Language-Independent Methods for Computer-Assisted Pronunciation Training

by

Ann Lee

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

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2016

© Massachusetts Institute of Technology 2016. All rights reserved.

Author

Department of Electrical Engineering and Computer Science
August 31, 2016

Certified by

James Glass
Senior Research Scientist
Thesis Supervisor

Accepted by

Leslie A. Kolodziejski
Chair of the Committee on Graduate Students
Language-Independent Methods for Computer-Assisted Pronunciation Training

by

Ann Lee

Submitted to the Department of Electrical Engineering and Computer Science on August 31, 2016, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

Computer-assisted pronunciation training (CAPT) systems help students practice speaking foreign languages by providing automatic pronunciation assessment and corrective feedback. Automatic speech recognition (ASR) technology is a natural component in CAPT systems. Since a nonnative speaker’s native language (L1) background affects their pronunciation patterns in a target language (L2), typically not only native but also nonnative training data of specific L1s is needed to train a recognizer for CAPT systems. Given that there are around 7,000 languages in the world, the data collection process is costly and has scalability issues. In addition, expert knowledge on the target L2 is also often needed to design a large feature set describing the deviation of nonnative speech from native speech.

In contrast to machines, it is relatively easy for native listeners to detect pronunciation errors without being exposed to nonnative speech or trained with linguistic knowledge beforehand. In this thesis, we are interested in this unsupervised capability and propose methods to overcome the language-dependent challenges. Inspired by the success of unsupervised acoustic pattern discovery, we propose to discover an individual learner’s pronunciation error patterns in an unsupervised manner by analyzing the acoustic similarity between speech segments from the learner. Experimental results on nonnative English and nonnative Mandarin Chinese spoken by students from different L1s show that the proposed method is L1-independent and can be portable to different L2s. Moreover, the method is personalized such that it accommodates variations in pronunciation patterns across students.

In addition, motivated by the success of deep learning models in unsupervised feature learning, we explore the use of convolutional neural networks (CNNs) for mispronunciation detection. A language-independent data augmentation method is developed to take advantage of native speech as training samples. Experimental results on nonnative Mandarin Chinese speech show the effectiveness of the model and the method. Moreover, both qualitative and quantitative analyses on the convolutional filters reveal that the CNN automatically learns a set of human-interpretable high-level features.

Thesis Supervisor: James Glass
Title: Senior Research Scientist
Acknowledgments

I would like to thank my advisor, Jim Glass, for his encouragement and support over the years. Five years ago, Jim’s insight has brought me into the field of CAPT. Throughout the years, Jim has given me great freedom to pursue the research that I am truly interested in. I also would like to thank my committee, Louis Braida and Victor Zue, for their feedback in our committee meetings and the comments that help broaden and improve this thesis.

I am grateful to Helen Meng and Nancy Chen, who have kindly provided the two nonnative corpora that have been extensively used in this thesis. Without their generosity, none of my work could have been done. Helen also gave me lots of suggestions every time we met in conferences. Nancy even offered me a chance for summer internship in Singapore. In addition, Zhou Yu’s deep learning model implementation forms the basis of the second half of this thesis.

I am extremely fortunate to be a part of the SLS group. Each member has helped me in many ways, from answering all my questions, no matter in research or in life, to creating an enjoyable working environment. Many thanks to Dave, Ekapol, Stephen, Tuka and Yu for the discussions directly related to this thesis. Thanks Trish for building the demo system. Thanks all SLS members who have helped resolve my technical problems, proofread my writings, and maintain all the computing resources. From all of you guys, I have learned not only about research, but also how to be a better person. Many past members are also like mentors to me, providing lots of helpful suggestions for my future life journey. Thanks Hung-An for sharing what working in a real world is like during my summer internship. To the SLS staff, thanks Marcia for always being there helping us with all sorts of things. We were so lucky to have Scott, who is knowledgeable in everything. Special thanks to Mandy, who has kindly offered to help print and submit my thesis for me.

Last but not least, I would also like to thank all my friends and family in both the US and Taiwan. Living abroad would be really tough without all your love and support. Thanks Chung-Heng Yeh for always having faith in me and encouraging me. To my dear parents, thank you for everything.

This work is supported by Quanta Computer and Ping An Insurance.
Contents

List of Figures 11

List of Tables 17

Glossary 19

1 Introduction 23
  1.1 Overview 23
  1.2 Main Contributions 26
  1.3 Thesis Outline 26

2 Background and Related Work 29
  2.1 Nonnative Pronunciation Analysis 29
  2.2 Computer-Assisted Pronunciation Training (CAPT) 31
    2.2.1 Front-end: Learner-System Interaction and Corrective Feedback 31
    2.2.2 Back-end: Mispronunciation Detection and Diagnosis 32
  2.3 Speech Corpora 35
    2.3.1 Chinese University Chinese Learners of English (CU-CHLOE) Corpus 35
    2.3.2 Institute for Infocomm Research Computer-Assisted Language Learning (iCALL) Corpus 36
    2.3.3 A Comparison of CU-CHLOE and iCALL 38
  2.4 Summary 38

3 Personalized Unsupervised Mispronunciation Detection and Diagnosis 39
  3.1 Background 40
3.1.1 Phonetic Context and Pronunciation Error Patterns . . . . . . . . . . 40
3.1.2 Extended Recognition Network (ERN) for Mispronunciation Detection 40
3.2 Learner-level Phonetic Error Pattern Characterization: A Case Study of English 41
  3.2.1 Dataset . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
  3.2.2 Error Pattern Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . 43
3.3 System . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 50
  3.3.1 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 50
  3.3.2 Selective Speaker Adaptation . . . . . . . . . . . . . . . . . . . . . . 52
  3.3.3 Personalized Unsupervised Error Pattern Discovery . . . . . . . . . 53
  3.3.4 Error Pattern Decoding . . . . . . . . . . . . . . . . . . . . . . . . . . 57
3.4 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 61

4 Nonnative English Mispronunciation Detection Experiments 63
  4.1 Experimental Setup . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 64
    4.1.1 Dataset . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 64
    4.1.2 System Implementation . . . . . . . . . . . . . . . . . . . . . . . . . . 64
    4.1.3 Evaluation Metrics . . . . . . . . . . . . . . . . . . . . . . . . . . . . 65
  4.2 Results and Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 66
    4.2.1 Error Pattern Discovery . . . . . . . . . . . . . . . . . . . . . . . . . . 66
    4.2.2 Mispronunciation Detection and Diagnosis . . . . . . . . . . . . . . . 72
  4.3 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 78

5 Nonnative Mandarin Chinese Mispronunciation Detection Experiments 79
  5.1 Experimental Setup . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
    5.1.1 Dataset . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
    5.1.2 System Implementation . . . . . . . . . . . . . . . . . . . . . . . . . . 80
    5.1.3 Evaluation Metrics . . . . . . . . . . . . . . . . . . . . . . . . . . . . 83
  5.2 Results and Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 84
    5.2.1 Error Pattern Discovery . . . . . . . . . . . . . . . . . . . . . . . . . . 84
    5.2.2 Mispronunciation Detection and Diagnosis . . . . . . . . . . . . . . . 89
  5.3 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 94
6 CNN and Data Augmentation for Mispronunciation Detection

6.1 Background

6.1.1 Deep Learning Techniques for Computer-Assisted Pronunciation Training

6.1.2 Convolutional Neural Network (CNN)

6.2 CNN for Mispronunciation Detection and Native Training Data Augmentation

6.2.1 Motivation

6.2.2 Model

6.2.3 Data Augmentation

6.3 Experiments on Nonnative Mandarin Chinese

6.3.1 Experimental Setup

6.3.2 Results

6.4 Filter Analysis

6.4.1 Filter Activation Map

6.4.2 Filter Activation vs. Underlying Pronunciation

6.4.3 Filter Activation vs. Surface Pronunciation

6.4.4 Application: Tone Recognition

6.5 Summary

7 Conclusions and Future Work

7.1 Summary and Contributions

7.2 Future Work

7.2.1 Error Pattern Discovery in Spontaneous Speech

7.2.2 More Sophisticated Deep Learning Models

7.2.3 Towards a Complete CAPT System

7.2.4 Beyond Nonnative Speech: Disordered Speech and Dialects

A Symbols for English Phonemes

B Mapping from Standard Pinyin to Initial and Final Pinyin Units

Bibliography
## List of Figures

2-1  The pitch contours of the four main tones in Mandarin Chinese. .................................................. 37

3-1  (a) An FST of the canonical pronunciation of the word “north”, and (b) an example of an ERN of the word “north,” including one substitution error ($n \rightarrow l$) and one deletion error ($r \rightarrow \epsilon$). $\epsilon$ denotes an empty string in FST input/output. ................................................................. 41

3-2  Number of instances per underlying phone of the 256 words in the minimal pair set of the CU-CHLOE corpus, sorted by the count. Only underlying phones that appear more than 10 times are considered. .................................................. 42

3-3  (a) Number of unique underlying triphone patterns of each underlying phone class, and (b) histogram of the number of instances per underlying triphone pattern. ................................................................. 44

3-4  The underlying pronunciation of the word “north”, and an example of its surface pronunciation. By aligning the two pronunciations, one substitution error ($n \rightarrow l$), and one deletion error (the deletion of $r$) can be detected. .................................................. 45

3-5  (a) Confusion matrix between underlying pronunciations and surface pronunciations compiled from all the 100 learners in Set A. The sum of each row is normalized to 1. (b) The number of different realizations for each underlying phone class from all the 100 learners in Set A. For example, the underlying $ey$ has five unique surface pronunciations appearing in the set: $ae$ (0.3%), $ay$ (0.2%), $eh$ (12.5%), $ey$ (86.8%), $ih$ (0.1%). .................................................. 46
3-6  (a) Confusion matrix between underlying pronunciations and surface pronunciations compiled from utterances from speaker \textit{FC001}, (b) confusion matrix between underlying pronunciations and surface pronunciations compiled from utterances from speaker \textit{MC020}, and (c) mean and standard deviation of the number of unique surface pronunciations of each underlying phone class from individual learners in \textbf{Set A}. For example, the underlying \textit{ey} has 1.54 unique surface pronunciations, averaged from 100 speakers, with a standard deviation of 0.54.

3-7  (a) Normalized histogram of the number of unique surface pronunciations for each underlying phone class from individual learners in \textbf{Set B}. In this set, 81.4\% of the underlying phone have only one surface pronunciation. (b) Normalized histogram of the number of unique surface pronunciations for each underlying triphone from individual learners in \textbf{Set B}. For 93.5\% of the time, an individual learner produces only one surface pronunciation for a triphone pattern.

3-8  System flowchart. (a) The system gathers a number of utterances from a single learner, performs speaker adaptation, and runs forced alignment with a canonical lexicon to segment the utterances into target units. (b) An unsupervised error pattern discovery process is carried out to determine possible context-dependent pronunciation error candidates. (c) With the set of learner-specific error candidates, the system carries out forced alignment with ERNs, and an optional template-based rescoring process to decode the final output error patterns.

3-9  An example of the distance matrix between the utterance of a student reading an English text prompt \textit{“wrapped in a warm”} and the same utterance itself. The student made three pronunciation errors (\textit{r} \rightarrow \textit{w}, \textit{ae} \rightarrow \textit{aa} and a deletion error in \textit{r}). The two red boxes highlight the distance between \textit{r} and \textit{w}, and \textit{ae} and \textit{aa}, showing that low distance between speech segments of different underlying acoustic unit labels correlates with mispronunciations.
3-10 An example of how pronunciation rules are generated given an error candidate set for an English triphone $ao_r\_th$. Three types of errors, substitution, deletion and insertion errors, are considered.

4-1 Hierarchical structure of the metrics for evaluating the system’s performance in mispronunciation detection and diagnosis (adapted from [104]).

4-2 Evaluation of the quality (error coverage vs. size) of the error candidate sets from the five test scenarios under three degrees of speaker adaptation.

4-3 Confusion matrices built based on (a) ground truth error patterns, and (b) error patterns obtained from the unsupervised error pattern discovery process, projected onto the first two principal components after PCA.

4-4 Experimental results on mispronunciation detection (FAR vs. FRR) and diagnosis (DER vs. TR) on the Cantonese test set under three different degrees of speaker adaptation. The two oracles (shown in dashed lines) extract error patterns from the nonnative training set and perform one-pass ERN-based decoding, while the baseline performs phoneme recognition.

