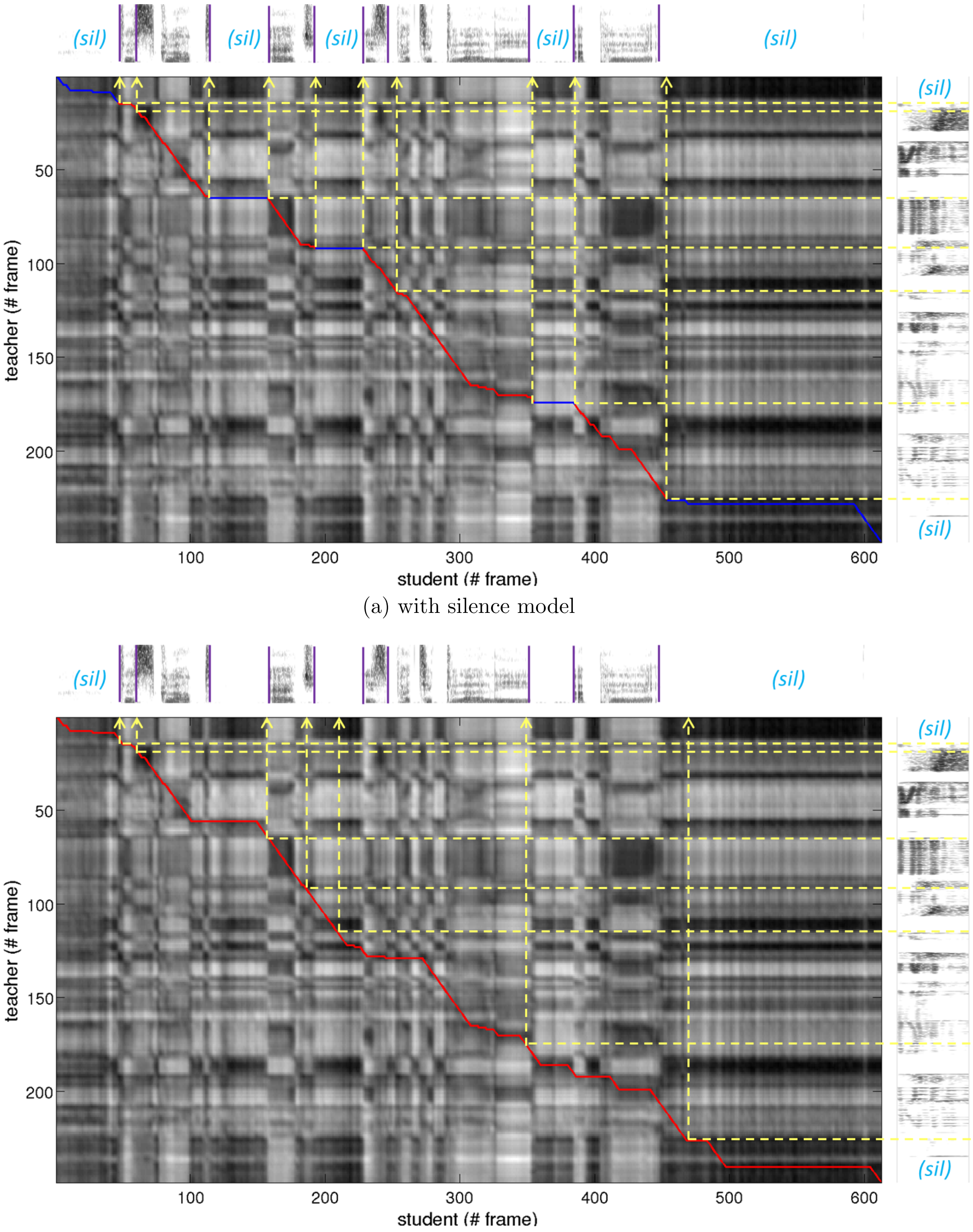
Then, Eq. 3.3 becomes

$$C_{ts}(i,j) = \begin{cases} \min(\Phi_{ts}(i,j), \phi_s(i,j)), & \text{if } i = j = 1 \\ C_{ts}(i,j-1) + \min(\Phi_{ts}(i,j), \phi_s(i,j)), & \text{if } i = 1, 1 < j \leq m \\ C_{ts}(i-1,j) + \min(\Phi_{ts}(i,j), \phi_s(i,j)), & \text{if } j = 1, 1 < i \leq n \\ \min(C_{ts}(i-1,j), C_{ts}(i,j-1), C_{ts}(i-1,j-1)) & \\ \quad + \min(\Phi_{ts}(i,j), \phi_s(i,j)), & \text{if } 1 < i \leq n, 1 < j \leq m \end{cases} \tag{3.7}$$

When computing C_{ts}, we keep track of both the direction and also whether an element comes from ϕ_{sil} or not. The second pass uses the same backtracing procedure as before, but this time the path contains extra information about whether each element belongs to silence or not.

Locating word boundaries in S is then easy. We first remove those pauses in S according to the information embedded in $\boldsymbol{\psi}^*_{ts}$. Then, we can map each word boundary in T through $\boldsymbol{\psi}^*_{ts}$ to locate boundaries in S. If there are multiple frames in S aligned to a boundary frame in T (i. e. a horizontal segment in the path), we choose the midpoint of that segment as the boundary point.

Fig. 3-3 shows an example of how we locate word boundaries, and how including a silence model affects DTW alignment. We can see that the silence model not only helps us detect intra-word pauses (the blue segments in Fig. 3-3a), but also affects the direction of the aligned path, especially when there are no intra-word pauses in the reference utterance, which is usually the case. If there is no silence between words in the native utterance, the silence in the nonnative utterance may not have low distance to the beginning of the next word in



(a) with silence model

(b) without silence model

Figure 3-3: *Comparison of the aligned path from MFCC-based DTW (a) with, and (b) without silence model. The red line shows the aligned path* $\boldsymbol{\psi}^*_{ts}$*, and the blue segments in (a) indicate the silence detected from the silence model. The purple lines on the student's spectrogram mark the word boundaries. The yellow dotted lines illustrate how we locate word boundaries by finding the intersection between the word boundaries in the teacher's utterance and the aligned path. (the script: "the scalloped edge is particularly appealing")*

the native utterance when we perform alignment at a word boundary. Thus, if there is no silence model to consider, the path would choose a warp that can lead to lower accumulated distance.

3.3 Experiments

In this stage, we focus on examining how well DTW with a silence model can capture the word boundaries in nonnative speech.

3.3.1 Dataset and Experimental Setups

We use nonnative data in both the training set and the test set for evaluation, resulting in 2,261 utterances, including 19,732 words. Ground truth timing information on the nonnative data is generated by running a standard SUMMIT recognizer [40] for forced alignment. We extract 39-dimensional MFCC vector, including first and second order derivatives, at every 10-ms frame. A 150-mixture GMM is trained on all TIMIT data. Note that we did not include the CU-CHLOE data into GMM training since experimental results show that this would make the mixtures implicitly discriminate between native and nonnative speech instead of different phones.

For evaluation, we consider

1. *deviation in frame*: the absolute difference in frame between the detected boundary and the ground truth,

2. *accuracy within a 10-ms window*: the percentage of the detected boundaries whose difference to the ground truth are within 1 frame,

3. *accuracy within a 20-ms window*: the percentage of the detected boundaries whose difference to the ground truth are within 2 frames.

On average, there are 2.34 same-gender reference utterances for one nonnative utterance. If there is more than one reference native utterance for an utterance, the one that gives the best performance is considered.

3.3.2 Word Boundary Detection

We first examine how the parameter r, the size of the silence window in the beginning of each teacher's utterance, affects the performance. Fig. 3-4 shows that the performance is relatively stable with respect to different r's, as long as the window size is not so big that it includes some non-silence frames.

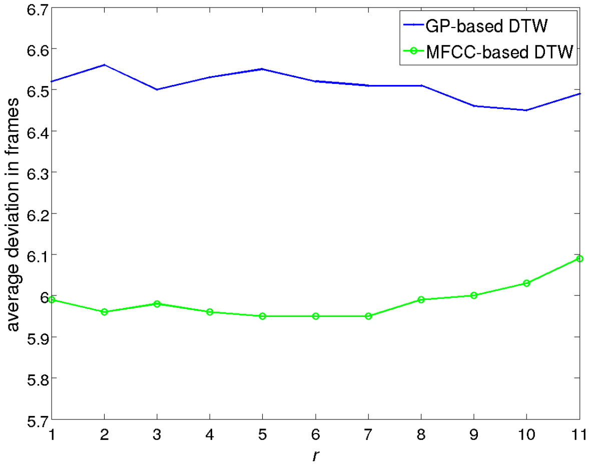


Figure 3-4: *Performance in terms of average deviation in frames vs. different sizes of the silence window*

To understand how incorporating a silence mechanism can help improve the performance, four scenarios are tested as shown in Table 3.1. Compared with the cases where no silence model is considered, with the help of the silence model, MFCC-based DTW obtains a 40.6% relative improvement and GP-based DTW has a 31.6% relative improvement in terms of deviation in frames. In both cases, more than half of the detected word boundaries are within a 20-ms window to the ground truth.

	deviation (frames)	accuracy ($\leq$ 10ms)	accuracy ($\leq$ 20ms)
MFCC	10.1	35.2%	45.2%
GP	9.5	38.2%	47.7%
GP+sil	6.5	41.5%	51.9%
MFCC+sil	6.0	42.2%	53.3%

Table 3.1: *Performance of word segmentation from same-gender alignment under different scenarios (MFCC: MFCC-based DTW, GP: GP-based DTW, sil: silence model)*

The silence model helps both GP and MFCC-based approaches due to the significant

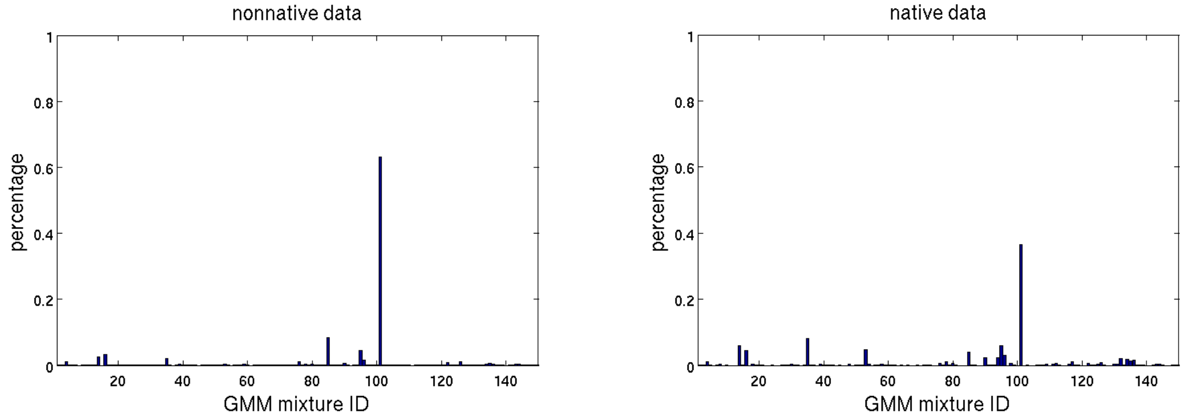
	precision	recall	f-score
MFCC-based	90.0%	77.4%	83.2%
GP-based	86.1%	72.3%	78.6%

Table 3.2: *Performance of the silence model detecting silence frames based on best aligned pairs*

amount of hesitation between words in the nonnative data. In fact, the total time duration of these 2,261 utterances is about 226 minutes long, with 83 minutes of silence (37.0%). If we view the silence model as a VAD and evaluate its performance, the results are shown in Table 3.2, where *precision* is the ratio of the number of correctly detected silence frames to the total number of detected frames, *recall* is the ratio of the number of correctly detected silence frames to the total number of silence frames in ground truth, and *f-score* is the harmonic mean of the two. We can see that for both MFCC-based and GP-based approaches, the model can detect most of the silence frames with high precision, and thus, the word boundaries can be more accurately captured.

Moreover, the higher f-score of MFCC-based DTW for detecting silence can also account for the gap between the performance of MFCC-based and GP-based approaches as shown in Fig. 3-4. There are two possible explanations of the lower performance of the GP-based approach. First of all, there may be more than one mixture in the unsupervised GMM that captures the characteristics of silence. Therefore, when decoding, silence frames in different utterances may have different distributions over the mixtures, and thus the distance would not be very low. Fig. 3-5 shows the normalized histogram over the most likely component, i. e. the mixture with the highest posterior probability after decoding, of all silence frames in the native and nonnative utterances. Although the most likely component that corresponds to silence is the 101^{th} mixture component for both cases, there are over 60% silence frames in the nonnative data having this behavior, while only around 40% in the native data. The mismatch between the decoded probability distributions causes the problem.

One may argue that the above problem is solvable by decreasing the number of mixtures when training the GMM so that one phonetic unit can be captured by exactly one component. However, empirical experiences have shown that this would only cause a higher degree of confusion and even further decrease the granularity in analysis. Therefore, choosing the



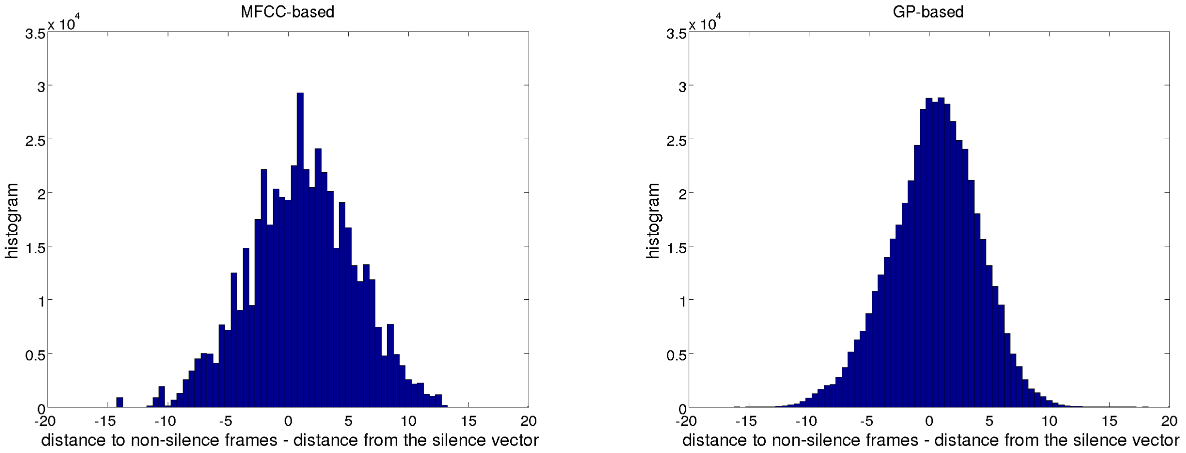
(a) (b)

Figure 3-5: *Normalized histograms of the most likely component for silence frames*

number of mixtures is an important issue when considering posteriorgram-based approaches. Another possible solution might be to train a special silence Gaussian by using the first r silence frames from the training data, and use the rest of the frames to train a GMM. Then, $\phi_s(i,j)$ can be computed from the posteriorgrams decoded from the special silence Gaussian, and $\Phi_{ts}(i,j)$ can be computed from the posteriorgrams decoded from the GMM.

The second explanation of the lower performance of GP-based DTW is the lower discriminability of the GP-based distance measure. To show this, we compute the difference between the average distance between a silence frame in the nonnative utterance, as well as the boundary frames in the native utterance, and the average distance between the same silence frame to the first r silence frames in the native utterance. In other words, for each silence frame f_{s_j} in S, we compute $\frac{1}{|B_t|}\sum_{i\in B_t}\Phi_{ts}(i,j)-\phi_{sil}(j)$. If this number is greater than 0, Φ'_{ts} will choose the element from ϕ_{sil} (recall Eq. 3.5) and correctly record the frame in the nonnative utterance as silence. Fig. 3-6 shows the histograms of the results from silence frames of all nonnative utterance. For an MFCC representation, there are 37.0% of the silence frames having lower distance to the non-silence boundary frames in the reference utterance, while for a GP representation, it is 40.4% of the data. These numbers tell us that there is a slightly higher probability for the GP-based DTW of misidentifying silence frames as non-silence ones, and thus the performance on word boundary detection would be lower.

In the above analyses, we only consider the reference speaker that gives the best perfor-



(a) MFCC-based

(b) GP-based

Figure 3-6: *Histograms of the average distance between one silence frame in S to all non-silence frames in T minus the corresponding distance from the silence vector ϕ_{sil}, i. e. the average distance between the given silence frame in S to r silence frames in T, in terms of either MFCC or GP representation*

mance if there is more than one reference utterance for a script. Here we examine how poor the result could be for each scenario in order to understand the range of the performance. Table 3.3 lists the best and the worst performance, as well as the results when we randomly pick one native speaker as the teacher for each script. There are two things worth noticing from the table. First, the worst performances of the scenarios with a silence model are still better than the best performances of those without the model. This again shows the fact that there are plenty of intra-word pauses, and removing them is one of the keys to improving the detection of word boundaries. However, the f-score on detecting silence frames based on the worst aligned pairs is 80% for MFCC-based and 76% for GP-based DTW, which are not too

deviation (frames)	MFCC	MFCC+sil	GP	GP+sil
best	10.1	6.0	9.5	6.5
worst	14.4	9.2	12.7	9.0
random	11.2 (0.1)	7.2 (0.1)	10.8 (0.1)	7.6 (0.1)

Table 3.3: *Comparison between the best and the worst performance of word boundary detection, focusing on deviation in frames between the detected boundary and the ground truth. The last row comes from randomly picking one reference utterance during evaluation, and the whole process is carried out 10 times. Both average (and standard deviation) of the performance is listed for the random case.*

bad compared to the numbers in Table 3.2. Therefore, another factor affecting performance might be the essence of the native utterances. We have carried out various analyses on how the quality of the native utterances relate to the performance on word boundary detection. The speaking rate, the time duration of the utterance, the length of silence within the utterance, the dialect region of the native speaker, and the normalized accumulated distance along the aligned path are several components we have inspected. However, there is no obvious relation between all the above factors and the performance. As a result, we believe the performance is more related to the interaction between the voice characteristics of the native and nonnative speaker.

Second, the results from randomly choosing the reference speakers gives us an idea of how the performance would be in general. This is particularly important, since, in real application, we won't have access to the word boundary information of the student's utterance, it would be hard to determine which native speaker would give the best result. From the table we can see that the performance of alignments based on randomly chosen references is closer to that of the best alignments, and the low standard deviations suggest that the worst case does not occur with high probability. In order to further improve the performance of word boundary detection, one may consider voting from multiple reference speakers, or even combining the outputs from MFCC-based and GP-based DTW. However, we will leave this as future work.

3.3.3 Comparison of Alignment Between Same/Different genders

In the previous sub-section, we only considered alignments based on teacher and student pairs of the same gender. However, it is well known that MFCCs are sensitive to vocal tract shape, and unsupervised GMMs might have a problem of having some mixtures implicitly capturing the characteristic of one gender and some other mixtures capturing the other gender. Here we examine how the performance is when aligning two utterances from different genders.

In order to compare the two settings, we use a subset of 1,000 nonnative utterances that have reference speakers of both genders, with a total of 7,617 words. There are, on average, 3.6 reference speakers of the same gender, and 3.3 of the different gender, for each nonnative utterance in this subset. Table 3.4 shows the experimental results. We can see that alignments based on different-gender speaker pairs have greatly affected the worst case scenario. For the best case, the performance dropped about 1.2 frames in average, while that of the worst case dropped about 3.2 frames, and this also leads to an average of 2.0-frame increase in the random case. These results show that alignment from different-gender pairs of speakers do have lower performance.

In our current framework, we are assuming that the gender of the student is known. If such information is missing, a possible solution is to train a male GMM and a female GMM by using all data from the two genders, respectively. Given an input utterance from the student, we can decode the utterance by using both GMMs, see whether the input utterance is more likely to be a male utterance or a female one, and then use the GMM that matches the best to decode the final posteriorgrams. Still, in this stage, we are not sure how this 2.0 frame difference will affect the overall performance on mispronunciation detection. Therefore, in Chapter 5, we will carry out DTW between different genders and focus on the output of the whole framework.

	deviation (frames)	MFCC	MFCC+sil	GP	GP+sil
same-gender	best	7.3	4.5	7.1	5.1
	worst	17.6	12.2	15.0	11.1
	random	11.5	7.5	10.5	7.8
different-gender	best	8.8	5.5	8.3	6.1
	worst	21.0	15.1	18.2	14.3
	random	13.9	9.4	12.5	9.4

Table 3.4: *Comparison between the performance based on same/different-gender alignment. The best and the worst performance are listed, as well as results based on randomly picked reference speakers, averaged across 10 trials.*

3.4 Summary

In this chapter, we have presented the first stage of our mispronunciation detection framework – word boundary detection. Intra-word pauses happen very often in nonnative speech. Removing pauses can not only give us more precise knowledge about where the word boundaries are in a nonnative utterance, but also helps the subsequent analysis, since we will extract features based on the aligned path and the distance matrix, if there is silence in the student's utterance but not in the teacher's, then there will definitely be a mismatch. To deal with this problem, instead of building a speech/non-speech classifier, we extend basic DTW to incorporate a silence model. The silence model computes the distance of each frame in the nonnative utterance to some frames of silence. When running DTW, the path can consider the possibility of a frame in student's utterance being silence, instead of trying to align it with frames from the teacher's utterance that are very different from it. This approach does not add complexity to the original DTW, since it just provides one more choice for the path to consider. Another advantage of this method is that to compute the silence vector ϕ_{sil}, we need to average only the first few rows of the distance matrix Φ (which is already computed). In other words, we don't require extra data or annotations in order to obtain a good silence model. This does assume the beginning of the recording is silence however.

Experimental results have shown that DTW works well in aligning a native utterance with a nonnative one. This also implies that DTW-based keyword spotting approaches can work on nonnative speakers. Second, the silence model can detect pauses between words with a high f-score of 83%, and improves the relative frame deviation performance on locating word boundaries in nonnative speech by over 40%.

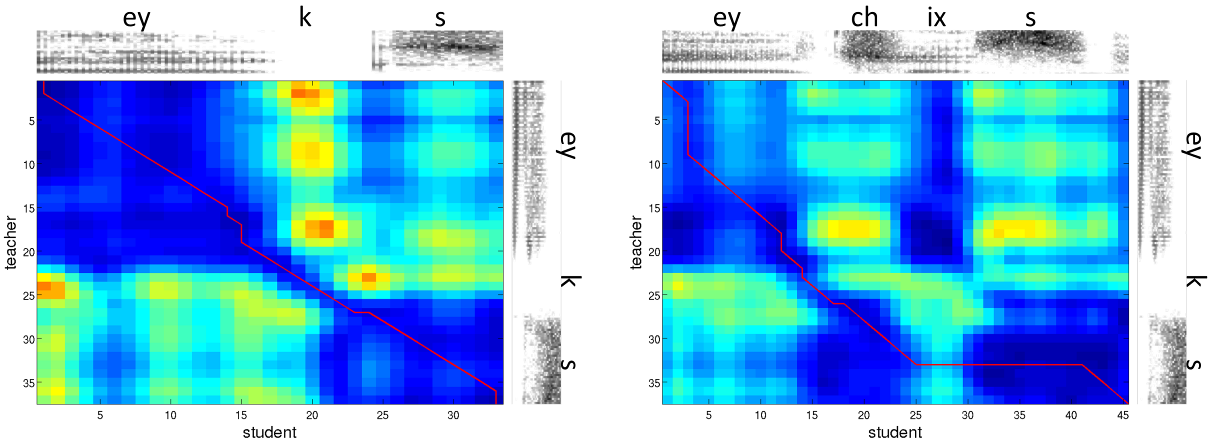
Chapter 4

Unsupervised Phoneme-like Unit Segmentation

4.1 Motivation

When aligned with a teacher's utterance, a good pronunciation and a bad one will have different effects on the distance matrix. Fig. 4-1a shows the alignment between a teacher and a student who pronounced the word "aches" correctly as /*ey k s*/, while Fig. 4-1b is the alignment between the same teacher and a student who mispronounced the word as /*ey ch ix s*/. The most obvious difference between the two is that there is a high-distance region in Fig. 4-1b, which is the distance between /*ch ix*/ in the student's utterance and /*k*/ in the teacher's.

Ideally, a good alignment between two sequences should be of 45° (i. e. diagonal in distance matrix). However, this is usually not the case, since the speaking rate of two speakers will not be exactly the same, and that of a nonnative speaker is usually slower. Fig. 4-2 contains a histogram of the time difference between a student and a teacher saying the same sentence, after removing silence in each utterance and considering both the training and testing data. It is clear that most of the time, a student spends more time pronouncing a given script than a teacher. Also, the relative duration of a phone within a word in nonnative speech and native speech are often different. Again, take Fig. 4-1 for example. The phone /*ey*/ pronounced by the teacher is about 21-frame long, taking up 57% of the whole word, while that pronounced by student (a) is about 17-frame long and takes up about 52%, and by (b) is about 15-frame long with a relative ratio of 33%.



(a) good pronunciation

(b) mispronunciation

Figure 4-1: *GP-based distance matrices of alignment between a teacher saying "aches" and (a) a student who pronounced correctly, (b) a student who mispronounced as /ey ch ix s/*

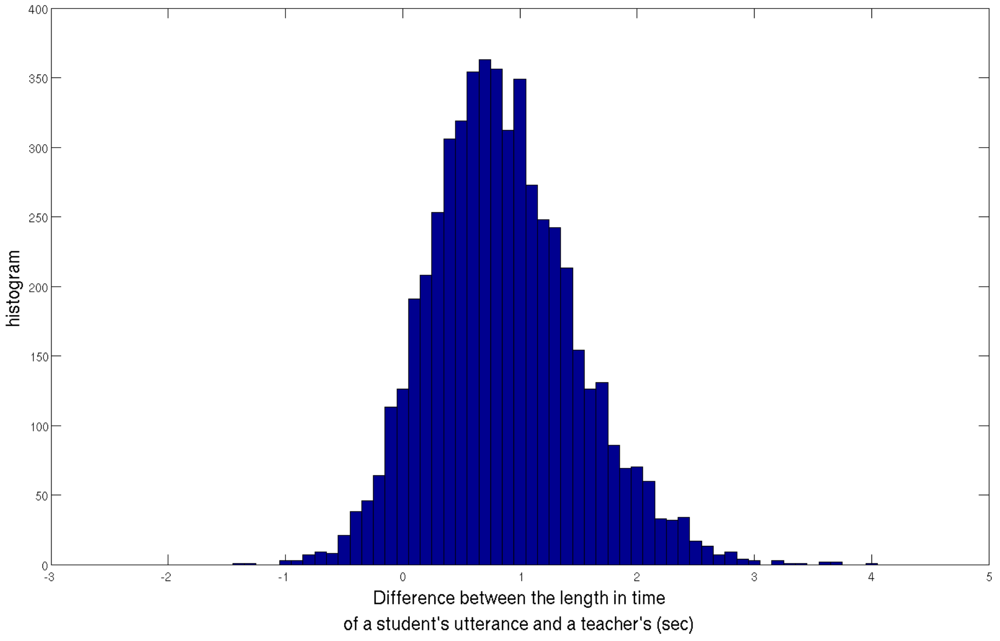


Figure 4-2: *Difference in time duration between a student and a teacher saying the same sentence*

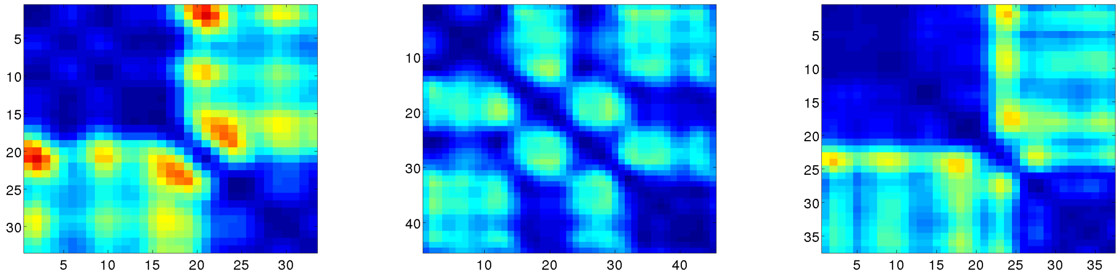
Since neither the absolute time duration, nor the relative duration of a phone within a word is the same between a student and a teacher, the aligned path in the distance matrix would not be 45° (diagonal), even though the student may pronounce the phone correctly. Therefore, before extracting features, we need to further divide each word into smaller phone-like units for analysis. Then, word-level and phone-level features can be extracted to predict whether a word is mispronounced or not.