4-5 Experimental results on mispronunciation detection (FAR vs. FRR) and diagnosis (DER vs. TR) on the Mandarin Chinese test set under three different degrees of speaker adaptation.

5-1 An example of an ERN of the character (in Pinyin) “zhong”. Error patterns are incorporated on the Pinyin unit level (initial and final), as shown in dashed arcs. $\epsilon$ denotes an empty string in FST input/output. Tone errors are not considered here.

5-2 Evaluation of the quality (error coverage vs. size) of the error candidate sets on (a) the English test set, (b) the French test set, and (c) the Russian test set. The proposed unsupervised error pattern discovery process is compared with a supervised baseline that compiles error patterns from the English training set.

5-3 Confusion matrices built based on the ground truth error patterns from each speaker in (a) the three test sets, and (b) the three test sets and the English training set, projected onto the first two principal components after PCA.
5-4 Evaluation of the minimum error coverage on individual learners of the two systems on (a) the English test set, (b) the French test set, and (c) the Russian test set. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 88

5-5 Evaluation of FAR and FRR of the proposed framework operating with or without the speaker-dependent template-based rescoring process on (a) the English test set, (b) the French test set, and (c) the Russian test set. . . . . 90

5-6 Evaluation of DER and TR of the proposed framework operating with or without the speaker-dependent template-based rescoring process on (a) the English test set, (b) the French test set, and (c) the Russian test set. . . . . 92

5-7 Percentage of overlap with respect to different lengths between the output list from the system without rescoring and the output list from the system with rescoring on (a) the English test set, (b) the French test set, and (c) the Russian test set. . . . . . . . . . . . . . . . . . . 93

6-1 (a) An example of a single filter in the convolutional layer with a $3 \times 3$ filter size. (b) An example of the max-pooling layer with a $2 \times 2$ window size, and both the horizontal stride and the vertical stride equal to 2. (c) An example of how a feature map connects to an MLP layer. . . . . . . . . . . . . . . . 98

6-2 The CNN architecture for mispronunciation detection. The input to the CNN model is a combination of speech and text. The latter is represented in binary vectors. The model consists of one convolutional layer and one max-pooling layer, followed by one MLP layer and a two-class softmax layer for classification.100

6-3 (a) An LSTM cell and its mathematical formulation. $\sigma(\cdot)$ stands for the sigmoid function. $W$’s and $b$’s are model parameters (weights and biases). (b) The baseline BLSTM model. In the reverse LSTM layer, the input time sequence is fed into the LSTM cells in reversed order. The output vectors, $h$, from the last time steps from the two layers are concatenated and used as input to an MLP layer followed by a softmax layer for classification. . . . . 105
6-4 (a) Accuracy, and (b) FAR and FRR with respect to different amounts of non-native data (0%, 10%, 30%, 50%, 80% and 100%) used for model adaptation. Under the same amount of non-native data used, all performance differences between the adapted model and a pure non-native model are statistically significant \( p < 10^{-6} \) using McNemar’s test.

6-5 The filter activation map built on the non-native training set and the underlying pronunciations. The Pinyin units are sorted based on the order in hierarchical clustering shown in the left. Initials are colored in black, tone 1 finals are colored in blue, tone 2 in green, tone 3 in magenta, tone 4 in red and tone 5 in yellow.

6-6 Three examples of the distribution of the filter activation values of samples of a target tone versus samples of the rest of the tones.

6-7 The degree of separation of each filter, together with the best target underlying tone.

6-8 The filter activation map built on the non-native training set and the surface pronunciations. The Pinyin units are sorted based on the order in hierarchical clustering shown in the left. Initials are colored in black, tone 1 finals are colored in blue, tone 2 in green, tone 3 in magenta, tone 4 in red and tone 5 in yellow.

6-9 (a) The degree of separation of each filter, together with the best target surface tone. (b) The distribution of the filter with the best degree of separation score, filter 7 with surface tone 4 as the target.

6-10 An example of how pairing different texts with the same speech input generates different filter response.

6-11 Surface tone prediction accuracy with respect to different amounts of training data. The proposed model is built based on a set of pre-trained CNN filters, and the baseline is a CNN classifier trained from scratch.
Two possible ways to incorporate an attention mechanism into the proposed CNN model for mispronunciation detection. One way is to apply weights along the time axis, resulting in $\sum_{i=1}^{T} \alpha_i t_i$ as the new input to the MLP layer. The other way is to apply weights to each filter, resulting in $\sum_{i=1}^{K} \alpha_i f_i$ as the new input to the MLP layer.

A prototype of an English CAPT system. (a) Students can read the provided scripts, record and listen to their voice repeatedly until satisfied. (b) The system provides a list of detected errors with the mispronounced segments highlighted. The students can listen to their own pronunciation and compare with the correct pronunciation produced by the speech synthesis module.
List of Tables

2.1  A comparison between the CU-CHLOE corpus and the iCALL corpus. (\textasteriskcentered: only the transcribed part of the corpus is considered when we are assessing the scale for possible experiments) \hspace{0.5cm} 38

3.1  Two subsets from the minimal pair set for error pattern characterization. \hspace{0.5cm} 43

4.1  Division of the CU-CHLOE dataset for experiments. \hspace{0.5cm} 64

4.2  Top 10 most frequent ground truth error patterns from one Cantonese learner and one Mandarin Chinese learner. $\epsilon$ represents a deletion, and \# represents a word boundary. \hspace{0.5cm} 71

4.3  Top 10 most dissimilar error candidate sets from one Cantonese learner and one Mandarin Chinese learner. $\epsilon$ represents a deletion, and \# represents a word boundary. \hspace{0.5cm} 72

4.4  Top 10 output error patterns of the Cantonese learner and the Mandarin Chinese learner in Table 4.2 from a system without speaker adaptation. Correctly detected triphone patterns and diagnosis are shown in bold. $\epsilon$ represents a deletion, and \# represents a word boundary. \hspace{0.5cm} 73

4.5  Top 10 most frequently detected mispronunciations from the top 10 outputs from each speaker in the Cantonese test set. The number of occurrences of each triphone pattern in the text scripts is listed in parentheses. The diagnosis is sorted based on frequency with correct system diagnosis (diagnosis that appear in both system output and ground truth) shown in \textbf{bold}. $\epsilon$ represents a deletion, and \# represents a word boundary. \hspace{0.5cm} 76
4.6 Top 10 most frequently detected mispronunciations from the top 10 outputs from each speaker in the Mandarin Chinese test set. The number of occurrences of each triphone pattern is listed in parentheses. The diagnosis is sorted based on frequency with correct system diagnosis (diagnosis that appear in both system output and ground truth) shown in **bold**. $\epsilon$ represents a deletion, and $\#$ represents a word boundary.

5.1 Division of the iCALL dataset for experiments.

5.2 The mapping from Pinyinns ((a) initials and (b) finals) to phonemes for training the Mandarin Chinese speech recognizer. Note that tones should be included when a final is mapped to the corresponding phoneme sequence. For example, $ANG4$ should be transformed as “AE4 NG4”.

5.3 Top 10 output error patterns of one English L1 learner, one French L1 learner, and one Russian L1 learner. Correctly detected tripinyin patterns and diagnosis are shown in bold. $\epsilon$ represents a deletion error, and $\#$ represents a word boundary.

6.1 Division of the iCALL corpus for word-level mispronunciation detection experiments. Genders, L1 backgrounds, levels of proficiency and ages are all balanced across the three nonnative datasets.

6.2 Parameters used in CNN model training. All parameters are tuned using the development set except for the parameters for LRN, which are chosen following the experiment setting in [42].

6.3 Features for the SVM baseline for word-level mispronunciation detection.

6.4 Word-level mispronunciation detection performance.

6.5 Confusion matrix between different surface tone groups from the model with the highest prediction accuracy.

A.1 Mapping between the ARPAbet and IPA symbols of the English phonemes used in this thesis (adapted from [9]).

A.2 The 39 TIMIT phoneme groups used in the experiments. The phoneme symbols that are chosen to represent the whole group are shown in bold.
Glossary

**ASR** automatic speech recognition.

**BLSTM** Bidirectional Long Short Term Memory.

**CALL** computer-aided language learning.

**CAPT** computer-assisted pronunciation training.

**CD** Correct Diagnosis.

**CMN** Cepstral Mean Normalization.

**CMU** Carnegie Mellon University.

**CNN** convolutional neural network.

**CU-CHLOE** Chinese University Chinese Learners of English.

**CUHK** Chinese University of Hong Kong.

**DBN** Deep Belief Network.

**DE** Diagnostic Error.

**DER** Diagnostic Error Rate.

**DET** detection error trade-off.

**DNN** Deep Neural Network.

**DTW** Dynamic Time Warping.
ERN  Extended Recognition Network.

F0  Fundamental frequency.

FA  False Acceptance.

FAR  False Acceptance Rate.

fMLLR  feature-space Maximum Likelihood Linear Regression.

FR  False Rejection.

FRR  False Rejection Rate.

FST  Finite-State Transducer.

GALE  Global Autonomous Language Exploitation.

GMM  Gaussian Mixture Model.

GOP  Goodness of Pronunciation.

HMM  Hidden Markov Model.

iCALL  Institute for Infocomm Research Computer-Assisted Language Learning.

IPA  International Phonetic Alphabet.

L1  native language.

L2  second language.

LDA  Linear Discriminant Analysis.

LLR  log-likelihood ratio.

LRN  Local Response Normalization.

LSTM  Long Short Term Memory.
MALL  Mobile-Assisted Language Learning.

MAP  Maximum A Posteriori.

MFCC  Mel-Frequency Cepstral Coefficient.

MLP  Multilayer Perceptron.

MMI  Maximum Mutual Information.

MOOC  Massive Open Online Course.

OOV  Out of Vocabulary.

PCA  Principal Component Analysis.

PCM  Pulse-Code Modulation.

ReLU  Rectified Linear Unit.

RNN  Recurrent Neural Network.

SGD  Stochastic Gradient Descent.

SGMM  Subspace Gaussian Mixture Model.

SVM  Support Vector Machine.

TA  True Acceptance.

TR  True Rejection.

UBM  Universal Background Model.
Chapter 1

Introduction

1.1 Overview

With increasing globalization, there has been a rapid growth in the number of people with various native language (L1) backgrounds learning a second language (L2). On the other hand, the resources that traditional classroom teaching provides are limited. Let us examine English as an example. According to the Chinese Ministry of Education, there are roughly 270 million students across China learning English, while only 530,000 schools offered courses as of 2012 [8]. There is a gap towards the ideal setting of one-to-one teacher student interactions for language learning. As a result, computer-assisted language learning (CALL) systems have gained popularity due to the flexibility they provide for empowering students to practice their language skills at their own pace. Computer-assisted pronunciation training (CAPT) systems are a specific sub-area of CALL that helps students practice speaking by providing automatic pronunciation assessment and corrective feedback, and automatic speech recognition (ASR) technology has become a natural component in CAPT systems.

ASR technology is highly data-dependent. In the case of CAPT systems, most existing mispronunciation detection algorithms rely heavily on a large amount of labeled training data and linguistic knowledge in both the target L2 and the learners’ L1s. The pronunciation error patterns that a system focuses on are extracted by either comparing human transcriptions and canonical pronunciations from a lexicon, or consulting with language teachers. As a result, the system’s assessment ability is constrained by the training data or experts’ input.
To make matters worse, the data collection process is costly and has scalability issues. Since a nonnative speaker’s native language background affects their pronunciation patterns, it is almost impossible to collect a dataset that covers the 7,000 language backgrounds in the world [5].

On the other hand, in recent years, unsupervised speech processing has attracted much interest. For example, unsupervised acoustic pattern discovery [118] locates repeated segments in speech signals without any human annotations. Inspired by the human language acquisition process, unsupervised acoustic unit discovery [62] infers latent structures of a language, such as phones, syllables and words, directly from a collection of speech. Even without any expert knowledge, unsupervised approaches can achieve decent performance on tasks such as keyword spotting [118] and topic modeling on spoken documents [43].