4.2 Related Work

Segmenting speech into sub-word units, such as phones, has been a research topic for many years, since accurate segmentation can potentially improve the effectiveness of training data for speech recognition. An HMM-based forced alignment is the most commonly used approach to segment an utterance into phone-level units. However, it requires pre-trained acoustic models and linguistic knowledge. In the 1970's, Bridle and Sedgwick [6] proposed an acoustic segmentation method that was based on successively sub-dividing an utterance to optimize a segmentation criterion, which was the sum of the squares of the approximation errors. Cohen [8] developed a probabilistic model for segmenting speech into phoneme-size units, and introduced how to set a prior on segment length and compute the likelihood. Glass et. al [17] dealt with the segmentation problem by representing the speech signal with a multi-level acoustic description, in which each level corresponds to acoustically meaningful units.

More recently, Dusan et. al [11] introduced a spectral transition measure (STM), which is the mean squared value of the rate of change of each dimension of the MFCC. After calculating STM for every frame, a peak picking method and a post-processing method are proposed and used to detect phone boundaries. Estevan et. al [14] adopt maximum margin clustering (MMC), which is an unsupervised two-class classification technique, at every frame along the time index. After clustering, the distance between the two clusters is also computed, and a distance vs. time relation can be obtained in the end. If at a specific time, the distance is low, this means that the samples near that time index are similar to each other. As a result, boundaries can be determined by locating peaks in the distance vs.

time relation. Qiao et. al [30] assume that the number of phones in an utterance is known beforehand, formulate the segmentation problem in a probabilistic framework, and develop three different objective functions based on statistics and information theory analysis. While all of the above are one-stage and bottom-up segmentation methods, Lee et. al [23] further integrate the tasks of segmentation, clustering segments and modeling each cluster into a complete acoustic modeling framework. Their process is iterative: sub-word models are trained based on the hypothesized phone boundaries, and the newly learned models are used to segment the utterances.



(a) /ey k s/

(b) /ey ch ix s/

(c) /ey k s/

Figure 4-3: *GP-based self aligned matrices of (a): the student in Fig. 4-1a, (b): the student in Fig. 4-1b, and (c): the teacher in both Fig. 4-1a and 4-1b*

4.3 The Self-Similarity Matrices (SSMs)

Let Φ_{tt} be the self-aligned distance matrix of T, where $\Phi_{tt}(i,j) = D(f_{t_i}, f_{t_j})$. Φ_{tt} is a square, symmetric, matrix and symmetric from the diagonal (see Fig. 4-3). Along the diagonal, each low-distance block indicates frames that are phonetically-similar. These frames may relate to a single phoneme, a part of a diphthong, or a chunk of acoustically-similar phonemes. Therefore, to segment a word into smaller phoneme-like units, we can focus on finding the boundaries between those blocks.

Jensen has extracted features related to the rhythm and tempo of a pop song and computed the self-similarity matrices for the task of music segmentation [20]. Similarly, blocks along the diagonal may indicate a structural segment in music, such as an intro, chorus, verse, bridge or outro. Here we first deployed his idea of segmenting the SSMs and further incorporate prior knowledge on the length of each segment in hopes of regulating the size of the resulting segments.

4.4 Optimizing the Unit Boundaries

4.4.1 Known number of phones in a word

Following the formulation in [20], given the number of phones in a word, K, we can determine the boundaries by minimizing the sum of the average distance in the lower-triangle of each possible block:

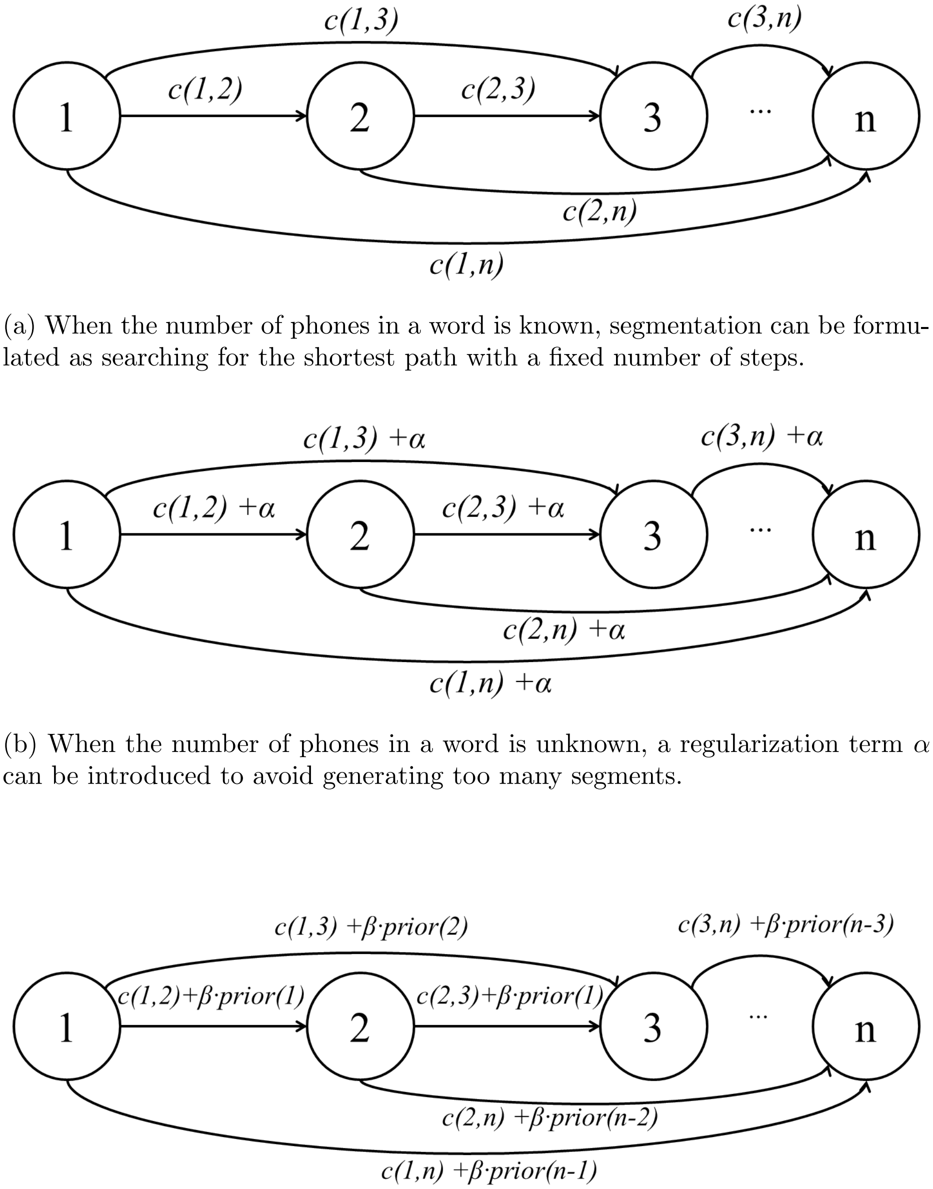
$$(b_0^*, b_1^*, ..., b_{K-1}^*) = \underset{\substack{(b_0, b_1, ..., b_{K-1}) \\ 1=b_0<b_1<...<b_{K-1}\leqslant n}}{\operatorname{argmin}} \sum_{z=1}^{K-1} \frac{1}{b_z - b_{z-1}} \sum_{y=b_{z-1}}^{b_z-1} \sum_{x=b_{z-1}}^{y} \Phi_{tt}(y, x), \tag{4.1}$$

where $(b_0, b_1, ..., b_{K-1})$ are the K possible starting indices of each segment. The whole optimization process can be regarded as a shortest-path searching problem on a graph as shown in Fig. 4-4a, with the constraint that the path length is known to be K. Thus, the optimization problem can be solved using dynamic programming (refer to Appendix A). One thing worth mentioning is that when averaging the distance, we only divide the total distance by the length of the segment instead of the area of the block. This is because empirical results have shown that dividing by the area would cause the high distance region in a large segment to be averaged out too much and thus bring only little influence on the boundary decision [20].

4.4.2 Unknown number of phones in a word

If the number of phones in a word is unknown, we can include that into the optimization process and perform joint optimization on the segment boundaries as well as the number of segments. Here we denote the unknown number of segments as K_u, and the formulation becomes

$$(b_0^*, b_1^*, ..., b_{K_u^*-1}^*, K_u^*) = \underset{\substack{(b_0, b_1, ..., b_{K_u-1}, K_u) \\ 1=b_0<b_1<...<b_{K_u-1}\leqslant n \\ 1\leq K_u\leq n}}{\operatorname{argmin}} \sum_{z=1}^{K_u-1} \alpha + \frac{1}{b_z - b_{z-1}} \sum_{y=b_{z-1}}^{b_z-1} \sum_{x=b_{z-1}}^{y} \Phi_{tt}(y, x), \tag{4.2}$$



(a) When the number of phones in a word is known, segmentation can be formulated as searching for the shortest path with a fixed number of steps.

(b) When the number of phones in a word is unknown, a regularization term α can be introduced to avoid generating too many segments.

(c) A prior, which is a function of segment length, can be further incorporated to guide the process to generate segments of reasonable length.

Figure 4-4: *A graphical view to the problem of segmentation. An arc between state i and j represents a segment starting at i and ending at $j-1$. $c(i,j)$ is the cost introduced by the corresponding segment, which equals to $\frac{1}{j-i}\sum_{y=i}^{j-1}\sum_{x=i}^{y}\Phi_{tt}(y,x)$.*

where the parameter α is introduced as a regularization term to avoid generating too many segments. If it is large, then the total penalty would be high and thus fewer segments would be favored. Similarly, this formulation can be transformed into a shortest-path searching problem on a graph shown in Fig. 4-4b, and K_u^* would be the length of the resulting shortest path.

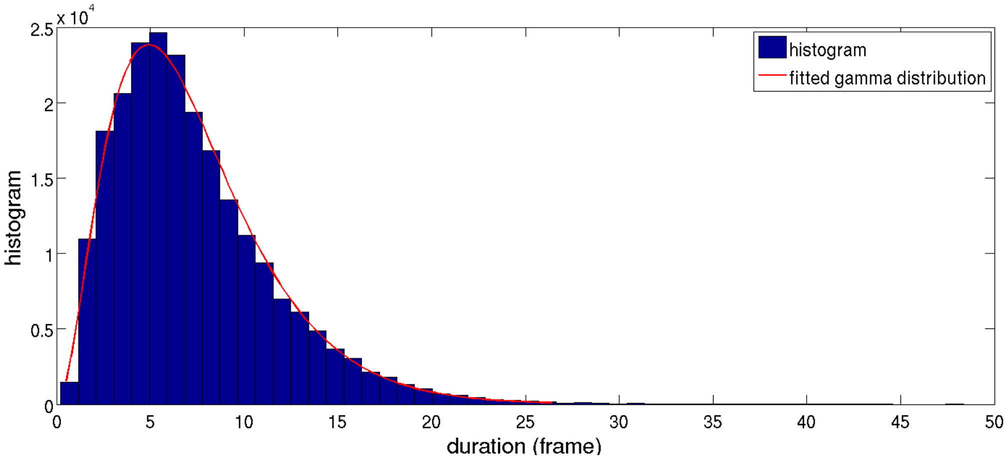


Figure 4-5: *Histogram of phone length and the fitted gamma distribution*

4.4.3 Including a prior on segment length

The α in Eq. 4.2 proposed by Jensen [20] seems somewhat heuristic, since it is only a scalar, and thus it only cares about the number of segments but does not consider what kind of segments have been generated. Fig. 4-5 is the histogram of phone duration in frame from all utterances in the TIMIT corpus. We can see that there is obviously a distribution over phone duration. In other words, the duration of a phone is more likely to fall within a certain region than the others. Therefore, we can fit the histogram by using a gamma distribution, which is commonly used in speech processing techniques to model duration, and treat it as a prior to guide the segmentation. As a result, Eq. 4.2 becomes

$$(b_0^*, b_1^*, ..., b_{K_u^*-1}^*, K_u^*) = \underset{\substack{(b_0, b_1, ..., b_{K_u-1}, K_u) \\ 1=b_0<b_1<...<b_{K_u-1}\leqslant n \\ 1\leq K_u \leq n}}{\operatorname{argmin}} \sum_{z=1}^{K_u-1} \beta \times prior(b_z - b_{z-1}) + \frac{1}{b_z - b_{z-1}} \sum_{y=b_{z-1}}^{b_z-1} \sum_{x=b_{z-1}}^{y} \Phi_{tt}(y, x), \tag{4.3}$$

where β is a weight between the prior and the average distance. We set $prior(len)$ to be $-\log(\Gamma(len, \theta))$, where θ is the parameters from the fitted gamma distribution. In fact, for the case where the number of phones in a word is known beforehand, we can also incorporate the same prior to force the segments that were generated to have a more reasonable length. Thus, Eq. 4.1 can be rewritten as

$$\begin{aligned}&(b_0^*, b_1^*, ..., b_{K-1}^*) = \\ &\underset{\substack{(b_0,b_1,...,b_{K-1}) \\ 1=b_0<b_1<...<b_{K-1}\leqslant n}}{\operatorname{argmin}} \sum_{z=1}^{K-1} \beta \times prior(b_z - b_{z-1}) + \frac{1}{b_z - b_{z-1}} \sum_{y=b_{z-1}}^{b_z-1} \sum_{x=b_{z-1}}^{y} \Phi_{tt}(y, x).\end{aligned} \tag{4.4}$$

4.5 Experiments

4.5.1 Dataset

We evaluate the performance of our phoneme-like unit segmentor on the task of phonetic boundary detection. Note that our goal is to segment SSMs into low-distance blocks along the diagonal, and each of those regions may not necessarily correspond to one single phone. Though our goal is not exactly the same as the task, the result can still give us some basic ideas about how the segmentor works.

The TIMIT training set, which includes 4,620 utterances consisting of 172,460 phones, is a commonly used dataset for evaluating speech segmentation since it consists of reliably hand labelled data. *Precision*, *recall* and *f-score* are the three metrics we use for evaluation. A 20-ms tolerance window is allowed between a true boundary and a detected boundary. We examine both the cases where the number of segments is known and unknown, and compare the results with those from the one-stage segmentation framework.

4.5.2 Experimental Results

Four scenarios were examined: either an MFCC-based, or GP-based SSM, with or without a prior. For the case where the number of phones in an utterance is known, the precision is the same as the recall since the number of the detected boundaries is the same as the number of the reference boundaries. We can see that MFCC-based approaches perform better than

GP-based ones because of the confusion between acoustically similar phonetic units that the GP representation has. This also explains why the prior helps GP-based segmentation but not MFCC-based segmentation, as the acoustically similar region in a GP-based SSM will be larger, the prior can guide the optimization process to segment each unit into a more reasonable size. After carefully examining the results from a number of utterances, we found that many of the missed boundaries are related to stops. As the duration of the release might be very short, deleting the boundary between the closure and the release only merges a small portion of high-distance into the combined segment, and this cost can be amortized by other segments, which is a classic problem with averaging.

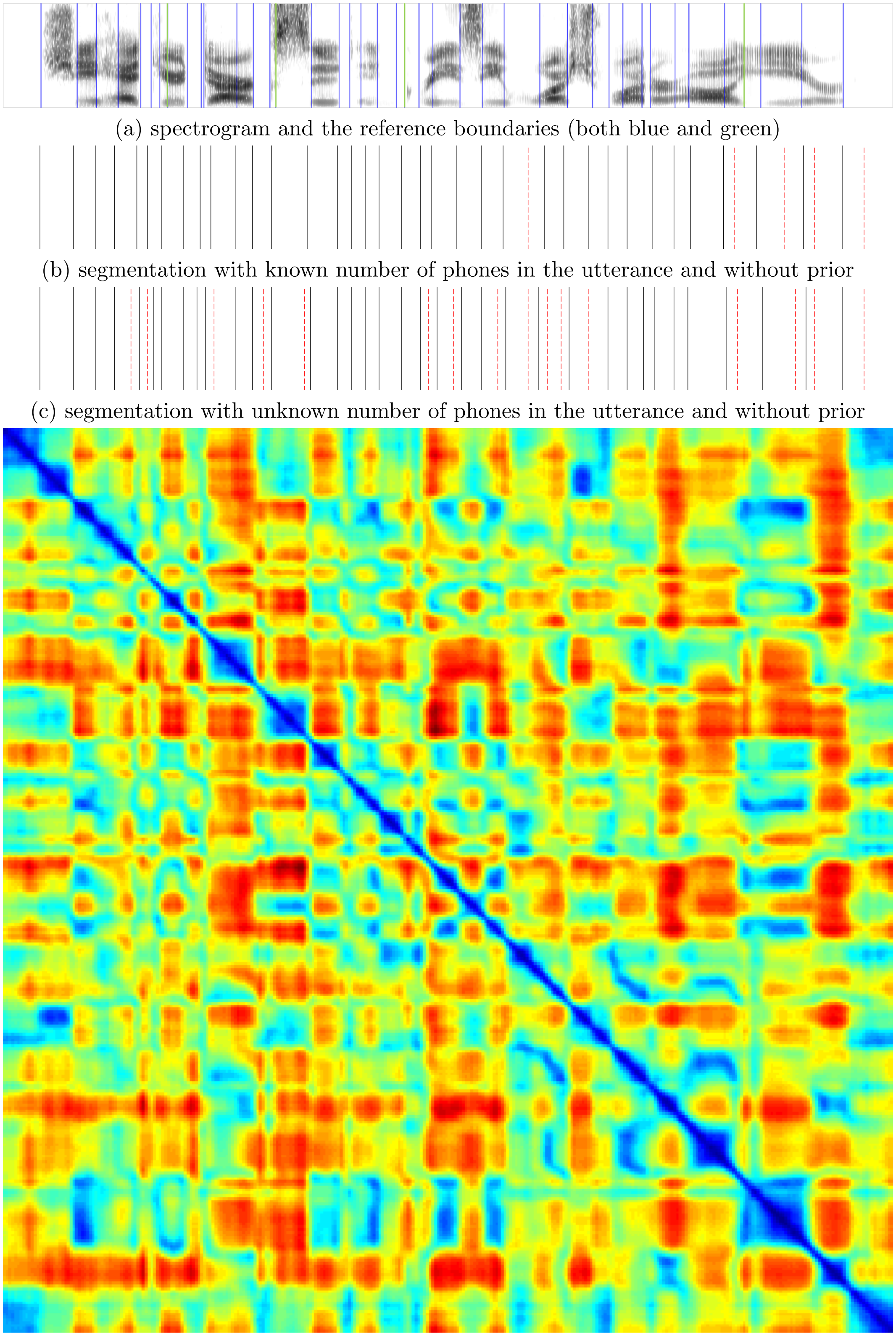
		precision (%)	recall (%)	f-score (%)
known # phones	MFCC w/ prior ($\beta = 1$)	74.8	74.8	74.8
	MFCC w/o prior	74.8	74.8	74.8
	GP w/ prior ($\beta = 5$)	72.3	72.3	72.3
	GP w/o prior	71.5	71.5	71.5
	Qiao et. al [30]	77.5	77.5	77.5
unknown # phones	MFCC w/ prior ($\beta = 20$)	63.2	76.7	69.3
	MFCC w/o prior ($\alpha = 30$)	64.3	80.3	71.4
	GP w/ prior ($\beta = 5$)	60.4	80.0	68.9
	GP w/o prior ($\alpha = 10$)	61.5	77.1	68.4
	Dusan et. al [11]	66.8	75.3	70.8

Table 4.1: *A summary of the results*

When the number of phones in an utterance is unknown, the performance drops because of the lower precision. Fig. 4-6 shows an example of the resulting segmentation of an utterance whose SSM is shown in Fig. 4-6d. In a fully unsupervised case (i. e. without knowing the number of segments beforehand), the segmentor tends to over-generate boundaries (see Fig. 4-6b), and again, the prior helps GP-based segmentation a little, but not the MFCC-based one. Compared with Dusan and Rabiner [11], our approach improves the performance slightly when segmenting MFCC-based SSM with a fixed regularization term α.

4.6 Summary

In this chapter, we have introduced a phoneme-like unit segmentor that can segment a word (or an utterance) into smaller subword units based on its self-similarity matrix. The motivation is to divide each word into phonetically similar units so that we can have a more precise analysis of each subword region, without any requiring human labeling efforts. Evaluation was carried out on the task of phonetic boundary detection. Experimental results have shown that our approach can improve the performance on fully unsupervised phonetic boundary detection, while in general, it has a tendency to over-generate segments. However, as shown in Fig. 4-6, by mapping the boundaries onto the SSM, we can see that each segment does have low self-distance. Therefore, our goal of segmenting each word into phonetically similar units is achieved.



(a) spectrogram and the reference boundaries (both blue and green)

(b) segmentation with known number of phones in the utterance and without prior

(c) segmentation with unknown number of phones in the utterance and without prior

(d) MFCC-based self-similarity matrix

Figure 4-6: *The script: "she had your dark suit in greasy wash water all year". The green lines in (a) are those deleted in both (b) and (c). The black lines in (b) and (c) represent correctly detected boundaries, while the red dotted lines represent insertions.*

Chapter 5

Mispronunciation Detection

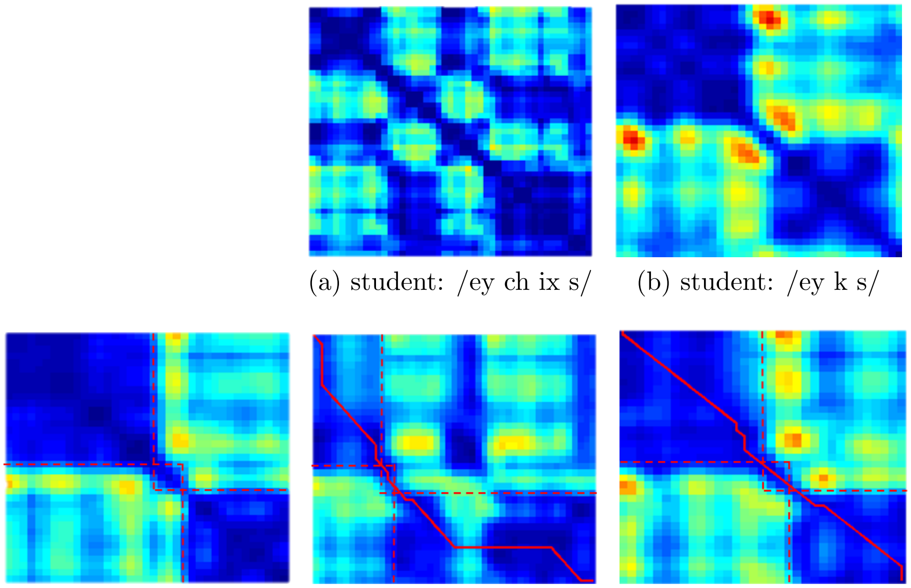
Fig. 5-1 shows the examples of a teacher saying “aches”, one student mispronouncing as */ey ch ix s/* and the other one pronouncing correctly. The distance matrices are shown in Fig. 5-1d and 5-1e, and Fig. 5-1a, 5-1b and 5-1c are the SSMs of the teacher and the two students, respectively. We can see that there is indeed a high-distance region between the teacher's */k/* and the student's */ch ix/*. This high-distance region also causes the warping path to be not in quasi-diagonal. Also, there is an obvious difference between the SSMs. On the basis of theses observations, in this chapter, we describe the design of phone-level and word-level features in detail.

5.1 Feature Extraction

5.1.1 Phone-Level Features

To divide each word into smaller units for analysis, we adopt the phoneme-like unit segmentor described in the previous chapter on reference utterance. The dotted lines in Fig. 5-1c are the resulting boundaries for example. After that, we use the aligned path, together with the unit boundaries, to segment the distance matrix into several blocks (see the regions bounded by dotted lines in Fig. 5-1d, 5-1e). Then, the phone-level features can be extracted within each block.

Though the segmentor we use does not guarantee to segment the reference word into


(a) student: /ey ch ix s/ (b) student: /ey k s/

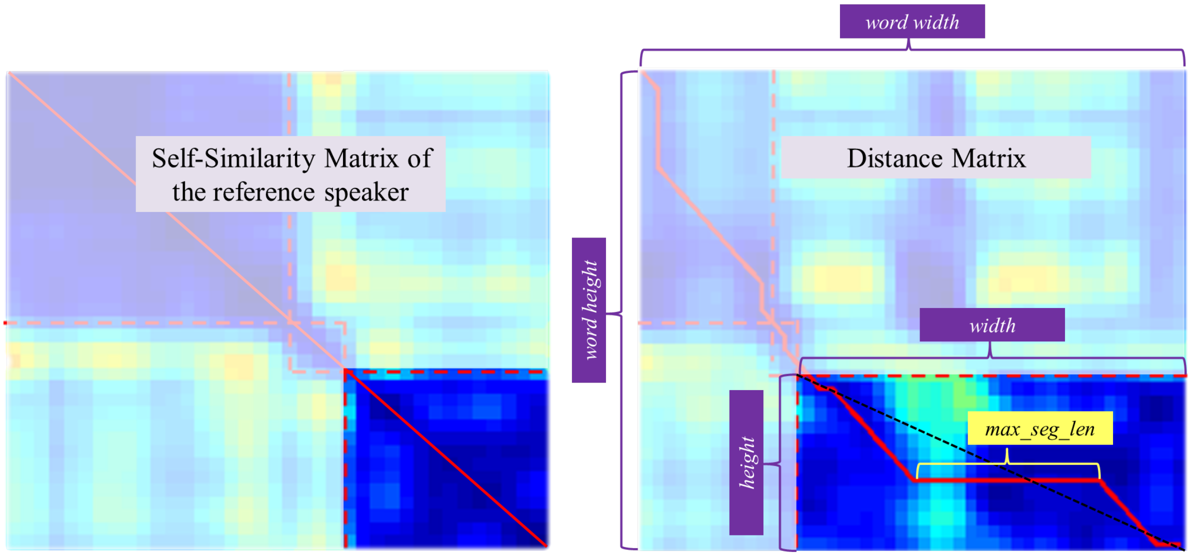
(c) teacher: /ey k s/ (d) (e)

Figure 5-1: *(a) and (b) are the self-aligned matrices of two students saying "aches", (c) is a teacher saying "aches", (d) shows the alignment between student (a) and the teacher, (e) shows the alignment between student (b) and the teacher. The dotted lines in (c) are the boundaries detected by the unsupervised phoneme-unit segmentor, and those in (d) and (e) are the segmentation based on the detected boundaries and the aligned path, i. e. the intersection between the warping path and the self-similarity boundaries of the teacher's utterance*

phones but into acoustically similar frames, it still suffices our needs. As we are looking for mismatches within a sub-region, if a segment in the teacher's utterance corresponds to two consecutive phones that are acoustically similar in GP representation, the corresponding segment from the student's utterance should also be acoustically similar not only within itself but also with that of the teacher's, if the student is doing well.

Another thing worth mentioning is that, if the student made a mistake, each segment in the student's utterance will not necessarily be the same phone as the corresponding segment in the teacher's. Take Fig. 5-1d for example. The distance between the */k/* from the teacher and the */ch ix/* from the student is high, so the aligned path would not choose to go across that region, resulting in the */s/* from the teacher mapping to the second part of */ch/* and */ix s/* from the student. On the other hand, each phone in the student's utterance matches perfectly to the teacher's in Fig. 5-1e.