In this thesis, we investigate unsupervised speech processing techniques to address the challenge of the limited amount of nonnative training data that is available when building CAPT systems. In contrast to a CAPT system’s demand on nonnative training data, it is relatively easy for native speakers to identify patterns that deviate from the norm without being trained on nonnative examples beforehand. This unsupervised mispronunciation detection capability is one of our interests in this research. As a result, we depart from the conventional supervised approach and frame the mispronunciation detection problem as an unsupervised error pattern discovery problem. Instead of depending on nonnative training data, we attempt to discover an individual learner’s common error patterns by exploiting the acoustic similarities between speech segments produced by the learner. The idea leads to a framework that identifies possible pronunciation confusion pairs when there are abnormally close phoneme pairs. The framework does not require nonnative training data and thus is L1-independent. Moreover, in theory, the system can be applied to different L2s as long as there is a recognizer available. We carry out mispronunciation detection experiments on both nonnative English and nonnative Mandarin Chinese spoken by students from different L1s to empirically verify these features of the proposed system.

Another challenge is that many mispronunciation detection techniques depend on a large set of human engineered features. Various features, such as spectral features, articulatory-acoustic features and intonation and stress-related features, have been designed to charac-
terize how nonnative pronunciations differ from the native pronunciations [107]. The feature engineering process is often based on expert knowledge on the target L2s and is separated from the detection model such that the features remain fixed during the training process.

To tackle the second challenge, we take advantage of the recent success in training deep learning models for unsupervised feature learning. Deep learning models have been shown to be effective in reducing the amount of domain knowledge needed for feature engineering [16]. Nevertheless, the training process requires a great amount of data. Take the state-of-the-art deep learning-based speech recognizers for example. The speech recognizer reported by IBM [90] uses 2000 hours of training data, while Microsoft [92] reports using 3300 hours of training data for a production-scale model, and Google [69] uses more than 20k hours of data for voice search. In light of the limited amount of the nonnative data that is available in CAPT applications, we propose a language-independent data augmentation method that takes advantage of native speech and creates mispronounced samples by randomly modifying the data. We explore the use of convolutional neural networks (CNNs) to learn a set of features jointly with the mispronunciation detector. A small amount of nonnative data can be used for model adaptation to further improve the system’s performance.

CAPT is an interdisciplinary research field. Aside from the ASR component, another issue one has to pay attention to while assessing a CAPT system is its pedagogical value. Individual tutoring has been shown to lead to significantly higher student performance than conventional classroom learning, since students receive personalized feedback [19]. Unfortunately, in current CAPT systems, the concept of personalization only exists in forms such as student performance tracking [3] and automatic material generation [94]. There is indeed a need for personalization in the mispronunciation detection module in ASR-based CAPT systems. Moreover, a good CAPT system should prioritize its feedback to the students instead of trying to address all the errors at once [102], as the latter may discourage the learners. As a result, in the experiments we will also evaluate the system’s ability in detecting personalized error patterns and prioritizing its output.

The vision is that with this thesis, we can provide language-independent mechanisms to develop CAPT systems that can behave like a native tutor, who is able to provide personalized feedback without being trained beforehand. The next section summarizes the main
contributions of this thesis, and an outline of the chapters is presented in the last section.

1.2 Main Contributions

The main contributions of this thesis can be summarized as follows:

**L1-independent and L2-portable mispronunciation detection and diagnosis.** We propose a personalized mispronunciation detection system based on unsupervised error pattern discovery for individual learners. Dealing with each single learner’s speech separately, the system is able to discover an individual learner’s pronunciation error patterns by analyzing the acoustic similarity between speech segments in an unsupervised manner. In this way, the assessment ability of the proposed framework is not confined by the nonnative training data that is available. We perform experiments on two corpora that are distinct in their L1s and L2s to demonstrate that the system is L1-independent and it is portable to different L2s. Experimental results also show that the personalized error pattern discovery process can accommodate high variations in pronunciation error patterns across speakers.

**CNN model for mispronunciation detection.** To date, there exists limited research on applying deep learning techniques on CAPT, mainly due to the limitation on the amount of nonnative data that is available. We explore using CNNs for mispronunciation detection, which is the first attempt in this field to the best of our knowledge. A data augmentation method is proposed to take advantage of native data for model training. Experimental results show that the CNN-based system outperforms other non-deep learning baselines. Moreover, the visualization on the convolutional filters shows that CNNs are able to automatically learn human-interpretable features, leading to the potential of building mispronunciation detectors without relying on human-engineer features.

1.3 Thesis Outline

The remainder of this thesis is organized as follows:

Chapter 2 provides an overview on previous research on CAPT. We discuss the pros and cons of different techniques and introduce the corpora used in the study.
Chapter 3 presents the personalized mispronunciation detection system that we propose. We start with data analysis on nonnative English speech showing how phoneme contexts affect an individual learner's pronunciation patterns. The data analysis motivates the design of our personalized mispronunciation detection system introduced next. The proposed system consists of two main steps: personalized unsupervised error pattern discovery and error pattern decoding, which will be explained in detail.

Chapter 4 presents experimental results on mispronunciation detection on nonnative English produced by Cantonese and Mandarin Chinese L1 speakers. We discuss the feasibility and effectiveness of the unsupervised error pattern discovery process, and the system's ability in prioritizing feedback.

Chapter 5 demonstrates the L2 portability of the proposed system by presenting experimental results on nonnative Mandarin Chinese produced by speakers of European descents. We discuss the system's ability in attending to variation in pronunciation error patterns across speakers, and the effectiveness of an unsupervised personalized rescoring module in the proposed system.

Chapter 6 presents a CNN-based mispronunciation detection framework and a data augmentation technique proposed to take advantage of native speech in model training. Both qualitative and quantitative analyses on the automatically learned convolutional filters are presented.

Chapter 7 concludes the thesis and discusses possible future work.
Chapter 2

Background and Related Work

In this chapter, we provide a review of the previous research in field of computer-assisted pronunciation training (CAPT). While CAPT is a huge research area consisting of not only technology but also pedagogical aspects, we put more emphasis on the related work in ASR for mispronunciation detection, which is the focus of this thesis. In addition, we introduce the two corpora used in the following study in this thesis.

2.1 Nonnative Pronunciation Analysis

Pronunciation errors in nonnative speech can occur at the phoneme level, syllable level, word level and sentence level, and they can be mainly divided into two categories: phonemic or prosodic errors. Phonemic errors include the more severe errors such as phoneme substitution, deletion and insertion, as well as less serious errors, which are phonetic realizations with a slight distortion of the native phoneme. The distortion usually does not affect the speaker’s intelligibility even though a native speaker can tell there is an accent. On the other hand, prosodic errors can involve a distortion in stress, rhythm or intonation. The tones in tonal languages such as Mandarin Chinese and Vietnamese fall into this category as well.

Language transfer theory has pointed out that learners apply knowledge from their L1 to an L2 [32], which indicates that learners coming from the same L1 background will very likely share the same set of pronunciation error patterns. As a result, many studies have focused on a specific L1 and L2 pair or even specific phonetic units. The studies are often
based on linguistic theory and supported by corpus analysis. A corpus-based study consists of comparisons between a collection of native and nonnative speech in various aspects, such as phonetic transcriptions, formant transitions and acoustic structures, ranging from segmental to suprasegmental levels [13].

On the basis of Japanese phonology, Goto [40] performed perception tests to demonstrate that Japanese speakers have difficulty in distinguishing the /l/-/r/ contrast in English. Meng et al. [71] carried out cross-language phonological comparisons between Cantonese and English phoneme inventory. Their study identified the missing and confusing phones in Cantonese compared to English. For example, since the Cantonese vowel inventory does not contain /e̯, æ, ø, η, ø/, these sounds may easily trouble Cantonese speakers when speaking English. A small corpus consisting of 21 speakers was collected and confirmed their findings. Burgos et al. [20] examined a corpus of Spanish learners of Dutch. Spanish is a system with only five vowels, while there are 16 vowels in Dutch including tense/lax contrasts. Their study verified that Spanish speakers of Dutch make vowel errors frequently, and they tend to fall back on the pronunciation of their L1 vowels. A subsequent analysis on the durations and spectral features of the vowels was also carried out [21]. Similar corpus comparison studies can be also be found in [46] for the Japanese(L1)-Korean(L2) pair and in [114] for Chinese(L1) and Korean(L2).

On the prosody side, Tseng et al. [101] compared the magnitude and contrast degree of F0 contours from native English speech and L2 English speech produced by Taiwanese speakers. The comparison results showed that the most significant difference between the L1 and L2 speech is the degree of contrasts, where the degree of F0 contrasts in Taiwanese English is much smaller than that in native English speech. The narrower range of F0 was also found in [76] for Japanese learners of English. On the other hand, German learners of Chinese seem to exaggerate the F0 contours while learning the tones [52].

Though a systematic study focusing on some specific L1 or L2 can reveal common pronunciation issues for a certain population or a target L2, which can benefit the design of CAPT systems, the size of the corpora used in the studies is usually small. While linguistic analyses and corpus studies can improve our understanding to the nature of nonnative pronunciations, the assessment ability of a CAPT system that relies on prior knowledge to
a specific L1 is constrained by the data available. This makes it hard to tailor a system to meet every student’s need. Moreover, within the same L1 population, there is great variation. Other factors, such as a learner’s level of competency in the target language, also affect error patterns [14, 33]. Just as different people have different speaking and writing styles, there is also a “pronunciation style” associated with an individual’s speech. As a result, there is indeed a need for unsupervised personalization in ASR-based CAPT systems.

2.2 Computer-Assisted Pronunciation Training (CAPT)

CAPT has been an interdisciplinary research area over the past two decades [33, 107]. The focus of research spans from front-end human computer interaction and material design based on pedagogical and linguistic knowledge to back-end speech processing techniques. In addition to being an active field of research, many ideas on CAPT systems have already been commercialized. In [107], a summary of existing commercial CAPT systems is listed. Below we discuss related work in both the front-end and back-end of CAPT systems, while more focus is put on the latter, which is the main topic of this thesis.

2.2.1 Front-end: Learner-System Interaction and Corrective Feedback

CAPT systems can be implemented in different formats. The most common one is reading practice, where the system provides text scripts for the students to read. The students are expected to follow the prompts, and the system evaluates their speech based on the prompts. The instructions provided by the system can be either text only or in speech. With audio instructions, the CAPT system behaves like a human instructor and allows students to repeat or practice under shadowing. To enrich the interactions and to attract students’ attention, the reading practice is often embedded in a gaming scenario or in a real-life dialogue scenario [94]. The content of the lessons can be fixed, or generated randomly or in an adaptive manner [94].

A more challenging type of CAPT system allows students to compose and speak in free speech. Question answering practices [103] or task-oriented dialogue systems [18, 112] are examples that properly set up a context for the students while allowing them to respond
spontaneously. As mobile devices have become prevalent nowadays, mobile-assisted language learning (MALL) systems have also emerged [65].

From a pedagogical point of view, it is clear that a CAPT system should be capable of not only detecting mispronunciations, but also providing corrective feedback. Considerable work has focused on providing multimodal feedback, such as audio feedback from speech synthesis, and visual feedback focusing on animations of lip, tongue and vocal tract movement [109]. Another issue that needs to be considered is the priority of a system’s feedback. While a student may make many mistakes, a good CAPT system should be able to prioritize its feedback in order to guide the student. The priority can be determined from the error frequency in a training corpus [77], or from human perception scores [102].

2.2.2 Back-end: Mispronunciation Detection and Diagnosis

Generally speaking, a CAPT system can provide two types of evaluation: individual error detection and pronunciation assessment. The former focuses on detecting errors at the word or sub-word level, and providing corrective feedback. The latter is an assessment on fluency on a holistic level, which is usually one or more sentences, and the feedback to the students is in the form of a numerical score. Below we focus on previous work on individual error detection, and especially on phonemic pronunciation error detection.