Several kinds of features have been designed based on the assumption that within each block, if the aligned path is off-diagonal, or the average distance is high, there is a higher probability that the word is mispronounced. We can divide the features into the following categories. Fig. 5-2 illustrates the terms used in computing the features.



MEAN_REF_SCORE	the average distance within the highlighted region
AVG_PATH_REF_SCORE	the average distance along the aligned path (red line)

path_len	the length of the aligned path (red line)
path_score	the accumulated distance along the aligned path (red line)
diag_len	the length of the diaognal (black dotted line)
diag_score	the length of the diaognal (black dotted line)
rel_height	*height / word height*
rel_width	*width / word width*
MEAN_SCORE	the average distance within the highlighted region

Figure 5-2: *An illustration of phone-level features*

Aligned Path

Ideally, a good alignment should result in a path along the diagonal of each region. If the student mispronounced a word, no matter whether it is an insertion, deletion, substitution or a mix of the three, there must be some high-distance regions along the diagonal of the distance matrix, and thus the aligned path would be off-diagonal. If a path is off-diagonal, there will be some vertical or horizontal segments within it. Therefore, we can treat the longest vertical/horizontal segments of the aligned path as a feature describing the shape of the path. However, an off-diagonal path does not necessarily mean a mispronunciation. It could be the case where most of the elements in a region from a distance matrix have very low values, i. e. two segments are well matched, but there are some slightly higher values around the diagonal, so the path is warped to the off-diagonal. Therefore, we should further look at the average distance along the diagonal, as well as the average distance along the aligned path, and compare the two of them. As a result, there are five features in this category:

- RATIO_MAX_SEG: the ratio of the length of the longest vertical/horizontal segment to the length of the aligned path ($max\_seg\_len/path\_len$)
- AVG_PATH_SCORE: the average distance along the aligned path ($path\_score/path\_len$)
- AVG_DIAG_SCORE: the average distance along the diagonal ($diag\_score/diag\_len$)
- DIFF_AVG_DIAG_PATH_SCORE: the difference between the above two ($AVG\_DIAG\_SCORE - AVG\_PATH\_SCORE$)
- RATIO_AVG_DIAG_PATH_SCORE: the ratio between the two ($AVG\_DIAG\_SCORE/AVG\_PATH\_SCORE$)

Distance Matrix

Besides the aligned path, the high-distance regions in the distance matrix are also good indications to whether there is mispronunciation. We extract one feature based on it:

- MEAN_SCORE: the average distance across the region

Duration

The time duration is also an important aspect of pronunciation. Not only the absolute time duration of a phone but also its relative duration with respect to the word are crucial. We design three features to compare the absolute/relative time duration of a phoneme-like unit in the teacher's utterance and the corresponding one in the student's utterance.

- RATIO_ABS_DUR: the ratio between the absolute time duration of the segment in S and the absolute time duration of the segment in T, or the ratio of the opposite, whichever is larger ($\max(width/heigh, height/width)$)

- DIFF_REL_DUR: the absolute difference between the relative time duration of the segment with respect to the whole word in S and the relative time duration of the segment with respect to the whole word in T ($|rel\_width - rel\_height|$)
- RATIO_REL_DUR: the ratio of the relative time duration of the two segments ($\max(rel\_width/rel\_height, rel\_height/rel\_width)$)

Comparison with the Reference

Sometimes, a distance matrix may look fuzzy and it is hard to tell whether it links to mispronunciation or not. At these times, it is better to have some examples of good pronunciation versus bad ones. Unfortunately, when processing each pair of a teacher and a student, what we have at hand are only the two utterances. Still, we can regard the teacher as a "good student", look at the SSM of the teacher to have a sense of a good alignment, and compare the SSM with the distance matrix from the student. On the basis of this idea, we develop three features:

- DIFF_MEAN_REF_SCORE: the difference between the average distance of the region from the distance matrix and of the region from the SSM of the reference word ($MEAN\_SCORE - MEAN\_REF\_SCORE$)
- DIFF_AVG_PATH_REF_SCORE: the difference between the average distance along the aligned path in the distance matrix and the average distance along the diagonal of the SSM. Note that the best alignment in an SSM is the diagonal.
  ($AVG\_PATH\_SCORE - AVG\_PATH\_REF\_SCORE$)
- DIFF_AVG_DIAG_REF_SCORE: the difference between the average distance along the diagonal in the distance matrix and the average distance along the diagonal of the SSM ($AVG\_DIAG\_SCORE - AVG\_PATH\_REF\_SCORE$)

Others

For the last phone-level feature, we take an overall view of the segmented phoneme-like unit. We first average the speech features that represent the frames within a unit, say

$\bar{f}_{t_i} = \frac{1}{b^*_{i+1}-b^*_i}\sum_{j=b^*_i}^{b^*_{i+1}-1} f_{t_j}$ for the teacher, and also $\bar{f}_{s_i}$ for the student, and treat $\bar{f}_{t_i}$ and $\bar{f}_{s_i}$ as two single vectors representing the unit in the teacher's utterance and the student's, respectively. Then, we take $D(\bar{f}_{t_i}, \bar{f}_{s_i})$ as another phone-level feature.

Phone-level Features Summary

For all of the features described above, larger values indicate worse alignment. We pick the maximum value among all segments for each category to form the final feature vector. The idea is that if there is a phone being mispronounced, the whole word is mispronounced, and thus we can take the most badly aligned phone (and therefore probably the most badly pronounced phone) in the word to represent the whole word. Experimental results have shown that doing so gives much better results than averaging all features across all phones within the word, since this would average out the misbehavior of a specific phone. In the end, we have a 13-dim phone-level feature for each word.

5.1.2 Word-Level Features

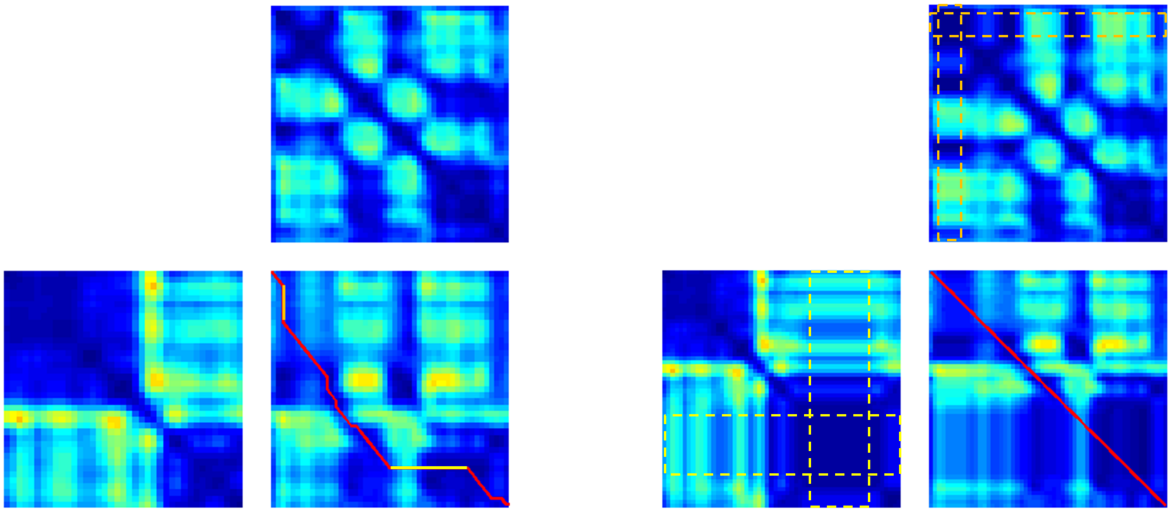
From Fig. 5-1a-5-1c, we can see that a mispronounced version (Fig. 5-1a, with one substitution and one insertion errors) results in a different appearance of the SSM. Note that this isn't true all the time. For example, if the student mispronounced the word as $/ow\ k\ s/$, the SSM would still look similar to the teacher's as there are also three phonetic units. But in this case, the distance matrix would not have three low-distance blocks along the diagonal.

On the basis of these observations, we can also detect mispronunciation from the word-level by comparing the two SSMs from the teacher and the student saying the same word, or the SSM from the teacher and the distance matrix. The features can be categorized as the following.

Aligned Path

Just as the phone-level features related to the aligned path, we can also extract features on the word-level in the same manner. Though the diagonal of the distance matrix of a word does not necessarily represent the most precise alignment between phones due to the difference in their time duration, it still serves as a good target to compare with. There are four features under this category:

- AVG_WORD_PATH_SCORE: the average distance along the aligned path
- AVG_WORD_DIAG_SCORE: the average distance along the diagonal
- DIFF_AVG_WORD_DIAG_PATH_SCORE: the difference between the above two $(AVG\_WORD\_DIAG\_SCORE - AVG\_WORD\_PATH\_SCORE)$
- RATIO_AVG_WORD_DIAG_PATH_SCORE: the ratio between the two $(AVG\_WORD\_DIAG\_SCORE / AVG\_WORD\_PATH\_SCORE)$



(a) before rewarping (top: SSM of the student, bottom-left: SSM of the teacher, bottom-right: distance matrix)

(b) after rewarping (top: SSM of the student, bottom-left: SSM of the teacher, bottom-right: distance matrix)

Figure 5-3: *An example of how the SSMs and the distance matrix would change after being rewarped. A vertical segment would expand the student's utterance, and a horizontal one would expand the teacher's. The orange boxes in (b) shows the resulting expansion from the orange segment in (a), and the yellow boxes in (b) shows the resulting expansion from the yellow segment in (a).*

Element-wise Comparison

The most direct way to compare two images is to compute their pixel-wise difference. To compute the element-wise difference between two matrices, the first step is to make sure they are of the same size. Before, T is of length n and S is of length m. The aligned path $\boldsymbol{\psi}_{ts}^*$ is a sequence of l index pairs, $((\psi_{11}^*, \psi_{12}^*), (\psi_{21}^*, \psi_{22}^*), \ldots, (\psi_{l1}^*, \psi_{l2}^*))$. T and S can be rewarped into two sequences of the same length, l, according to $\boldsymbol{\psi}_{ts}^*$:

$$T^w = (f_{t_{\psi_{11}^*}}, f_{t_{\psi_{21}^*}}, \ldots, f_{t_{\psi_{l1}^*}}),\ S^w = (f_{s_{\psi_{12}^*}}, f_{s_{\psi_{22}^*}}, \ldots, f_{s_{\psi_{l2}^*}}). \tag{5.1}$$

Then Φ_{ts}^w can be computed from T^w and S^w, Φ_{tt}^w from the self-alignment of T^w, and Φ_{ss}^w from the self-alignment of S^w. All of the above three matrices are of size $l \times l$, and thus four features can be extracted:

- DIFF_SSM_T_S: absolute element-wise difference between the self-aligned matrices Φ_{tt}^w and Φ_{ss}^w, averaged by the total area $l \times l$
- DIFF_SSM_T_DISMAT: absolute element-wise difference between the self-aligned matrix Φ_{tt}^w and the rewarped distance matrix Φ_{ts}^w, averaged by the total area $l \times l$
- DIFF_BLOCK_T_S: absolute element-wise difference between the self-aligned matrices Φ_{tt}^w and Φ_{ss}^w averaged by the total area, only focusing on the regions along the diagonal, which are resulted from applying the phoneme-like unit segmentor on Φ_{tt}^w
- DIFF_BLOCK_T_DISMAT: absolute element-wise difference between Φ_{tt}^w and Φ_{ts}^w averaged by the total area, only focusing on the regions along the diagonal

Image Structure

Directly computing the element-wise difference between two matrices may not be robust enough. What we really care about is whether the underlying "structure" of the two matrices are similar. Muscariello et. al [25, 26] have proposed to view an SSM as an image and applied image processing techniques to extract the edge information from it. They have shown that this method helps in the task of spoken term detection. Similarly, we can also take advantage of this technique to compare the edge information of two SSMs of the same word.

The image processing techniques that has been used is called the local histograms of oriented gradients (local HOGs) [9], which is basically computing the distributions of local

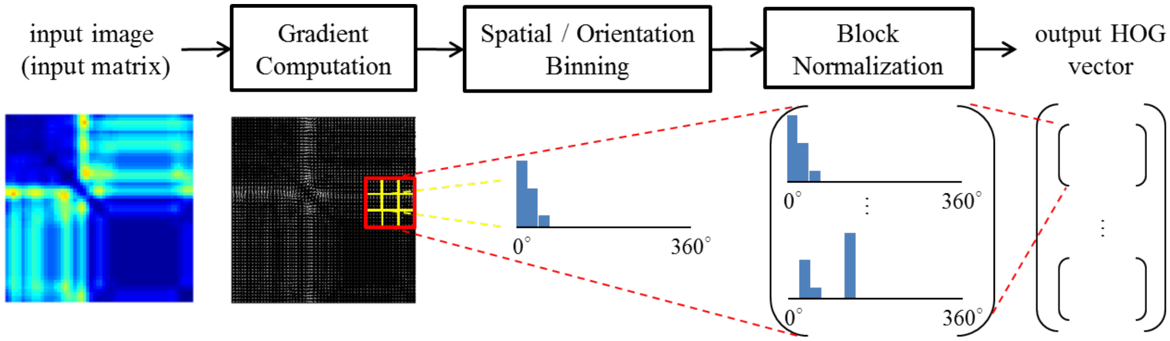


Figure 5-4: *The flowchart of extracting local histograms of gradients*

gradients and edge orientations. Fig. 5-4 is the flowchart of how to compute local HOGs. Here we explain this process step by step:

1. Gradient Computation:

   For each pixel, i. e. each element in the matrix, apply a simple 1-D mask $[-1, 0, 1]$ centered on it to compute the x-direction gradient. Apply $[1, 0, -1]^T$ to compute the y-direction gradient. By combing the above two, we can obtain the magnitude and the orientation of the gradient centered at a specific location. Fig. 5-5 shows an example of an input image (Fig. 5-5a), the pixel-wise gradient (Fig. 5-5b) and the pixel-wise gradient of only the diagonal regions (Fig. 5-5c). We can see that the gradients around the boundary between a low-distance region and a high-distance region will be large so that the gradient map can tell us something about the structure of the matrix.