Automatic speech recognition (ASR) technology has been a core component in CAPT systems [33, 107]. As a result, the development of ASR-based CAPT systems is closely related to the advancement of ASR. Early work in mispronunciation detection includes Kewley-Port et al. [53] using a speaker-independent, isolated-word, template-based recognizer to build a speech training system for children. As ASR technology improves, various approaches based on hidden Markov model (HMM) likelihood probability, or posterior probability have been proposed. For example, Franco et al. [36] examined the posterior probabilities from acoustic models trained on different levels of nativeness. Witt and Young [108] took the ratio between the likelihood scores from forced alignments and recognition hypotheses into account and proposed goodness of pronunciation (GOP) scores. Over the past decade, different variations of GOP scores have been tested. With improvement in acoustic modeling, GOP-based approaches have also become more reliable [48, 49].
While in GOP-based mispronunciation detection, the GOP scores are only compared with phone-dependent thresholds for decision making, more sophisticated classifiers have also been explored. Wei et al. [106] considered log-likelihood ratios (LLRs) between the canonical phone model and a set of pronunciation variation models, and used a support vector machine (SVM) for classification. Inspired by the model adaptation techniques in speaker identification, Franco et al. [35] proposed to adapt a universal background model (UBM) by using correct phones and mispronounced phones separately and computed LLR between the two adapted models. More recently, Hu et al. [50] trained a neural network-based logistic regression classifier based on a set of log-posterior scores as features and showed great improvement over GOP and SVM-based systems. In addition to likelihood or posterior scores as features, Tepperman and Narayanan [97] incorporated articulatory information and train a decision tree for classification. Structural features, which describe the distance between phoneme distributions of a student’s speech, were also examined in [79, 120]. On the other hand, Lee et al. [59, 61] explored features from image processing describing the degree of misalignment between native and nonnative speech and used an SVM classifier for word-level mispronunciation detection.

However, the above methods only focus on binary detection and lack the capability of pinpointing the types of errors that were made, and thus do not have the ability to provide corrective feedback. To tackle this problem, some prior work focuses on specific phoneme pairs that are known to be problematic, and apply many existing pattern classification techniques. Strik et al. [93] focused on the velar fricative /x/ and the velar plosive /k/ in Dutch, extract acoustic-phonetic features, and apply linear discriminant analysis (LDA) to distinguish between the two. Amdal et al. [12] also trained LDA classifiers to distinguish between nine pairs of long and short vowels in Norwegian speech. Under the supervised framework, exact pronunciation error types are part of the system output and can be provided to the users, and thus the pedagogical value of the system can be improved.

Instead of defining a special set of classification targets, other work takes a more general approach. In [34, 66], the researchers directly treated the problem of mispronunciation detection as a speech recognition problem. However, the mismatch between the acoustic model, which is usually trained on native speech, and the student’s nonnative speech input often
hinders the recognition accuracy. In light of this issue, the idea of “extended recognition networks” (ERNs) has been proposed to constrain the recognizer’s search space. In an ERN, possible error types are incorporated into the lexicon during recognition [54, 71, 41, 68]. The enhanced lexicon has the advantage that the errors and the error types are detected together, and thus can be used for the system to provide diagnostic feedback. The incorporated possible error patterns can be identified in a knowledge-driven manner either by consulting with experienced language teachers [31, 54, 71, 41], or by carrying out phonological comparisons between the students’ L1 and the L2 [71, 41, 54]. Another way involves data-driven approaches that discover error patterns by aligning a human-transcribed L2 dataset with canonical pronunciations from a dictionary [31, 68].

While the above approaches carry the benefit of being able to provide corrective feedback, there exists the limitation that the common error patterns for a given L2, or an L1-L2 pair, must be known. However, both the linguistic expertise and a fully transcribed nonnative corpus are expensive and time-consuming to collect. To resolve this data dependency issue, Molina et al. [72] proposed to generate possible confusion words based on distance between acoustic models from an ASR engine. Wang and Lee [104] performed unsupervised phoneme posteriorgram clustering to discover mispronunciation patterns directly from data. On the other hand, Qian et al. [83] proposed a two-pass recognition framework. In the first pass forced alignment, an ERN built by pairing each canonical phone with an anti-phone model, which is built from all the non-canonical phone instances, is used. In other words, the first pass forced alignment detects regions where there are possible errors. In the second pass, free phone recognition is carried out in the problematic regions to identify the actual errors.

In this thesis, we present both ERN-based and classifier-based techniques for mispronunciation detection. For the ERN-based approach, we propose to discover personalized pronunciation error patterns directly from a student’s speech. For the classifier-based approach, we explore deep learning models to train a classifier directly from speech without the need for extracting human-engineered features. Both methods that we propose do not make assumption about students’ L1 backgrounds and the target L2. As a result, the methods are language-independent.
2.3 Speech Corpora

While there is an increasing population of students learning foreign languages, there exists a limited number of publicly available nonnative speech corpora, as compared with common speech corpora. In [85, 27], a summary of the available corpora is listed. As the goal of this thesis is to develop language-independent techniques to mispronunciation detection, we conduct experiments on the two corpora described below.

2.3.1 Chinese University Chinese Learners of English (CU-CHLOE) Corpus

The Chinese University Chinese learners of English (CU-CHLOE) corpus collected at the Chinese University of Hong Kong (CUHK) consists of two parts: 100 Cantonese speakers, including 50 males and 50 females, and 111 Mandarin speakers, including 61 males and 50 females, both reading a set of specially-designed English scripts [71]. The speakers were all university students who have learned English for a number of years, and their English pronunciation is viewed as intermediate to good by professional teachers.

There are 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 six 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): This subset contains 20 sentences.
   - Confusable words (cw): This subset is composed of 10 groups of confusable words, such as debt doubt dubious, or saga sage sagacious sagacity.
   - Minimal pairs (mp): This subset is composed of 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 [37] are included.

All recordings were collected using close-talking microphones and were sampled at 16kHz. Overall, each speaker contributed 367 utterances to this corpus, and all utterances except the TIMIT recordings were phonetically hand transcribed. The human transcription contains the mispronunciations that the learners have made and is called the “surface pronunciation.” On the other hand, the canonical pronunciation from a lexicon is called the “underlying pronunciation.”

### 2.3.2 Institute for Infocomm Research Computer-Assisted Language Learning (iCALL) Corpus

The second corpus we use in our study is a nonnative Mandarin Chinese corpus from the Institute for Infocomm Research (I2R) at Singapore. Below we provide an overview on Mandarin Chinese phonology first, and then introduce the corpus.

**Mandarin Chinese phonology**

Each character in Mandarin Chinese corresponds to a single syllable. A syllable starts with an optional initial (consonant), and then a final (a monophthong or diphthong vowel and an optional nasal consonant) together with a lexical tone. In addition to Chinese characters, each syllable can also be expressed in Pinyin, a standard romanization system that is often used for teaching Chinese.

Mandarin Chinese encodes semantics in tones, and different tones are characterized by different pitch patterns. There are four main tones and one neutral tone in Mandarin Chinese. Fig. 2-1 shows the pitch contours of the four main tones. In general, tone 1 has a high and level pitch contour, and tone 2 is a rising tone. Tone 3 has a more complex pattern, where the pitch first drops and then rises. Tone 4 can be characterized by a sharp fall in pitch. Tone 5, the neutral tone, is associated with weak or unstressed syllables, and its pitch pattern is often determined by the tone of the preceding syllable. While Fig. 2-1 shows the pitch pattern of
a tone in an isolated syllable, the pitch patterns become more complex when under different contexts. For example, when there are two consecutive tone 3 syllables, the first is realized as a tone 2. This phenomenon is called third tone sandhi [24].

The iCALL corpus

The Institute for Infocomm Research (I2R) computer-assisted language learning (iCALL) corpus [28] consists of 295 beginning learners of Mandarin Chinese (145 females and 150 males) of European origin reading 300 Pinyin prompts. The L1 backgrounds of the speakers are as follows: 52% Germanic (e.g., English, German), 32% Romance (e.g., French, Spanish and Italian) and 15% Slavic (e.g., Russian) [25]. The Pinyin prompts include 200 words and 100 sentences, which are specially designed so that the phonetic frequency matches that of the natural phonetic distribution in Mandarin Chinese.

All recordings were collected in quiet office rooms, sampled at 16kHz and encoded in 16 bit pulse-code modulation (PCM). All utterances are manually transcribed in Pinyin. In addition, every utterance has a proficiency score provided by an expert, ranging from 1 to 4, with 4 being the highest level. In [25, 27, 28], detailed analyses on the error patterns, including phonetic errors and tonal errors, are presented.
2.3.3 A Comparison of CU-CHLOE and iCALL

The CU-CHLOE corpus and the iCALL corpus differ not only in their L1s and L2s but also in their students’ proficiency levels and sizes. Table 2.1 summarizes the differences.

<table>
<thead>
<tr>
<th></th>
<th>CU-CHLOE</th>
<th>iCALL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>L1</strong></td>
<td>Cantonese and Mandarin Chinese</td>
<td>Mainly European languages, including English, German and French, etc. (24 in total) [27]</td>
</tr>
<tr>
<td><strong>L2</strong></td>
<td>English</td>
<td>Mandarin Chinese</td>
</tr>
<tr>
<td><strong>Proficiency level</strong></td>
<td>Intermediate to high</td>
<td>Beginners</td>
</tr>
<tr>
<td><strong>Annotation</strong></td>
<td>Partially transcribed</td>
<td>Fully transcribed with one proficiency score per utterance</td>
</tr>
<tr>
<td><strong>Scale</strong>*</td>
<td>86 short utterances per speaker (9.5 mins in duration per speaker)</td>
<td>300 utterances per speaker, including short words and longer sentences (27 mins in duration per speaker)</td>
</tr>
</tbody>
</table>

Table 2.1: A comparison between the CU-CHLOE corpus and the iCALL corpus. (*: only the transcribed part of the corpus is considered when we are assessing the scale for possible experiments)

2.4 Summary

In this chapter, we have presented an overview of CAPT research. We start from introducing research efforts on characterizing nonnative pronunciation patterns. Following that, we describe related work in both front-end and back-end of CAPT systems. A number of previous work focusing on phonemic mispronunciation detection is discussed. In the end, we introduce the two corpora used in our experiments, the CU-CHLOE corpus and the iCALL corpus.
Chapter 3

Personalized Unsupervised Mispronunciation Detection and Diagnosis

In this chapter, we propose a personalized mispronunciation detection framework that does not rely on nonnative training data. We start by analyzing an individual learner’s pronunciation error patterns. The analysis shows that a single learner tends to repeat the same error with very high probability if the context remains the same. On the basis of this finding, we design a mispronunciation detection framework that discovers an individual learner’s pronunciation error patterns in an unsupervised manner and decodes the actual pronunciation errors with context constraints. The proposed framework does not make any assumptions about the student’s L1 background and the target L2. As a result, the system is our general solution to the data dependency issues compared to the conventional approaches to mispronunciation detection.
3.1 Background

3.1.1 Phonetic Context and Pronunciation Error Patterns

When we speak, an underlying phonetic unit will be realized differently depending on its surrounding phonetic context, which accounts for the success of context-dependent acoustic modeling for ASR. Similarly, when a nonnative talker speaks, the phonetic context, coupled with the speaker’s L1 knowledge, also affects the surface pronunciation produced by the speaker. Harrison et al. [41] demonstrated that modeling context-sensitive pronunciation rules can remove the implausible pronunciation variations from the context-insensitive rules and thus performs better in detecting mispronunciations in English spoken by L1 Cantonese speakers. Kim et al. [54] built an English pronunciation dictionary for Korean learners by incorporating context-dependent phonological rules compiled from a Korean learner corpus, while Yang et al. [113] built an extended Korean lexicon for recognizing Korean spoken by L1 Mandarin Chinese speakers.

From the system’s perspective, considering context-sensitive pronunciation rules can reduce the search space of possible pronunciation errors. From the user point of view, receiving feedback with a specific context can make the learning process more efficient. While previous work has analyzed context-dependent pronunciation error patterns produced by students from a range of L1 backgrounds speaking various target L2s, to the best of our knowledge, no analysis has been done on examining an individual learner’s pronunciation error patterns under specific contexts.

3.1.2 Extended Recognition Network (ERN) for Mispronunciation Detection

In a finite-state transducer (FST)-based recognizer, the lexicon is represented as an FST that maps phone sequences to words. Fig. 3-1(a) is an example of the FST representation of the English word “north” in the lexicon. To deal with mispronunciations in nonnative speech, the FST can be enhanced by allowing multiple arcs corresponding to possible phone variations to be added, and an ERN is formed (Fig. 3-1(b)). Running recognition or forced alignment
Figure 3-1: (a) An FST of the canonical pronunciation of the word “north”, and (b) an example of an ERN of the word “north,” including one substitution error ($n \rightarrow l$) and one deletion error ($r \rightarrow \epsilon$). $\epsilon$ denotes an empty string in FST input/output.

with the expanded FST will result in output phone sequences that may be different from the canonical pronunciations. Pronunciation errors can be identified by comparing the canonical pronunciations and the output. For instance, if the decoded phone sequence from Fig. 3-1(b) is “l ao r th” \(^1\), we can conclude that a substitution error has occurred ($n \rightarrow l$).

The idea of adopting ERNs, sometimes called confusion networks or pronunciation networks, for mispronunciation detection can be applied to different L2s and on different speech units. For example, while the example shown in Fig. 3-1 is built for detecting phoneme-level errors in English [41, 84], authors in [60, 104] built ERNs for Pinyin (initials and finals) in Mandarin Chinese, and the work in [81] focuses on detecting word-level mispronunciations in Mandarin Chinese.

### 3.2 Learner-level Phonetic Error Pattern Characterization: A Case Study of English

In this section, we investigate phonetic pronunciation error patterns on a per learner basis. We take non-native English produced by Cantonese speakers as a case study. Our data analysis will demonstrate that, while a group of learners might have a wide range of pronunciation

\(^1\)In this thesis, we use ARPAbet to denote the English phonemes. See Table A.1 for the mapping between International Phonetic Alphabet (IPA) symbols and the ARPAbet symbols.
issues for a phone, an individual learner tends to consistently produce the same sound for the same triphone pattern. As a result, phonetic contexts and individual learner identity can help reduce the number of possible pronunciation variations the system has to consider. Our finding supports the idea of individual pronunciation style, and leads to the design of the system proposed in the next section.

3.2.1 Dataset

The following error pattern characterization focuses on the utterances from the minimal pair set of the Cantonese subset in the CU-CHLOE corpus. There are 256 words in this set of scripts. We use the CMU dictionary [1] as the reference canonical pronunciations, also called underlying pronunciations, and map the pronunciations to the TIMIT 39 phoneme sets [37]. Only underlying phones that appear more than 10 times are considered for analysis, resulting in 25 underlying phone classes. Fig. 3-2 shows the distribution of the number of instances of the 25 underlying phone classes. If we further consider the left and right contexts of an underlying phone, which forms an underlying triphone pattern, Fig. 3-3(a) shows the distribution of the number of unique underlying triphone patterns per underlying phone class. There are 407 underlying triphones in total. Fig. 3-3(b) shows the histogram of the number of instances per each underlying triphone pattern. The figure indicates that 66.6%
of the underlying triphone pattern appears only once.

Table 3.1 lists the two subsets used in the following discussion. Since errors with rare occurrences were often caused by misreading, Set A consists of phone instances whose surface pronunciations have been produced at least three times by an individual learner, contributing to 88.7% of the data. Next, to analyze the effect of phonetic contexts, we select a subset of Set A by only focusing on underlying triphones that appear more than once and removing the instances whose surface pronunciations appear less than three times across the entire dataset, which is again due to high possibility of misreading. This data selection process leaves us Set B, which takes up 55.6% of the data.

<table>
<thead>
<tr>
<th>Set</th>
<th>Description</th>
<th># speakers</th>
<th># instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Includes instances whose surface pronunciations were produced at least three times by an individual learner for each underlying phone class</td>
<td>100</td>
<td>69251</td>
</tr>
<tr>
<td>B</td>
<td>Includes instances whose underlying triphone patterns appear more than once and surface pronunciations were produced more than twice across the entire dataset</td>
<td>100</td>
<td>43405</td>
</tr>
</tbody>
</table>

Table 3.1: Two subsets from the minimal pair set for error pattern characterization.

### 3.2.2 Error Pattern Analysis

By aligning the human transcription (i.e., surface pronunciations) with canonical pronunciations from a lexicon (i.e., underlying pronunciations), phone-level pronunciation errors including insertion, substitution and deletion can be detected. Fig. 3-4 provides an example of an alignment and the detected pronunciation errors.

In the following analysis, we first compare the error patterns compiled from the whole dataset and the error patterns compiled from individual learners without the triphone context constraint. Then, we focus on error patterns from each individual learner and add triphone context constraints for further analysis.

**Error patterns from a group of learners vs. from individual learners**

To analyze pronunciation error patterns without context constraints, we use Set A in Table 3.1. After aligning the underlying pronunciations and the surface pronunciations for all
Figure 3-3: (a) Number of unique underlying triphone patterns of each underlying phone class, and (b) histogram of the number of instances per underlying triphone pattern.
100 learners in Set A, we count all the different phonetic realizations from the 100 learners and build a confusion matrix as shown in Fig. 3-5(a). Each row in the confusion matrix is normalized to sum to 1. As a result, the value of the \((i, j)\)-th element in the confusion matrix represents the percentage of the \(i\)-th underlying phone being mispronounced as the \(j\)-th surface phone. The confusion matrix suggests that the error rate depends upon underlying phone classes. The error rate is generally low, since 19 out of 25 underlying phone classes have an error rate of less than 10\%. The underlying phone \(er\) has the highest error rate (61.2\%) with the error pattern \(er \rightarrow ah\) being the most common. This result is due to the fact that the retroflex vowel is missing in the Cantonese phonetic inventory.

Fig. 3-5(b) shows the number of unique surface pronunciations for each underlying phone class (i.e., the number of nonzero elements in each row in the confusion matrix). We can see that the mispronounced segments are distributed into a wide range of error types. However, the error rate and the number of error types do not have high correlation. For example, while the error rate of \(d\) is high (22.5\%), there are only two ways of pronouncing the sound from the 100 learners, which are \(d\) and \(t\). On the other hand, while the error rates of \(ay\) and \(oy\) are similarly low (3.6\% and 3.0\%, respectively), there are seven different realizations for \(ay\) produced by the 100 speakers (\(aa, aw, ay, eh, ey, ih\) and \(iy\)), but the 100 learners only produce three different realizations for \(oy\), which are \(ay, ow\) and \(oy\).

The statistic on the number of error patterns becomes different as we change our focus from errors across the entire set to errors from individual learners. Fig. 3-6(a) and (b) are two examples of confusion matrices built by comparing underlying pronunciations and surface pronunciations for two individual learners, respectively. We can see that although

<table>
<thead>
<tr>
<th>Word</th>
<th>north</th>
</tr>
</thead>
<tbody>
<tr>
<td>Underlying pronunciation</td>
<td>(n\ ao r th)</td>
</tr>
<tr>
<td>Surface pronunciation</td>
<td>(l\ ao th)</td>
</tr>
</tbody>
</table>

Figure 3-4: The underlying pronunciation of the word “north”, and an example of its surface pronunciation. By aligning the two pronunciations, one substitution error \((n \rightarrow l)\), and one deletion error (the deletion of \(r\)) can be detected.
Figure 3-5: (a) Confusion matrix between underlying pronunciations and surface pronunciations compiled from all the 100 learners in Set A. The sum of each row is normalized to 1. (b) The number of different realizations for each underlying phone class from all the 100 learners in Set A. For example, the underlying ey has five unique surface pronunciations appearing in the set: ae (0.3%), ay (0.2%), eh (12.5%), ey (86.8%), ih (0.1%).
Figure 3-6: (a) Confusion matrix between underlying pronunciations and surface pronunciations compiled from utterances from speaker FC001, (b) confusion matrix between underlying pronunciations and surface pronunciations compiled from utterances from speaker MC020, and (c) mean and standard deviation of the number of unique surface pronunciations of each underlying phone class from individual learners in Set A. For example, the underlying ey has 1.54 unique surface pronunciations, averaged from 100 speakers, with a standard deviation of 0.54.
most of the pronunciation patterns are similar between the two learners due to their same L1 background, they pronounce some of the underlying phone classes quite differently, especially underlying *er, ey* and *uw*. In fact, although it seems that an underlying phone may have as many as seven different surface pronunciations according to Fig. 3-5(b), this number is much lower if the focus is on how an individual learner pronounces each sound. Fig. 3-6(c) plots the average number of unique surface pronunciations of each underlying phone class from the 100 speakers in Set A together with standard deviations. The figure shows that on average an individual learner pronounces a phone in one to three ways, with most standard deviations less than 0.5. These numbers are much lower than the numbers shown in Fig. 3-5(b). These statistics indicate that when a learner mispronounces a sound, he/she tends to repeat the errors rather than producing new types of errors in the future. As a result, restricting our focus to detecting pronunciation errors at an individual learner level can reduce the recognizer’s search space.

**Individual error patterns without context constraints vs. with context constraints**

Next, we use the data in Set B to examine the pronunciation variations for each underlying triphone produced by an individual learner. Fig. 3-7(a) plots the normalized histogram of the number of unique realizations per underlying phone accumulated from the 100 learners in Set B, while Fig. 3-7(b) shows the normalized histograms of the number of unique realizations per underlying triphone accumulated from the 100 learners.

The comparison between Fig. 3-7(a) and (b) shows that the number of different surface pronunciations for each triphone is lower than that for each monophone, implying that modeling context-dependent error patterns performs better than modeling context-independent error patterns, which is in line with the experimental results shown in [41]. Furthermore, Fig. 3-7(b) shows that more than 93.5% of the time, an individual learner pronounces an underlying phone in the same way if the contexts remain the same. This statistic indicates that most of the time, whenever the learner mispronounces an underlying triphone, the error remains the same across multiple instances. As a result, the above analysis shows that combining learner identity and context can help reduce the space of possible pronunciation
Figure 3-7: (a) Normalized histogram of the number of unique surface pronunciations for each underlying phone class from individual learners in Set B. In this set, 81.4% of the underlying phone have only one surface pronunciation. (b) Normalized histogram of the number of unique surface pronunciations for each underlying triphone from individual learners in Set B. For 93.5% of the time, an individual learner produces only one surface pronunciation for a triphone pattern.
errors. Moreover, this “pronunciation space” would be specific to each individual learner.

3.3 System

3.3.1 Overview

We propose a mispronunciation detection framework based on ERNs, since systems with ERNs can detect and diagnose pronunciation errors at the same time. When adopting ERNs for detecting mispronunciations, a smaller, yet more precise set of error candidates (i.e., the extended arcs in Fig. 3-1(b)) is more favorable due to the mismatch between the acoustic model in the recognizer, which is often trained with native speech in the target language, and the nonnative input speech. A larger error candidate set introduces noise during the decoding process.

On the basis of the lessons learned from the previous section, we take advantage of the learner identity information and triphone context to reduce the size of the error candidate set when building the ERNs. We propose decoding possible error patterns for each triphone pattern and for every individual learner from a collection of their speech, instead of relying on common error patterns that are known beforehand. Take English for example. Intuitively, if an underlying *aa* pronounced by a learner is very close to an underlying *ae* that is also pronounced by the same person, it is highly likely that the *aa*’s are mispronounced as *ae* (*aa* \(\rightarrow\) *ae*), the other way around (*ae* \(\rightarrow\) *aa*), or they are both mispronounced to the same target. As a result, our idea is to compute the similarity between speech segments produced by the learner and extract possible error candidates based on the similarity.

Fig. 3-8(a) shows the system flowchart. After a learner finishes reading the given set of text prompts, the system runs forced alignment based on a canonical lexicon in order to segment the utterances into target units depending on the granularity of the feedback. For example, for English the system can focus on phoneme level, while for Mandarin Chinese the feedback can be at the character (i.e., syllable) or Pinyin (i.e., initials and finals) level. Following the segmentation, the system works in two main steps.

First, we discover an individual learner’s possible pronunciation error patterns by analyzing the acoustic similarities between speech segments across all utterances produced by
Figure 3-8: System flowchart. (a) The system gathers a number of utterances from a single learner, performs speaker adaptation, and runs forced alignment with a canonical lexicon to segment the utterances into target units. (b) An unsupervised error pattern discovery process is carried out to determine possible context-dependent pronunciation error candidates. (c) With the set of learner-specific error candidates, the system carries out forced alignment with ERNs, and an optional template-based rescoring process to decode the final output error patterns.
the learner. A confusion matrix can be built based on the acoustic similarities, and possible error candidates can be identified from the highly confused speech segment pairs. This error pattern discovery process is carried out on a per-learner basis, resulting in a set of learner-specific error candidates. Second, the derived error candidates are incorporated into the lexicon to form a learner-specific ERN, and we can then decode the actual errors.

The proposed framework is unsupervised, since no transcription on nonnative speech is needed. Since the concept of the system design is independent of the target L2, below we explain each step in detail in a language independent fashion. Real system implementation depends on various factors, such as the granularity of feedback and the scale of per-speaker data, and will be discussed in the following chapters on experiments.