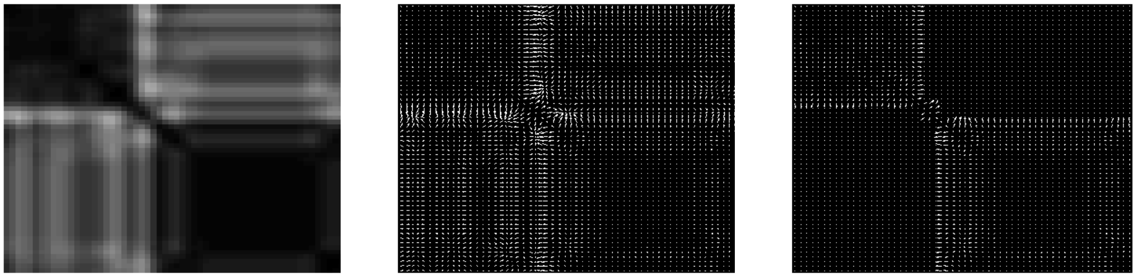
2. Spatial / Orientation Binning:

   A *cell* is a local spatial region consisting of several pixels. Each pixel within a *cell* can have a vote to the orientation bins that are evenly spaced over $0° - 360°$, weighted by the magnitude of its gradient. As a result, each *cell* has a histogram of orientations.

3. Block Normalization:

   A *block* is a group of *cells*. The normalization is carried out within each *block* by first concatenating the histograms of all cells within it to a vector $\mathbf{v}$. Then, L2-norm is used for normalization: $\mathbf{v} \to \mathbf{v}/\sqrt{\|\mathbf{v}\|_2^2 + \epsilon^2}$.

The final output is a vector which is the concatenation of all normalized histograms over all the *blocks*. We pick the size of a *cell* to be 3×3 pixels, and the size of a *block* to be



(a) input image: Φ_{tt}^{w}

(b) pixel-wise gradients

(c) pixel-wise gradients only in the diagonal regions (the gradients in the other regions are set to 0)

Figure 5-5: *An example of the results of gradient computation (the length of the arrows represents the magnitude, and the direction the arrows point to is the orientation)*

3 × 3 cells. The number of orientation bins is set to 18. There is an overlap of 2 *cells* in the x-direction or the y-direction of blocks. Again, we focus on the rewarped version of the matrices, $\Phi_{tt}^{w}, \Phi_{ss}^{w}$ and Φ_{ts}^{w}. Four kinds of features are extracted:

- DIFF_HOG_SSM_T_S: absolute difference between the local HOGs of Φ_{tt}^{w} and Φ_{ss}^{w}, averaged by the length of the HOGs vector

- DIFF_HOG_SSM_T_DISMAT: absolute difference between the local HOGs of Φ_{tt}^{w} and Φ_{ts}^{w}, averaged by the length of the HOGs vector

- DIFF_HOG_BLOCK_T_S: absolute difference between the local HOGs of the diagonal regions of Φ_{tt}^{w} and Φ_{ss}^{w}, averaged by the length of the HOGs vector

- DIFF_HOG_BLOCK_T_DISMAT: absolute difference between the local HOGs of the diagonal regions of Φ_{tt}^{w} and Φ_{ts}^{w}, averaged by the length of the HOGs vector

Acoustic Features

The final set of features are the raw acoustic features averaged across the word duration, as some words are easier to be mispronounced, and we encode this information through the raw acoustic features. In the end, we have a $16 + d$-dim word-level feature for each word, where d equals to the dimension of the acoustic features. Table 5.1 summarizes all the features we use in this work.

type	category	features
Phone-Level	Aligned Path	RATIO_MAX_SEG, AVG_PATH_SCORE, AVG_DIAG_SCORE, DIFF_AVG_PATH_DIAG_SCORE, RATIO_AVG_PATH_DIAG_SCORE
	Distance Matrix	MEAN_SCORE
	Duration	RATIO_ABS_DUR, DIFF_REL_DUR, RATIO_REL_DUR
	Comparison with the Reference	DIFF_MEAN_REF_SCORE, DIFF_AVG_PATH_REF_SCORE, DIFF_AVG_DIAG_REF_SCORE
	Others	$D(f_{t_i}, f_{s_i})$
Word-Level	Aligned Path	AVG_WORD_PATH_SCORE, AVG_WORD_DIAG_SCORE, DIFF_AVG_WORD_PATH_DIAG_SCORE, RATIO_AVG_WORD_PATH_DIAG_SCORE
	Element-wise Comparison	DIFF_SSM_T_S, DIFF_SSM_T_DISMAT, DIFF_BLOCK_T_S, DIFF_BLOCK_T_DISMAT
	Image Structure	DIFF_HOG_SSM_T_S, DIFF_HOG_SSM_T_DISMAT, DIFF_HOG_BLOCK_T_S, DIFF_HOG_BLOCK_T_DISMAT
	Acoustic Features	average raw acoustic features across the duration of the word

Table 5.1: *A summary of the features used in our mispronunciation detection framework*

5.2 Classification

Given the extracted features and a set of good or mispronounced labels, detecting mispronunciation can be treated as a classification task. We adopt libsvm [7] to implement SVM classifiers with an RBF kernel. From Table 2.1 we can see that the ratio between the positive training samples (mispronounced words) and the negative training samples (good words) is low. This causes the learning problem to be hard. To deal with the data-imbalance problem, we divide the negative training data into five, and combine each fifth with all the positive training samples to form five balanced training sets. Five classifiers can be trained, and during testing, their output probabilities are averaged to form the final output.

We treat the alignments of different teachers' utterances and the same student's utterance in the training set as different training samples. For the test set, if there are multiple matching reference utterances for a student's utterance, the following three different voting schemes are used to combine the output probability from the classifiers:

- random vote: randomly pick one reference speaker and consider its output only
- max-margin vote: pick the output whose absolute difference between the predicted probability for the two classes is the maximum
- average vote: average the output probability from all the reference speakers

5.3 Experiments

5.3.1 Experimental Setup and Evaluation Metrics

We follow the same experimental setup described in section 3.3.1: 39-dim MFCCs at every 10 ms and a 150-mixture GMM trained on all TIMIT data. The parameters of the SVM are optimized for different scenarios, respectively. *Precision*, *recall* and *f-score* are used for evaluation. They are defined as

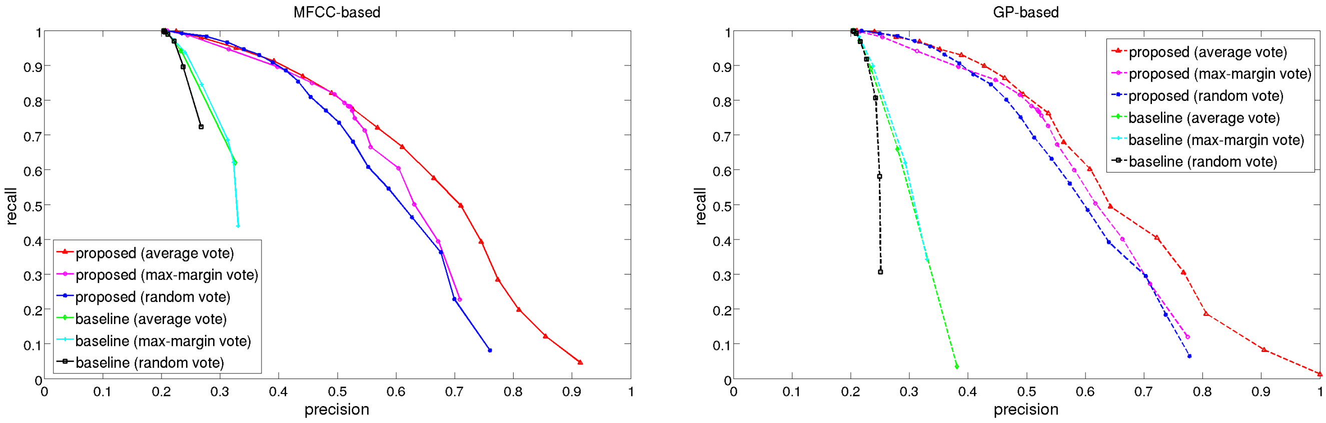
$$pr = \frac{TM}{TM+FM}, \; re = \frac{TM}{TM+FG}, \; f = \frac{2 \cdot pr \cdot re}{pr + re}, \tag{5.2}$$

where TM is the number of mispronunciations correctly identified by the classifier, FM is the number of hypothesized mispronunciations that were actually considered to be acceptable, and FG are those that are misclassified as good pronunciations.

5.3.2 Baseline

For baseline, we build a naive framework with only three features which are a subset of word-level features, AVG_WORD_PATH_SCORE, AVG_WORD_DIAG_SCORE and DIFF_AVG_WORD_PATH_DIAG_SCORE. In other words, the baseline considers word-level features related to the DTW score only. It does not involve any analysis on the shape of the aligned path or the pattern of the distance matrix, or the use of the unsupervised phoneme-like unit segmentor.

5.3.3 Performance Under Different Voting Schemes



(a) MFCC-based framework

(b) GP-based framework

Figure 5-6: *Performance under different voting schemes*

Fig. 5-6 shows the ROC curves for different voting schemes, either based on MFCC-based alignment (Fig. 5-6a) or GP-based alignment (Fig. 5-6b). First of all, for both MFCC-based and GP-based alignments, average voting scheme works the best, with max-margin the second and random voting the worst. For the baseline, average voting and max-margin voting have similar performance, while random voting is still the worst. This result is coherent with the common claim that comparison-based methods require a lot of reference data. However, since we do not need phone-level labelings, we believe this effort should still be less than that of building a recognizer. In the following sections, we will only consider the performance from average voting for different scenarios.

	MFCC (proposed)	GP (proposed)	MFCC (baseline)	GP (baseline)
f-score (%)	63.7	63.0	42.8	39.3

Table 5.2: *Best performance under each scenario*

Table 5.2 shows the best performance under each scenario. There are many factors affecting the overall performance, e.g. erroneous labels from AMT, and the confusions between phones in GMM. Nevertheless, our framework improves the baseline by at least 49% relatively. This shows that merely considering the distance along the aligned path is not enough. Extracting features based on the shape of the aligned path or the appearance of the distance matrix, or segmenting a word into subword units for more detailed analysis, can give us more information about the quality of the alignment, and thus the quality of the pronunciation.

5.3.4 Comparison Between Different Levels of Features

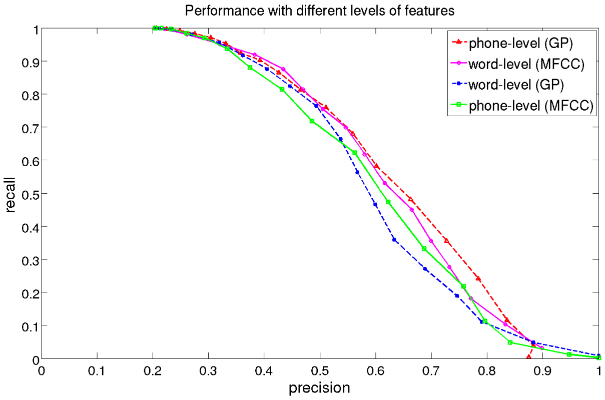


Figure 5-7: *Performance with different levels of features*

f-score (%)	MFCC-based	GP-based
phone-level	59.1	61.4
word-level	61.3	60.0
overall	63.7	63.0
baseline	42.8	39.3

Table 5.3: *Best f-scores from different levels of features*

Fig. 5-7 shows the ROC curves for either word-level or phone-level features only and with either MFCC-based DTW or GP-based DTW, and Table 5.3 compares the best f-scores from different cases. First of all, compared with the baseline, a system with word-level features only can achieve a relative increase around 45%. This again shows the benefits of having features

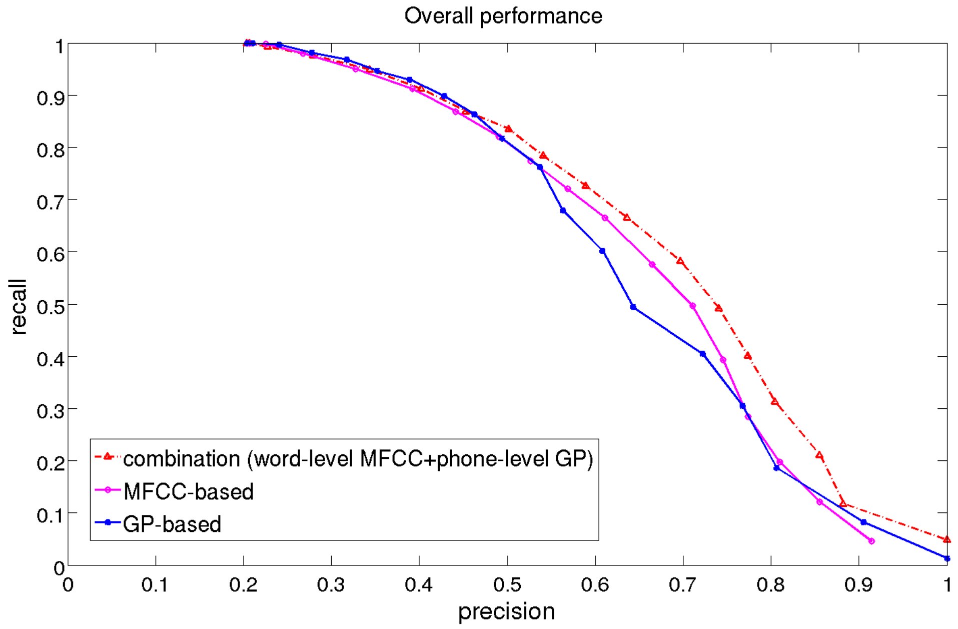


Figure 5-8: *The overall performance can be improved by combining different levels of features from different speech representation*

that compare the structure of the distance matrices. A system with phone-level features only improves the performance by 47% relative to the baseline. This also proves that an analysis which is beyond word-level can exhibit more details about the alignment. Finally, we can see that combining the features from different levels did help improve the performance. This indicates that the features from the two levels have complimentary information to one another.