3.3.2 Selective Speaker Adaptation

Since the acoustic model of the recognizer is trained on native data, model adaptation can yield significant recognition accuracy improvements on nonnative speech [105, 110, 116]. In our case, we perform Maximum A Posteriori (MAP) adaptation [38] using the learner’s input utterances. One problem of adapting with all available material from the learner is that the model can easily be over adapted to mispronunciations. As a result, a selective speaker adaptation scheme is proposed.

We compute goodness of pronunciation (GOP) score [108] as an indication of how likely that a speech segment is pronounced correctly. The GOP score of an acoustic segment $p$ is defined as follows:

$$GOP(p) = \log \left( \frac{\sum_{q \in Q} p(O(p)|p) P(p)}{NF(p)} \right),$$

where $p(O(p)|p) P(p)$ is the posterior (likelihood times the prior) of the acoustic segment $O(p)$ with respect to the model for an acoustic unit $p$, $Q$ is the acoustic unit inventory, and $NF(p)$ is the duration (number of frames) of the acoustic segment. If we assume a uniform prior and use the maximum value to approximate the sum in the denominator in the logarithm
term, the GOP score of a phone segment \( p \) can further be defined as:

\[
GOP(p) = \frac{\left| \log \left( \frac{p(O^p|p)}{\max_{q \in Q} p(O^p|q)} \right) \right|}{NF(p)}.
\] (3.2)

In other words, GOP score is the duration normalized absolute difference between the log-likelihood of an acoustic segment from forced alignment (the numerator in Eq. 3.2) and the best log-likelihood score from acoustic unit classification within that segment (the denominator in Eq. 3.2). The larger the difference is, the more likely that the segment is mispronounced. As a result, only segments whose GOP score is below a threshold are used for adaptation. By adjusting the threshold, we can adjust the amount of data used for adaptation.

### 3.3.3 Personalized Unsupervised Error Pattern Discovery

Forced alignment with the adapted acoustic model produces a set of underlying acoustic segments. The acoustic segments can be phonemes, syllables, such as characters in Mandarin Chinese, or sub-syllable units, such as Pinyins in Mandarin Chinese. Assume that there are \( U \) acoustic units in the acoustic unit inventory. For each underlying acoustic segment, there are \( O(U) \) possible surface pronunciations. The goal of error pattern discovery is to select a subset of acoustic units that represent possible error candidates for each segment.

To achieve this goal, we propose a two-pass process for discovering a set of error candidates for a specific learner as shown in Fig. 3-8(b). We first identify confusion pairs by exploiting the acoustic similarities between speech segments. Then, we run forced alignment with ERNs built based on the confusion pairs and collect outputs from nbest lists to filter out unlikely candidates.

Before going into detail, we first define the following symbols. After forced alignment with canonical pronunciations, assume the utterances are segmented into \( N \) acoustic segments, \( \{s_i\}_{i=1}^N \). Let \( U = \{u_i\}_{i=1}^U \) be the acoustic unit inventory set, and \( R = \{r_i\}_{i=1}^R \) be the set of unique tri-unit patterns, such as a triphone in English. Each segment \( s_i \) is associated with one canonical acoustic unit label \( p_i \in U \) and one tri-unit label \( t_i \in R \).
First pass: DTW-based candidate selection

In the first pass of the error pattern discovery process, we compute a dynamic time warping (DTW) distance \[51\] between frames of mel-frequency cepstral coefficients (MFCCs) \[86\] for all the segment pairs \((s_i, s_j)\), denoted \(DTW(s_i, s_j)\), as an indication of the similarity between the segment pairs.

DTW is a dynamic programming process commonly used in speech for finding the best alignment between two sequences \[51\]. Assume two segments \(s_i\) and \(s_j\) have lengths \(n_i\) and \(n_j\), so that they can be represented as \(s_i = s^1_i, s^2_i, \ldots, s^{n_i}_i\) and \(s_j = s^1_j, s^2_j, \ldots, s^{n_j}_j\), respectively. A distance matrix \(\Phi_{ij}\) can be computed as

\[
\Phi_{ij}(k, l) = D(s^k_i, s^l_j),
\]

where \(D(s^k_i, s^l_j)\) denotes any possible distance metric between \(s^k_i\) and \(s^l_j\). As we are using MFCCs as the feature representation, a Euclidean distance is used to compute the distance between two frames of MFCCs.

Fig. 3-9 shows an example of a distance matrix built by aligning the utterance from a student reading the English text “\textit{wrapped in a warm}” with itself. The spectrogram indicates that the student mispronounced the \textit{r} sound in the word \textit{wrapped} as a \textit{w} sound, and the \textit{ae} sound in the same word as an \textit{aa} sound. In addition, the \textit{r} sound in the word \textit{warm} was deleted. From Fig. 3-9 we can see that there is a low distance region (a region with blue color) in the distance matrix corresponding to the alignment between the \textit{r} sound in the word \textit{wrapped} and the \textit{w} sound in the word \textit{warm}. Similarly, the distance between the \textit{ae} sound in the word \textit{wrapped} and the \textit{aa} sound in the word \textit{warm} is also low. The low distance between speech segments with different underlying acoustic unit labels indicates that the student might have mispronounced the segments. We compute the average DTW distance along the best alignment path on the distance matrix between two speech segments to characterize the similarity between speech segment pairs.

Given the \(n_i \times n_j\) distance matrix \(\Phi_{ij}\), a feasible path \(\psi\) is a path \(((\psi_{1,1}, \psi_{1,2}), (\psi_{2,1}, \psi_{2,2}), \ldots, (\psi_{|\psi|,1}, \psi_{|\psi|,2}))\) on \(\Phi_{ij}\) that satisfies the following constraints:

1. \(\psi_{1,1} = \psi_{1,2} = 1\), \(\psi_{|\psi|,1} = n_i\) and \(\psi_{|\psi|,2} = n_j\)
Figure 3-9: An example of the distance matrix between the utterance of a student reading an English text prompt “wrapped in a warm” and the same utterance itself. The student made three pronunciation errors (r → w, ae → aa and a deletion error in r). The two red boxes highlight the distance between r and w, and ae and aa, showing that low distance between speech segments of different underlying acoustic unit labels correlates with mispronunciations.

2. \[ 0 \leq \psi_{i+1,j} - \psi_{i,j} \leq 1, \forall i = 1, 2, ..., |\psi| - 1, j = 1, 2 \]

3. \[ \sum_{j=1}^{2} \psi_{i+1,j} - \psi_{i,j} > 0, \forall i = 1, 2, ..., |\psi| - 1 \]

In other words, a feasible path starts from position (1, 1) and ends at position \((n_i, n_j)\). Along the path, we move along either one or both dimensions by one step.

Let \(\Psi\) be a set of all feasible \(\psi\)'s. The DTW process finds an optimal path \(\psi_{ij}^*\) along which the accumulated distance is the minimum:

\[
\psi_{ij}^* = \arg\min_{\psi \in \Psi} \sum_{k=1}^{|\psi|} \Phi_{ij}(\psi_{k,1}, \psi_{k,2}). \tag{3.4}
\]

The optimal path can be found by computing a \(n_i \times n_j\) cumulative distance matrix \(C_{ij}\), where \(C_{ij}(k,l)\) records the minimum accumulated distance along a legal path from (1, 1) to \((k,l)\).
Each element in $C_{ij}$ can be computed as:

$$C_{ij}(k, l) = \begin{cases} 
\Phi_{ij}(k, l), & \text{if } k = l = 1 \\
C_{ij}(k, l - 1) + \Phi_{ij}(k, l), & \text{if } k = 1, 1 < l \leq n_j \\
C_{ij}(k - 1, l) + \Phi_{ij}(k, l), & \text{if } l = 1, 1 < k \leq n_i, \\
\min(C_{ij}(k - 1, l), C_{ij}(k, l - 1), C_{ij}(k - 1, l - 1)) + \Phi_{ij}(k, l), & \text{otherwise}
\end{cases}$$

(3.5)

To define the similarity between speech segments, only the final minimum accumulated distance is needed. The DTW distance between $s_i$ and $s_j$ is the minimum accumulated distance normalized by the length of the optimal path:

$$DTW(s_i, s_j) = \frac{C_{ij}(n_i, n_j)}{|\psi^*_ij|} = \sum_{k=1}^{|\psi^*_ij|} \Phi_{ij}(\psi^*_{k,1}, \psi^*_{k,2}) / |\psi^*_ij|.$$  

(3.6)

The smaller the distance is, the more similar the two segments are. A threshold $\tau$ is set so that we obtain a set of acoustic unit labels for each $s_i$ by collecting the underlying acoustic unit labels of their nearest neighbors as error candidates:

$$A_i = \{p_j | p_j \neq p_i, DTW(s_i, s_j) \leq \tau\}, i = 1, ..., N.$$  

(3.7)

We gather the candidates from segments with the same underlying tri-unit label, and form $R$ first-pass tri-unit-specific error candidate sets:

$$EC_{first}^i = \bigcup_{j, t_j = r_i} A_j, i = 1, ..., R.$$  

(3.8)

Second pass: nbest filtering

As $EC_{first}$’s are obtained based on acoustic distances between segments, the direction of mispronunciation, i.e., whether $s_i$ is mispronounced as $p_j$ or $s_j$ as $p_i$, is unclear. As a result, a second pass of nbest filtering is carried out to disambiguate this uncertainty.

The nbest [30] filtering process works as follows. For each of the acoustic unit $u_i$ in $U$, we build an ERN by incorporating $\bigcup_{j, p_j = u_i} A_j$, and run forced alignment on all utterances using
the ERN. For a segment $s_j$ whose canonical acoustic unit label is $u_i$, a set $B_j$ can be built by collecting the errors that appear in the nbest outputs from the forced alignment. After $U$ forced alignments, we form $R$ second-pass tri-unit-specific error candidate sets:

$$EC_{second}^i = \bigcup_{j, t_j = r_i} B_j, i = 1, \ldots, R.$$  \hspace{1cm} (3.9)

The final error candidate set for a tri-unit $r_i$ is the intersection of the error candidate sets from the two stages:

$$EC_{final}^i = EC_{first}^i \cap EC_{second}^i, i = 1, \ldots, R.$$  \hspace{1cm} (3.10)

### 3.3.4 Error Pattern Decoding

The goal of the error pattern decoding process is to find the actual pronunciation errors from $EC_{final}$’s. As we have learned from the case study in the previous section that an individual learner pronounces an underlying acoustic unit under the same context in only one way with very high probability, we apply a constraint to the decoding process so that only one error pattern should be selected for each tri-unit. Moreover, an inevitable challenge in CAPT is the mismatch between the acoustic characteristics of native and nonnative speech. As a result, depending on the amount of per-speaker data available, a speaker-dependent rescoring process can also be carried out to further improve the system’s performance. Below we explain the three stages (shown in Fig. 3-8(c)) in details.

**Pronunciation rule generation**

Given the set of error candidates, we first expand the error candidates into pronunciation error rules by considering three types of pronunciation errors: substitution, insertion and deletion. If segments of an underlying acoustic unit $\alpha$ have an acoustic unit $\beta$ in their error candidate set, it is likely that there are substitution errors ($\alpha \rightarrow \beta$), insertion errors ($\alpha \rightarrow \alpha \beta$), or deletion errors ($\alpha \rightarrow \epsilon$) if $\alpha$ has right or left context of $\beta$. Fig. 3-10 shows an example of an English triphone pattern $ao_r th$ (e.g., the $r$ sound in the word *north*). Since $ao$ and $th$, which are the left and right contexts of the triphone pattern, appear in the error candidate set, we consider the possibility of a deletion error. Substitution errors are considered for each
error candidates

\[ \text{error rules} \]

\[
\begin{align*}
\text{ao\_r\_th} & \quad \rightarrow \quad \text{<del>} \\
\text{ao\_r\_th} & \quad \rightarrow \quad \text{ow} \\
\text{ao\_r\_th} & \quad \rightarrow \quad \text{f} \\
\text{ao\_r\_th} & \quad \rightarrow \quad \text{v} \\
\text{ao\_r\_th} & \quad \rightarrow \quad \text{r\_ao} \\
\text{ao\_r\_th} & \quad \rightarrow \quad \text{r\_ow} \\
\text{ao\_r\_th} & \quad \rightarrow \quad \text{r\_f} \\
\text{ao\_r\_th} & \quad \rightarrow \quad \text{r\_v}
\end{align*}
\]

Figure 3-10: An example of how pronunciation rules are generated given an error candidate set for an English triphone \textit{ao\_r\_th}. Three types of errors, substitution, deletion and insertion errors, are considered.

error candidate that is neither \textit{ao} nor \textit{th}. In addition, we generate insertion error rules by appending each error candidate that is not \textit{th} after the \textit{r} sound. With these three types of error conditions, we expand the \( R \) error candidate sets into \( L \) pronunciation error rules, which form the \textit{candidate\_list}.

**ERN decoding with context constraints**

In the decoding process, we enforce the constraint that for a tri-unit pattern, only one pronunciation rule should be chosen. However, conventional Viterbi decoding makes decisions locally at every time step. As a result, we propose to run forced alignment iteratively to score each pronunciation rule one at a time and select error patterns in a greedy manner.

Given the \textit{candidate\_list} as input, the decoding process works as follows:

0. Initialize \textit{error\_list} as an empty list. Start iterating with the current best score set as the likelihood score from forced alignment with a canonical lexicon.

1. In each iteration, run multiple forced alignments. At each alignment, incorporate only one error pattern from the \textit{candidate\_list}, together with those already in the \textit{error\_list}, into the lexicon to build the ERN.
2. Pick the error pattern from the candidate list that produces the maximum likelihood score in decoding.

3. If the score improves upon the current best score, move the pattern to the error list and update the current best score. All the error patterns with the same tri-unit target are removed from the candidate list. For the rest of the error patterns, those with scores worse than the previous best score are removed from the candidate list.

4. If the score is worse than the current best score, or the candidate list becomes empty after updating, the process is completed.

In the end, the algorithm outputs the error list, an ordered list of learner-specific context-dependent error patterns, which is also the final output of the system.

**Speaker-dependent template-based rescoring**

If there are large amounts of per-speaker data, we propose to perform a rescoring step after each iteration of the greedy decoding process to compensate for the mismatch between the acoustic model and the learner’s nonnative input speech. The proposed rescoring is done via a speaker-dependent template-based speech recognizer built using the learner’s input speech. It has been empirically shown that template-based approaches complement standard parametric ASR systems in speech retrieval tasks [111].

The templates consist of the segments that are marked as correct from all the $L$ times of forced alignment in the decoding process. Two types of templates are built. The first type contains templates for each tri-unit $r_i$:

$$M^i_{tri} = \{s_j | t_j = r_i, s_j \text{ is correct}\}, i = 1, \ldots, R \quad (3.11)$$

The second type contains templates for each unique acoustic unit $u_i$:

$$M^i_{mono} = \{s_j | p_j = u_i, s_j \text{ is correct}\}, i = 1, \ldots, U \quad (3.12)$$

To compute the distance score for a pronunciation rule $l_i$, we first locate the mispronounced segments from the forced alignment result. Assume the format of $l_i$ is $\alpha \beta \gamma$ (tri-unit) →
\( \delta \) (acoustic unit), where \( \alpha_{-\beta-\gamma} \) represents an acoustic unit \( \beta \) under a left context of \( \alpha \) and a right context of \( \gamma \). The distance score of \( l_i \) is computed as follows.

1. If \( \delta \) represents a substitution error, the target tri-unit is \( \alpha_{-\delta-\gamma} \). We average the pairwise DTW distance, which was previously computed in the error pattern discovery step, between the mispronounced segments and the templates from the set \( M_{\text{tri}} \) of the target tri-unit as the target distance. If the template set is empty, we back off to use \( M_{\text{mono}} \) of \( \delta \) to compute the average pairwise DTW distance as the target distance.

2. If \( \delta \) represents a deletion error, the duration of \( \beta \) from the forced alignment with canonical pronunciations now becomes a sub-segment of \( \alpha \) and a sub-segment of \( \gamma \). The target distance is computed as the average DTW distance from between the new duration of \( \alpha \) and \( M_{\text{mono}} \) of \( \alpha \) and between the new duration of \( \gamma \) and \( M_{\text{mono}} \) of \( \gamma \).

3. If \( \delta \) represents an insertion error, the duration of \( \beta \) from the forced alignment with canonical pronunciations now becomes \( \beta \) with a preceding or succeeding acoustic unit \( \delta \). Assume \( \alpha_{-\beta-\gamma} \) now becomes \( \alpha_{-\beta_{-\delta-\gamma}} \). The target distance is computed as the average DTW distance from between the new duration of \( \beta \) and \( M_{\text{tri}} \) of \( \alpha_{-\beta_{-\delta}} \) and between the new duration of \( \delta \) and \( M_{\text{tri}} \) of \( \beta_{-\delta-\gamma} \). If either of the template set is empty, we back off to use \( M_{\text{mono}} \) of \( \beta \) or \( \delta \).

4. Average DTW distance with \( M_{\text{tri}} \) of \( \alpha_{-\beta-\gamma} \) (or back off to \( M_{\text{mono}} \) of \( \beta \) if the template set if empty) is computed as reference distance.

5. The distance score, which is defined as the target distance minus the reference distance, is computed.

The distance score can be viewed as a confidence score from the learner-specific template-based speech recognizer. The final score for each rule \( l_i \) is a weighted sum between the negative log likelihood score from the forced alignment and the distance score. We can use this score instead of the maximum likelihood score to select the best pronunciation rule in ERN decoding.
3.4 Summary

In this chapter, we start by characterizing an individual learner’s pronunciation error patterns. We use a nonnative English dataset from the CU-CHLOE corpus as a case study. The data analysis shows that the pronunciation variations within each underlying phone class produced by an individual learner is much less than the variations across a group of learners. Moreover, with context constraints such as triphones, the variations are further reduced. This finding motivates us to build a personalized mispronunciation detection system. The proposed framework works in two main steps. In the first step, we propose to discover an individual learner’s possible pronunciation errors by examining the acoustic similarity between the learner’s speech segments. In the second step, we decode the actual pronunciation errors with the constraint that only one pronunciation rule is selected for a specific acoustic context.

The proposed framework is L1-independent in a sense that no nonnative training data is needed. Experiments on both the CU-CHLOE and the iCALL corpus will be demonstrated in the following chapters.
Chapter 4

Nonnative English Mispronunciation Detection Experiments

In the previous chapter, we presented a general framework for personalized mispronunciation detection. In theory, the proposed framework does not require nonnative training data or expert knowledge on students’ L1s and thus can deal with students from different L1 backgrounds. Moreover, the proposed framework is portable to different L2s as long as there is a recognizer available\(^1\).

In this chapter, we perform experiments on detecting mispronunciations in nonnative English speech using the CU-CHLOE corpus. Due to the small scale and the relatively uniform L1 background (Cantonese and Mandarin Chinese) of the corpus, we focus on discussing the effectiveness of the two-pass error pattern discovery process and the system’s ability to prioritize corrective feedback.

In the following sections, we start by introducing the dataset used for the experiments and the actual system implementation given English as the target L2. Then, we evaluate the system’s performance under different degrees of speaker adaptation for both Cantonese and Mandarin Chinese L1 speakers’ speech. Experimental results show that the proposed framework effectively discovers error patterns in an unsupervised manner, detects mispronunciations and prioritizes the feedback.

\(^1\)We assume the target L2 has enough native resource for building a recognizer with good performance.
4.1 Experimental Setup

4.1.1 Dataset

We use the utterances from the subset “The North Wind and the Sun” from the CU-CHLOE corpus for experiments [71]. All the utterances in this subset are phonetically transcribed by a human expert. Both the Cantonese and the Mandarin Chinese subsets of the data are considered. We use half of the data for training the oracle systems and the other half of the data for testing. Table 4.1 shows the division of the corpus for our experiments.

<table>
<thead>
<tr>
<th>L1</th>
<th>Speakers</th>
<th># instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training (for oracles)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cantonese</td>
<td>25 males, 25 females</td>
<td>19,218</td>
</tr>
<tr>
<td>Mandarin</td>
<td>25 males, 25 females</td>
<td>19,173</td>
</tr>
<tr>
<td>Testing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cantonese</td>
<td>25 males, 25 females</td>
<td>19,227</td>
</tr>
<tr>
<td>Mandarin</td>
<td>36 males, 25 females</td>
<td>23,361</td>
</tr>
</tbody>
</table>

Table 4.1: Division of the CU-CHLOE dataset for experiments.

4.1.2 System Implementation

We build a Gaussian mixture model (GMM)-HMM-based recognizer with a monophone acoustic model trained on the TIMIT training set [37] using the Kaldi toolkit [80]. We use the TIMIT 39 phoneme sets throughout the following experiments. All waveforms are transformed into 39-dimensional MFCCs every 10-ms, including first and second order derivatives. Cepstral mean normalization (CMN) is done on a per speaker basis.

We carry out phoneme-level mispronunciation detection. The CMU dictionary [1] is used as the canonical pronunciation for forced alignment and system evaluation. Context-dependent triphones are used as the target tri-units. No template-based rescoring is performed due to the small amount of per-speaker data.
Figure 4-1: Hierarchical structure of the metrics for evaluating the system’s performance in mispronunciation detection and diagnosis (adapted from [104]).

4.1.3 Evaluation Metrics

The system’s performance is evaluated in two stages. First, we evaluate the quality of the error candidate set found by the two-pass error pattern discovery process. We evaluate the quality of an error candidate set by computing its size and error coverage. The size is represented as the average number of error candidates per triphone per learner, and the error coverage is defined as the percentage of the ground truth error patterns that appear in the error candidate set.

Second, we treat the system’s output as a ranked list, and evaluate its correctness. We follow the definitions in Fig. 4-1 and compute the following three metrics:

1. False Rejection Rate (FRR): the ratio between the number of correct phonemes that are identified as mispronounced (false rejection, FR) and the total number of correct phonemes.

   \[
   FRR = \frac{FR}{TA + FR} \tag{4.1}
   \]

2. False Acceptance Rate (FAR): the ratio between the number of incorrect segments that are accepted by the system as correct (false acceptance, FA) and the number of all the incorrect phonemes.

   \[
   FAR = \frac{FA}{TR + FA} \tag{4.2}
   \]

3. Diagnostic Error Rate (DER): the percentage of the correctly detected pronunciation errors (true rejection, TR) that have incorrect diagnostic feedback (diagnostic error,
\[ DER = \frac{DE}{TR} \]  

4.2 Results and Analysis

We evaluate the two stages, error pattern discovery and error decoding, separately. In both stages, different thresholds on the GOP scores for selective speaker adaptation are examined. The threshold is adjusted so that in each scenario, 0\%, 50\% and 90\% of the frames are used for adaptation, respectively.

4.2.1 Error Pattern Discovery

The proposed two-pass unsupervised error pattern discovery process is compared with two baselines and two oracles, resulting in five test scenarios. The first baseline is the error candidate set found by the one-pass error pattern discovery process. In other words, no nbest filtering is performed. For the second baseline, we take advantage of the English recognizer and carry out phoneme recognition on the student’s speech. All the competitive phonemes in the nbest output are combined to form an error candidate set. On the other hand, the first oracle system utilizes the nonnative training data, aligns human transcriptions with the canonical pronunciations from the lexicon, and compiles a list of context-dependent pronunciation errors as the error candidate set. Similarly, the second oracle system also takes advantage of the nonnative training data, while it compiles a list of context-independent errors as the error candidate set. By adjusting the distance threshold in the one-pass and two-pass error pattern discovery process, the length of the nbest output from phoneme recognition, and the threshold on the number of occurrences of the error patterns in the training data, we can generate trade-off between the size of the error candidate set and the coverage for all the test scenarios.

Fig. 4-2 shows the results. We compare the five test scenarios under three different degrees of speaker adaptation and examine the performance on the Cantonese speakers and the Mandarin Chinese speakers separately. We focus on the Cantonese test set (Fig. 4-2(a), (c) and (e)) in the following discussion, since the Mandarin test set has similar perfor-
Figure 4-2: Evaluation of the quality (error coverage vs. size) of the error candidate sets from the five test scenarios under three degrees of speaker adaptation.
mance trend as the Cantonese test set.

An error candidate set is considered to be high quality if it covers a large number of ground truth pronunciation error patterns with a small number of error candidates, since a compact error candidate set can reduce the search space and make the decoding process more efficient. The two oracles (shown in dashed lines) demonstrate the best error coverage we can get if we have a fully transcribed training dataset consisting of speakers from the same L1 background reading the same set of scripts. Their performance remains the same across different degrees of speaker adaptation as the training set is fixed.