One thing worth noticing is that on the word level, an MFCC-based framework performs better than a GP-based framework, while it is the opposite on the phone level. This is interesting since it seems that the phoneme-like unit segmentor performs better in terms of detecting phone boundaries when using MFCC-based SSMs (recall the results from Section 4.5). Our experimental results may suggest that GP-based aligned paths preserve more details, while the MFCC-based distance matrices preserve more structural information.

Inspired by the observation that MFCC-based alignment and the GP-based one have different strength on the phone-level and the word-level, we further try to combine word-level and MFCC-based features with phone-level and GP-based features to see how the performance would be. Fig. 5-8 shows an encouraging result, where the overall performance is further improved to an f-score of 65.1% when combining the two. This result implies that not only the word-level and the phone-level features have complimentary information, but also the MFCC-based and the GP-based features.

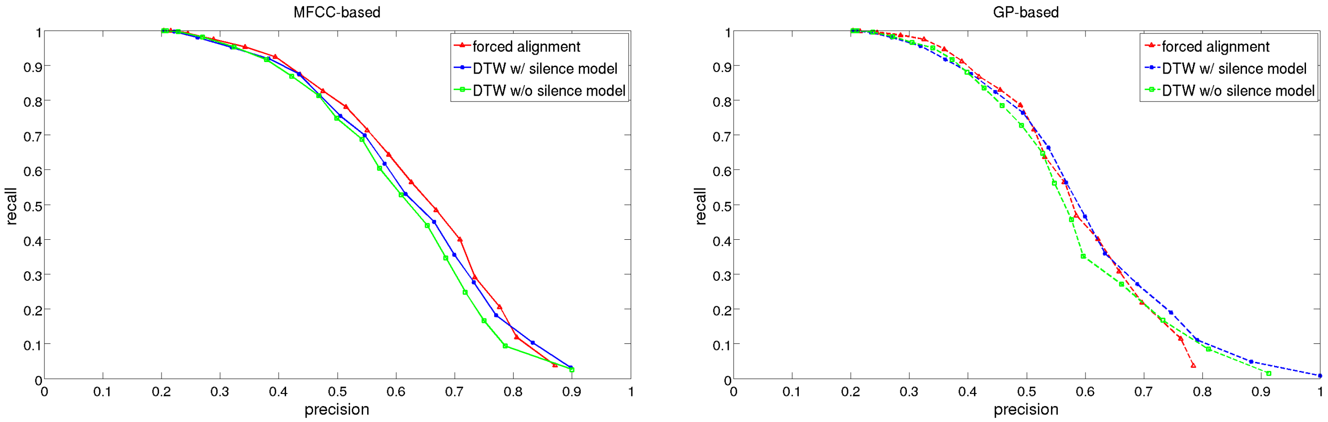
5.3.5 Performance with Different Amount of Information

In this sub-section, we examine how different amount of information can affect the performance of our framework. Note that in our original framework, we only need the information of word-level timing on the teacher's utterance. In the following, we further add information, such as the word-level timing on the student's utterance and the phone-level timing on the teacher's utterance, to see how the performance will change.

The Effect of Word Segmentation

First we focus on the word-level features. Before, the word-level timing on the nonnative utterance is obtained from running DTW with a silence model. Here we consider two more cases: the first one incorporates the word-level timing information from the forced-alignment results from a recognizer (i. e. with more accurate word boundaries), and the second one runs DTW without a silence model (i. e. with less accurate word boundaries).

Fig. 5-9 demonstrates the word-level performance with different amount of information. Not surprisingly, without a silence model, the word segmentation becomes less precise and



(a) MFCC-based framework

(b) GP-based framework

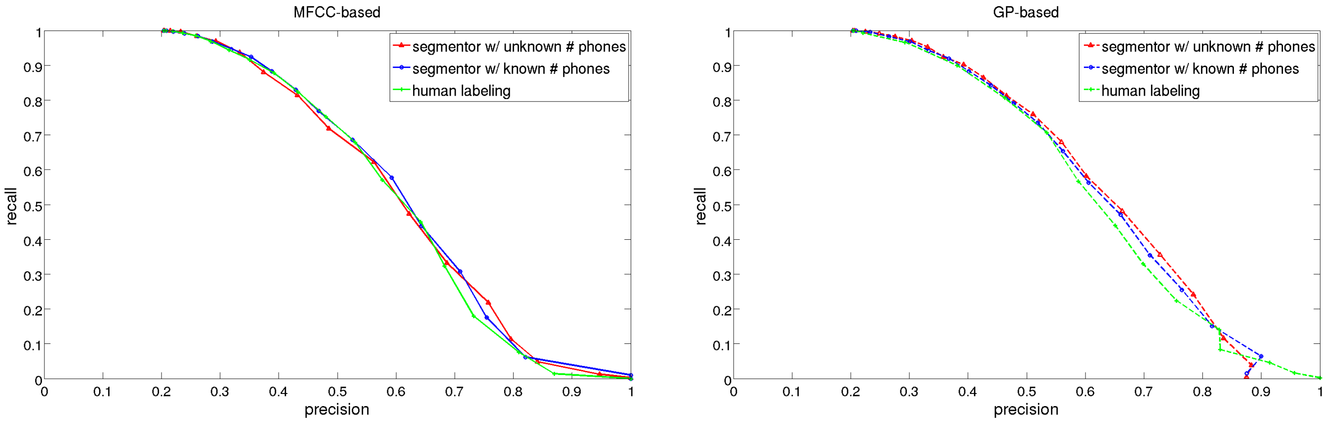
Figure 5-9: *Performance with different amount of information (word-level features only)*

the resulting word segments may include a lot of silence, causing the system performance to drop. However, the improvement benefited from more precise word boundaries is not very large. For an MFCC-based framework (Fig. 5-9a), there is only a 0.9% absolute increase in the best f-score from our original framework to using the forced alignment, and there is even a 1.1% absolute decrease for a GP-based framework (Fig. 5-9b). These results imply that the trade-off between the system performance and the required amount of knowledge on the data is not much, which is good for a system that aims at lowering the need of human efforts on data preparation.

The Effect of the Unsupervised Phoneme-like Unit Segmentor

For the phone level analysis, our framework makes use of a phoneme-like unit segmentor which incorporates a prior on segment length and does not assume the number of phones in a word to be known beforehand. Here we consider two other cases: 1) run a phoneme-like unit segmentor that knows the number of phones beforehand (i. e. partial knowledge), and 2) obtain the phone-level timing information on the teacher's utterance from human labeling (i. e. full knowledge).

The resulting ROC curves are shown in Fig. 5-10. For an MFCC-based framework (Fig. 5-10a), there is no obvious order of which case performs the best and which one the worst, while having full knowledge of the phone timing decreases the performance of a GP-based system



(a) MFCC-based framework

(b) GP-based framework

Figure 5-10: *Performance with different amount of information (phone-level features only)*

(Fig. 5-10b). This may suggest that for phone-level analysis, an exact segmentation of the word into phonemes is not the most important issue, as long as each segment is acoustically similar within itself. Even though a segment may not correspond to exactly one phone, it can be viewed as a unit that should be aligned equally well to a segment in a student's utterance.

In fact, since the segmentor finds the unit boundaries by optimizing the criterion in Eq. 4.3, and it can always obtain the global optimum, if the resulting boundaries are not the same as those from human labelings, this implies that the regions bounded by human labeling do not have the lowest sum of distance, and thus some regions may not be phonetically similar within itself in terms of the speech representation we choose. As the subsequent feature extraction is still based on distance matrices and SSMs of the same speech representation, the matching between the two explains the reason why phone-level human labelings do not really help the system.

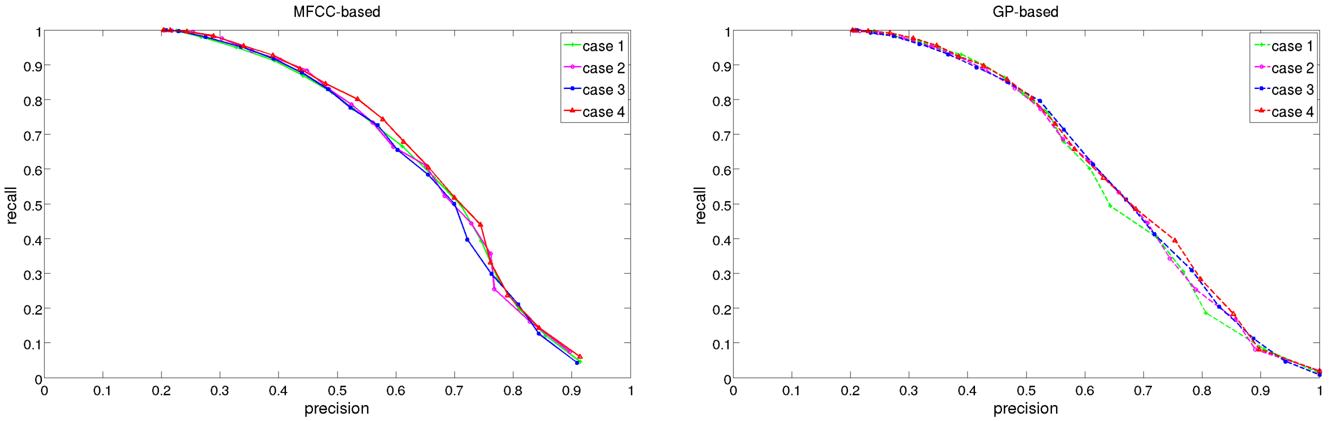
Overall Performance

To examine how the overall performance would change with different amount of information, four scenarios are compared, as shown in Table 5.4. The order of the amount of information is case 4 > case 3 > case 2 > case 1, and case 1 is our original framework. Fig. 5-11 shows the experimental results.

		phone	
		unsupervised segmentor w/ unknown number of phones	timing information obtained from human labeling
word	DTW w/ silence model	case 1	case 3
	forced alignment	case 2	case 4

Table 5.4: *Four scenarios that were implemented for overall performance comparison*

According to Fig. 5-11, case 4 performs the best in an MFCC-based framework (Fig. 5-11a), but the margin is really small. Similarly, case 3 and case 4 perform slightly better in a GP-based framework (Fig. 5-11b), but there isn't really a significant difference. Therefore, we can conclude that for the overall performance of the system, our current framework is comparable with the one that has complete detailed timing information on the data. As a result, in the future, we can focus on designing more sophisticated features and finding a



(a) MFCC-based framework

(b) GP-based framework

Figure 5-11: *Performance with different amount of information (both word-level and phone-level features). See Table 5.4 for the description of each case.*

better representation of speech without worrying too much about the segmentation of the utterances.

5.3.6 Performance based on Same/Different-gender Alignments

In the last part of the experiments, we examine how alignments between a teacher and a student of different genders would affect the system performance. Again, we use a subset of 1,000 utterances in the test set that have reference speakers from both genders. The training data is still the same, i. e. with same-gender alignment only. The goal is to see whether the alignments based on different-gender pairs are capturing the same information as the alignments based on same-gender pairs. If the gender does not affect the alignment but only the pronunciation does, the performance on detecting mispronunciation should be similar.

Table 5.5 summarizes the experimental results. The performance with different-gender

	precision (%)	recall (%)	f-score (%)
same-gender (MFCC)	60.9	67.3	63.9
same-gender (GP)	57.3	70.2	63.1
different-gender (MFCC)	21.1	89.6	34.2
different-gender (GP)	21.0	93.8	34.3

Table 5.5: *Performance on detecting mispronunciation based on same/different-gender alignment*

alignments drops about 47% relative to the performance based on same-gender alignments. The low precisions means that the system misclassified a lot of words as bad pronunciations but they are actually correct ones. This indicates that the aligned path between a teacher and a student of the different genders might be off-diagonal or with high distance so that it is viewed as a bad alignment when compared with an alignment based on a pair of same-gender speakers. We can thus infer from the results that the alignments based on different genders are not only affected by the phone identities but also the acoustic characteristics of different genders. As we learn from this table that these two kinds of alignments contain different information in them, we should only consider same-gender alignments when detecting mispronunciation.

5.4 Summary

In this chapter, we have introduced the mispronunciation detection stage of our framework. The basic flow of this stage is to first segment each word into smaller subword units, extract phone-level and word-level features and adopt SVM classifiers to predict whether the word is mispronounced or not.

We have introduced a set of specially designed phone-level and word-level features. These features focus on describing the shape of the aligned path and the structure of the distance matrix. Through a series of experiments, we have shown that the phone-level and word-level features can capture different aspects of pronunciation errors, and combining the two can give the best performance.

Also, we have examined how the system performance would be if we have more information on the data. Experimental results indicate that our system performance is relatively stable with respect to the decrease of the amount of information that is being used.

Chapter 6

Summary and Future Work

In this chapter, we summarize the thesis and suggest possible future research directions.

6.1 Summary and Contributions

In this thesis, we have presented a mispronunciation detection framework that works by analyzing the alignment between a student's utterance and a teacher's. The motivation is to avoid the problem of requiring well-labeled training data needed for conventional speech recognizer-based methods. The whole framework can be divided into three components – word segmentation, phoneme-like unit segmentation, and mispronunciation detection. Within each block, we propose several ideas to gradually tackle the challenges of not having enough information from human labelings. In the following, we summarize our main contributions.

In Chapter 3, we present the first stage of our framework – word segmentation. A student's utterance will not contain timing information on it. With the assumption that the student is trying to enunciate the given script, we carry out DTW to align the student's utterance with a reference utterance. Then, the word boundaries can be located by finding the intersection between the boundaries in the teacher's utterance and the aligned path. To deal with pauses which commonly appear between words in nonnative speech, we propose to run DTW with a silence model, which can simultaneously align the two utterances and detect silence in the student's utterance. Experimental results have shown the effectiveness of the silence model.

In Chapter 4, an unsupervised phoneme-like unit segmentor is proposed. The motivation is to segment each word into smaller phonetically similar units for analysis at a finer level. The segmentor works by dividing the self-similarity matrix of an utterance into a number of low-distance blocks along the diagonal. A dynamic programming solution to optimizing the segmented boundaries is presented. Though the goal is not segmenting a word into a single phoneme per acoustic unit, experimental results show that the proposed segmentor can achieve satisfying performance on the task of phonetic boundary detection.

In Chapter 5, we introduce a set of features that capture the shape of the aligned path and the structure of the distance matrix. SVM classifiers were used for the task of classifying whether a word is mispronounced or not. Experimental results have demonstrated that word-level and phone-level features capture complementary characteristics of mispronunciation, and the system performs equally well even if without phonetic unit labelings. Therefore, in the end, we only need word-level timing information on the native speech to build this system. In fact, if we have multiple native recordings of the same content, we only need word-level timing labels on one of them, and we can use that information to label the rest of the native utterances. As a result, the effort to prepare the teaching materials is much less than the effort to build a speech recognizer.

6.2 Future Work

On the basis of the framework presented in this thesis, there are many potential applications that can extend this work. Below we suggest some possible directions.

6.2.1 A More Complete Framework

The current system framework is an initial attempt at detecting pronunciation errors, including insertion, deletion and substitution errors. However, one common error that students often make is having a wrong lexical stress pattern. For now we cannot detect this kind of error through MFCC-based or GP-based alignment, since those features were not designed for extracting pitch information from speech. However, we believe the alignment technique can also be used for aligning two pitch sequences. Stress can be further divided into word-level, or sentence-level stress. Therefore, the alignment can be carried out either between two whole utterances, or between word segments that were previously found from MFCC-based or GP-based alignment.

Lastly, to further aid student learning, a good CAPT system should not only point out that they made errors, but also show what kind of errors were made. This would require the system to be able to distinguish between different types of errors. To achieve such a goal, one possible solution might be to train different detectors for different types of pronunciation errors.

6.2.2 A More Accurate Representation of Pronunciation

In our experiments, we use the Gaussian posteriorgram in hopes of representing speech in a more compact way. It has the advantage that it can be obtained completely in an unsupervised fashion. However, the performance of a GP-based framework compared to an MFCC-based framework is not as good as how a GP-based method performs in other unsupervised pattern matching tasks, such as spoken term detection. Though an MFCC-based framework performs slightly better in our experiments, MFCC still suffers from being sensitive to vocal tract differences.

Recent research results have shown that Deep Belief Learning techniques have great potential in improving speech-related applications from speech recognition to unsupervised speech pattern discovery. With only a small amount of labels, one can obtain results that are comparable to those from supervised training [39]. This characteristic satisfies our goal of requiring as little knowledge or annotation effort from humans as possible when building a CAPT system.

Moreover, unsupervised acoustic modeling methods are also improving. Another way to add some supervision in an unsupervised manner may be to first perform unsupervised segmentation. Then, assign each cluster a label ,and use those labels to train models and then decode posteriorgrams.

6.2.3 Application to Other Languages

Within our framework, we did not require any linguistic knowledge of the target language (English) and the students' L1 (Cantonese). In the entire process, from MFCC extraction, GP decoding, DTW alignment, to feature extraction and classifier prediction, we did not include any kind of information about the "language" itself. What the system does is simply look for differences between a teacher's utterance and a student's utterance. Thus, this comparison-based framework offers a considerable advantage in its applicability to any languages, not only for the case in which the target language differs, but also in situations where the L1s of the students are different. In the future, it would be interesting to actually carry out experiments on other languages to see how the performance holds.

6.2.4 Implementation of a Web-based CALL System

The Spoken Language Systems group at MIT has launched many web-based language learning games, ranging from reading games, translation games, or dialogues. Among these systems, one common theme is that they are all recognizer-based, and do not focus on correcting pronunciation errors.

A natural next step might be to integrate pronunciation evaluation into these existing language learning games. In the future, it would also be worthwhile to provide a platform for users to upload audio or video content of native speech that are of personal interest, empowering learners to take advantage of our framework to create their own language learning systems.

Appendix A

Implementation of the Unsupervised Phoneme-like Unit Segmentor

In this appendix, we show how to formulate the unsupervised phoneme-like unit segmentor described in Chapter 4 into a dynamic programming problem and present the implementation.

First we start with the implementation of the segmentor with known number of segments. Recall Eq. 4.1. This whole process can be formulated into a dynamic programming problem in a sense that, if there are K segments in this sequence, and we fix the first cut at b_1^*, then $(b_2^*, b_3^*, \ldots, b_{K-1}^*)$ should also be the optimal solution from segmenting the sub-SSM starting from b_1^* to the end. Bearing this in mind, we start the implementation by building a $scoreMat$ in which element (i, j) records the cost of having a segment starting at i and ending at j. Recall that $cost(i, j) = \frac{1}{j-i+1} \sum_{y=i}^{j} \sum_{x=i}^{y} \Phi_{tt}(y, x)$. The $scoreMat$ can be computed as shown in Function 1.

Then, we build a $costMat$ and an $idxMat$. The $(i, j)^{th}$ element in $costMat$ records the minimum sum of cost of segmenting the sub-SSM starting from j to the end into i segments, and the $(i, j)^{th}$ element in $idxMat$ keeps track of the place of the first cut that leads to the minimum sum of cost. These two matrices can be built in a row-by-row manner. In other words, to compute the $i + 1^{th}$ row in the two matrices, we can adopt the results from the i^{th} row. This is because when computing $costMat(i + 1, j)$, given any location where the first cut is, say x, we know its total cost to be $cost(j, x - 1) + costMat(i, x)$. Therefore, what we need to do is to find the best x and record it in $idxMat(i + 1, j)$, and the corresponding cost in $costMat(i + 1, j)$. Assume there should be K segments. In the end, we only have to look up $idxMat(K, 1)$ and backtrace from it to find all the cut boundaries. Function 2 describes how we compute them, and Function 3 describes the backtracing process.

To generalize the above computation to unknown number of segments, we can simply run Function 2 with input K equal to $nframe$, which is the maximum possible number of cuts on the SSM. After running Function 2, we can determine K_u^*, the optimal number of cuts, by searching for the minimum cost in the first column of $costMat$. To incorporate a prior, we can add $\beta \times prior(j - i + 1)$ to $scoreMat(i, j)$ in Function 1.

Function 1 Computing $scoreMat$

```
function computeZeroCutScore($ssm, nframe$)    ▷ ssm: the target SSM, nframe: the length of the input utterance
    for $i = 1 \to nframe$ do
        for $j = i \to nframe$ do
            $scoreMat(i, j) \leftarrow 0$
            for $y = i \to j$ do
                for $x = i \to y$ do
                    $scoreMat(i, j) \leftarrow scoreMat(i, j) + ssm(y, x)/(j - i + 1)$
                end for
            end for
        end for
    end for
    return $scoreMat$
end function
```

Function 2 Computing *costMat* and *idxMat*

```
function findBestCut($scoreMat, nframe, K$)    ▷ K: the total number of segments
    for $i = 1 \rightarrow nframe$ do    ▷ base case: 1 segment
        $costMat(1, i) \leftarrow scoreMat(i, nframe)$
        $idxMat(1, i) \leftarrow i$
    end for
    for $x = 2 \rightarrow K$ do
        for $i = 1 \rightarrow nframe - x$ do
            $minScore \leftarrow scoreMat(i, i) + costMat(x - 1, i + 1)$
            $minIdx \leftarrow i + 1$
            for  $j = i + 1 \rightarrow nframe - x$  do
                $tempScore \leftarrow scoreMat(i, j) + costMat(x - 1, j + 1)$
                if $tempScore < minScore$ then
                    $minScore \leftarrow tempScore$
                    $minIdx \leftarrow j + 1$
                end if
            end for
            $costMat(x, i) \leftarrow minScore$
            $idxMat(x, i) \leftarrow minIdx$
        end for
    end for
    return $costMat, idxMat$
end function
```

Function 3 Backtracing

```
function backTrace($idxMat, K$)
    $x \leftarrow K$
    $bid \leftarrow 1$
    while $x > 0$ do
        $bid \leftarrow idxMat(x, bid)$
        $x \leftarrow x - 1$
        $boundary.push\_back(bid)$
    end while
    return $boundary$
end function
```

Bibliography

[1] Ethnologue: Languages of the world. http://www.ethnologue.com/.

[2] http://www.nuance.com/for-business/by-solution/customer-service-solutions/solutions-services/inbound-solutions/self-service-automation/recognizer/recognizer-languages/index.htm.

[3] J. Anderson-Hsieh, R. Johnson, and K. Koehler. The relationship between native speaker judgments of nonnative pronunciation and deviance in segmentais, prosody, and syllable structure. *Language learning*, 42(4):529–555, 1992.

[4] G. Aradilla, J. Vepa, and H. Bourlard. Using posterior-based features in template matching for speech recognition. In *Ninth International Conference on Spoken Language Processing*, 2006.

[5] L.F. Bachman. *Fundamental considerations in language testing*. Oxford University Press, USA, 1990.

[6] J. Bridle and N. Sedgwick. A method for segmenting acoustic patterns, with applications to automatic speech recognition. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'77.*, volume 2, pages 656–659. IEEE, 1977.

[7] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[8] J.R. Cohen. Segmenting speech using dynamic programming. *The Journal of the Acoustical Society of America*, 69(5):1430–1438, 1981.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. Ieee, 2005.

[10] J. Dalby and D. Kewley-Port. Explicit pronunciation training using automatic speech recognition technology. *CALICO journal*, 16(3):425–445, 1999.

[11] Sorin Dusan and Lawrence Rabiner. On the relation between maximum spectral transition positions and phone boundaries. In *Proc. Interspeech*, pages 645 –648, 2006.

[12] M. Eskenazi. Detection of foreign speakers' pronunciation errors for second language training-preliminary results. In *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, volume 3, pages 1465–1468. IEEE, 1996.

[13] Maxine Eskenazi. An overview of spoken language technology for education. *Speech Communication*, 51(10):832 – 844, 2009.

[14] Yago Pereiro Estevan, Vincent Wan, and Odette Scharenborg. Finding maximum margin segments in speech. In *Proc. ICASSP*, volume 4, pages IV–937 –IV–940, april 2007.

[15] H. Franco, L. Neumeyer, M. Ramos, and H. Bratt. Automatic detection of phone-level mispronunciation for language learning. In *Sixth European Conference on Speech Communication and Technology*, 1999.

[16] J.S. Garofolo, L.F. Lamel, W.M. Fisher, J.G. Fiscus, D.S. Pallett, N.L. Dahlgren, and V. Zue. Timit acoustic-phonetic continuous speech corpus. *Linguistic Data Consortium*, 10(5):0, 1993.

[17] J.R. Glass and V.W. Zue. Multi-level acoustic segmentation of continuous speech. In *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on*, pages 429–432. IEEE, 1988.

[18] A.M. Harrison, W.Y. Lau, H.M. Meng, and L. Wang. Improving mispronunciation detection and diagnosis of learners' speech with context-sensitive phonological rules based on language transfer. In *Ninth Annual Conference of the International Speech Communication Association*, 2008.

[19] T.J. Hazen, W. Shen, and C. White. Query-by-example spoken term detection using phonetic posteriorgram templates. In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 421–426. IEEE, 2009.

[20] Kristoffer Jensen. Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal on Advances in Signal Processing*, 2007:1 – 11, 2007.

[21] D. Kewley-Port, C. Watson, D. Maki, and D. Reed. Speaker-dependent speech recognition as the basis for a speech training aid. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'87.*, volume 12, pages 372–375. IEEE, 1987.

[22] J. Kim, C. Wang, M. Peabody, and S. Seneff. An interactive english pronunciation dictionary for korean learners. In *proceedings of Interspeech*, pages 1677–1680, 2004.

[23] C.-Y. Lee and James Glass. Bayesian unsupervised acoustic modeling. In *Proc. ACL*, page to appear, 2012.

[24] H. Meng, Y.Y. Lo, L. Wang, and W.Y. Lau. Deriving salient learners' mispronunciations from cross-language phonological comparisons. In *Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on*, pages 437–442. IEEE, 2007.

[25] A. Muscariello, G. Gravier, and F. Bimbot. Towards robust word discovery by self-similarity matrix comparison. In *Proc. ICASSP*, 2011.

[26] A. Muscariello, G. Gravier, and F. Bimbot. Zero-resource audio-only spoken term detection based on a combination of template matching techniques. In *Proc. Interspeech*, 2011.

[27] CS Myers and L. Rabiner. Connected digit recognition using a level-building dtw algorithm. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 29(3):351–363, 1981.

[28] Mitchell A. Peabody. *Methods for pronunciation assessment in computer aided language learning*. PhD thesis, MIT, 2011.

[29] X. Qian, F.K. Soong, and H. Meng. Discriminative acoustic model for improving mispronunciation detection and diagnosis in computer-aided pronunciation training (capt). In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[30] Yu Qiao, Naoya Shimomura, and Nobuaki Minematsu. Unsupervised optimal phoneme segmentation: Objectives, algorithm and comparisons. In *Proc. ICASSP*, pages 3989 –3992, 2008.

[31] L. Rabiner and B.H. Juang. *Fundamentals of speech recognition*. Prentice hall, 1993.

[32] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43–49, 1978.

[33] C. Tsurutani. Foreign accent matters most when timing is wrong. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[34] Y. Wang, A. Jongman, and J.A. Sereno. Acoustic and perceptual evaluation of mandarin tone productions before and after perceptual training. *The Journal of the Acoustical Society of America*, 113:1033, 2003.

[35] Y.-B. Wang and L.-S. Lee. Improved approaches of modeling and detecting error patterns with empirical analysis for computer-aided pronunciation training. In *Proc. ICASSP*, 2012.

[36] SM Witt and SJ Young. Phone-level pronunciation scoring and assessment for interactive language learning. *Speech communication*, 30(2):95–108, 2000.

[37] H.S. Wohlert. German by satellite. *The Annals of the American Academy of Political and Social Science*, pages 107–118, 1991.

[38] Y. Zhang and J.R. Glass. Unsupervised spoken keyword spotting via segmental dtw on gaussian posteriorgrams. In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 398–403. IEEE, 2009.

[39] Y. Zhang, R. Salakhutdinov, H. Chang, and J. Glass. Resource configurable spoken query detection using deep boltzmann machines. In *Proc. ICASSP*, 2012.

[40] V. Zue, J. Glass, D. Goodine, M. Philips, and S. Seneff. The summit speech recognition system: Phonological modelling and lexical access. In *Proc. ICASSP*, pages 49–52, 1990.