From the figure we can see that the one-pass error pattern discovery process performs the worst, since the error candidate sets generated in this manner contains redundant information. This is the reason why the nbest filtering process is needed. With nbest filtering, the two-pass error candidate set achieves the same error coverage as the one-pass error candidate set with a smaller size. When both sets cover 60% of the ground truth errors, the nbest filtering process reduces the size of the error candidate set by at least 37% relative.

The performance of the nbest phoneme recognition baseline varies with different degrees of speaker adaptation. When there are on average three error candidates per triphone per speaker, the error candidate set generated from phoneme recognition with an acoustic model adapted using 90% of the student’s speech has an error coverage rate that is 4.6% absolute higher than the set generated with an acoustic model without adaptation. This difference is reasonable, since an adapted acoustic model matches the student’s speech better. On the other hand, the performance of the one-pass error pattern discovery process remains the steadiest across different degrees of speaker adaptation, since the process is fully dependent on the acoustic distance between segments from the student’s speech. While a speech recognizer is used for forced alignment, a recognizer without adaptation is still good enough to segment speech units of different acoustic characteristics.

The two-pass error pattern discovery process follows the characteristic of the one-pass approach such that the quality of the error candidate set remains steady across different degrees of speaker adaptation. As the two-pass process can be viewed as a combination of the one-pass acoustic-based approach and the ASR-based phoneme recognition approach, it generates an error candidate set with higher quality, i.e., smaller size and higher error coverage,
than the two approaches under all adaptation scenarios. Since the nbest phoneme recognition baseline grows stronger as the degree of speaker adaptation increases, the improvement of the two-pass process over the nbest phoneme recognition baseline decreases.

From Fig. 4-2(b), (d) and (f) we can see that the order of the performance of the five test scenarios from the Mandarin test set is the same as the order from the Cantonese test set. As a result, we believe that the conclusions we have drawn for Cantonese can also be applied to the Mandarin test set. While there remains a gap towards the supervised oracles, in the next set of experiments, we will show that the gap does not hinder the proposed system’s performance on mispronunciation detection.

The above results in Fig. 4-2 are compiled across all speakers in the Cantonese and Mandarin Chinese test sets. We further examine each individual learner’s error patterns separately. First, we focus on ground truth error patterns, which can be obtained by compiling the mismatch in the alignment between underlying canonical pronunciations and surface human phonetic transcriptions. On the basis of the ground truth error patterns, we build one confusion matrix for each individual learner. The $(i, j)$-th element in the confusion matrix records how many times the $i$-th underlying triphone is pronounced as the $j$-th surface phone. After computing the confusion matrix, we normalize each row so that all elements within each row sum to one, and concatenate all the rows in the matrix to form one single vector for each learner. Principal component analysis (PCA) is then carried out on vectors from all speakers in the two test sets for dimension reduction. Fig. 4-3(a) shows the confusion matrices of each learner projected onto the first and second principal components. Learners are labeled according to their L1 backgrounds, i.e., Cantonese or Mandarin Chinese.

From Fig. 4-3(a) we can see that the confusion matrices form two clusters corresponding to the two L1s in the subspace spanned by the first two principal components. The clusters indicate that students’ pronunciation error patterns are closely related to their L1 backgrounds. We select the two learners that are the closest to the centers of the two clusters as representatives to analyze the difference between the error patterns made by Cantonese and Mandarin Chinese learners. Table 4.2 shows the top 10 most frequent triphone pronunciation error patterns made by the two learners, respectively. We can see that there are common problems, such as the retroflex sounds ($er \rightarrow ah$ and $r \rightarrow e$), and the devoicing of a voiced
Figure 4-3: Confusion matrices built based on (a) ground truth error patterns, and (b) error patterns obtained from the unsupervised error pattern discovery process, projected onto the first two principal components after PCA.
Table 4.2: Top 10 most frequent ground truth error patterns from one Cantonese learner and one Mandarin Chinese learner. $\epsilon$ represents a deletion, and # represents a word boundary.

<table>
<thead>
<tr>
<th>Triphone</th>
<th>Word Example</th>
<th>Error</th>
<th>Triphone</th>
<th>Word Example</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>#_dh_ah</td>
<td>the</td>
<td>d</td>
<td>#_dh_ah</td>
<td>the</td>
<td>l</td>
</tr>
<tr>
<td>v_ah_l</td>
<td>traveler</td>
<td>$\epsilon$</td>
<td>t_r_ae</td>
<td>traveler</td>
<td>$\epsilon$</td>
</tr>
<tr>
<td>r_th_#</td>
<td>north</td>
<td>f</td>
<td>r_th_#</td>
<td>north</td>
<td>s</td>
</tr>
<tr>
<td>n_d_#</td>
<td>and, around, wind</td>
<td>$\epsilon$</td>
<td>n_d_#</td>
<td>and, around, wind</td>
<td>ah</td>
</tr>
<tr>
<td>l_er_#</td>
<td>traveler</td>
<td>ah</td>
<td>n_d_#</td>
<td>and, around, wind</td>
<td>$\epsilon$</td>
</tr>
<tr>
<td>ae_v_ah</td>
<td>traveler</td>
<td>f</td>
<td>aa_r_th</td>
<td>north</td>
<td>$\epsilon$</td>
</tr>
<tr>
<td>aa_r_th</td>
<td>north</td>
<td>$\epsilon$</td>
<td>#_r_r</td>
<td>traveler</td>
<td>ch</td>
</tr>
<tr>
<td>ng_er_#</td>
<td>stronger</td>
<td>ah</td>
<td>r_aa_ng</td>
<td>stronger</td>
<td>ah</td>
</tr>
<tr>
<td>l ow_k</td>
<td>cloak</td>
<td>aa</td>
<td>l_er_#</td>
<td>traveler</td>
<td>ah</td>
</tr>
<tr>
<td>ih_z_#</td>
<td>his</td>
<td>s</td>
<td>ih_z_#</td>
<td>his</td>
<td>s</td>
</tr>
</tbody>
</table>

fricative ($z \rightarrow s$). While both the Cantonese learner and the Mandarin Chinese learner have problem when pronouncing the voiced dental fricative sound $dh$, since it is missing in both Cantonese and Mandarin Chinese inventory, Cantonese learners tend to mispronounce it as a $d$ sound, and Mandarin Chinese learners tend to mispronounce it as an $l$ sound. Another difference is the voiceless dental fricative sound $th$. Most Cantonese learners tend to mispronounce it as an $f$ sound, while most Mandarin Chinese learners tend to mispronounce it as an $s$ sound.

Next, we examine the error candidates obtained from the unsupervised error pattern discovery process. Similarly, we build one confusion matrix for each individual learner. The $(i,j)$-th element in the confusion matrix records whether the $i$-th underlying triphone has the $j$-th phone in its error candidate set (1 or 0). After concatenating all the rows to form one single vector representing each learner and applying PCA for dimension reduction, we plot the confusion matrices projected onto the first two principal components in Fig. 4-3(b). Again, learners are labeled according to their L1 backgrounds.

Fig. 4-3(b) shows that although we do not have prior knowledge to a learner’s L1 background in the error pattern discovery process, the error patterns discovered in the unsupervised manner automatically reveal the L1 characteristics embedded in the L2 speech, resulting in two clear clusters corresponding to the two L1s, respectively. Table 4.3 lists the top 10 most dissimilar error candidate sets between one Cantonese learner and one Mandarin
Chinese learner, who are the two learners that are the closest to the centers of the two clusters. The error candidate sets are noisy and may contain many candidates that do not exist in ground truth errors. However, since a learner’s L1 background affects the realizations of different phoneme categories in the acoustic space, and the error candidates reflect the acoustic distance between phoneme segments produced by a learner, by comparing the similarity between the sets, we can distinguish between learners from different L1s. This demonstrates the potential of applying the proposed technique in L1 recognition.

### 4.2.2 Mispronunciation Detection and Diagnosis

As presented in Section 3.3.4, the iterative decoding process generates a ranked list of pronunciation errors. Table 4.4 shows the first 10 pronunciation errors from the outputs of the Cantonese and the Mandarin Chinese learners in Table 4.2. The ranking in the output list can be used to prioritize the feedback shown to the users. As a result, we can adjust the length of the output list and obtain a detection error trade-off (DET) curve between FAR and FRR.

The proposed system is compared with one unsupervised baseline and two supervised oracles. For the unsupervised baseline, we run phoneme recognition, compare the output with the canonical lexicon and detect mispronunciation when there is mismatch between the two. It is unsupervised, since the recognizer is trained on native speech, and no nonnative
training data is required. In the supervised oracles, we compile error patterns from the training data of the same L1, build an ERN and run one-pass forced alignment to decode the errors. Both context-dependent and context-independent error patterns are considered. The outputs from the baseline and the two oracles are ranked based on their GOP scores such that segments with higher GOP scores have higher priority. The length of the output list can also be adjusted so that we obtain one DET curve per system.

We start with discussing the system’s performance on the Cantonese test set. Fig. 4-4 shows the results. First, we focus on the DET curves (Fig. 4-4(a), (c) and (e)). As the degree of speaker adaptation increases, the FRRs of all four test scenarios decrease. In fact, the number of the detected errors ($TR + FR$) also decreases, since the acoustic model is partially adapted to mispronunciations. The phoneme recognition baseline consistently has the highest FRR under the same FAR as the other test scenarios, since the phoneme recognition process can be viewed as searching for mispronunciations in the full phoneme inventory space without any guidance.

If we focus on the top 50 outputs from each test scenario, the proposed system achieves lower FRR than a supervised system built based on context-independent error patterns. In the top 20 outputs, the proposed system achieves an FRR that is less than 1% absolute higher than a supervised system built based on context-dependent error patterns. These

<table>
<thead>
<tr>
<th>Triphone</th>
<th>Word Examples</th>
<th>Diagnosis</th>
<th>Triphone</th>
<th>Word Examples</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>w_ih_n</td>
<td>wind</td>
<td>iy</td>
<td>hh_ih_z</td>
<td>his</td>
<td>iy</td>
</tr>
<tr>
<td>ng_er_#</td>
<td>stronger</td>
<td>ah</td>
<td>#_ih_n</td>
<td>in</td>
<td>iy</td>
</tr>
<tr>
<td>#_dh_ah</td>
<td>the</td>
<td>d</td>
<td>n_d_#</td>
<td>and, around, wind</td>
<td>iy</td>
</tr>
<tr>
<td>aa_r_th</td>
<td>north</td>
<td>ow</td>
<td>#_w_ih</td>
<td>wind</td>
<td>l</td>
</tr>
<tr>
<td>l_er_#</td>
<td>traveler</td>
<td>eh</td>
<td>w_ih_n</td>
<td>wind</td>
<td>iy</td>
</tr>
<tr>
<td>#_b_l</td>
<td>blew</td>
<td>b_er</td>
<td>t_r_ae</td>
<td>traveler</td>
<td>€</td>
</tr>
<tr>
<td>#_w_ih</td>
<td>wind</td>
<td>uh</td>
<td>r_th_#</td>
<td>north</td>
<td>s</td>
</tr>
<tr>
<td>d_er_d</td>
<td>considered</td>
<td>eh</td>
<td>s_t_#</td>
<td>first, last</td>
<td>t_ah</td>
</tr>
<tr>
<td>t_r_ae</td>
<td>traveler</td>
<td>e</td>
<td>dh_ah_#</td>
<td>the</td>
<td>eh</td>
</tr>
<tr>
<td>k_ey_m</td>
<td>came</td>
<td>ah</td>
<td>ae_v_1</td>
<td>traveler</td>
<td>n</td>
</tr>
</tbody>
</table>

Table 4.4: Top 10 output error patterns of the Cantonese learner and the Mandarin Chinese learner in Table 4.2 from a system without speaker adaptation. Correctly detected triphone patterns and diagnosis are shown in bold. € represents a deletion, and # represents a word boundary.
Figure 4-4: Experimental results on mispronunciation detection (FAR vs. FRR) and diagnosis (DER vs. TR) on the Cantonese test set under three different degrees of speaker adaptation. The two oracles (shown in dashed lines) extract error patterns from the nonnative training set and perform one-pass ERN-based decoding, while the baseline performs phoneme recognition.
results demonstrate that although the error candidate set produced by the unsupervised error pattern discovery process may not cover as many ground truth errors as a supervised error candidate set, the set is sufficient for the iterative decoding process.

Next, we focus on DER (Fig. 4-4(b), (d) and (f)). As the degree of speaker adaptation increases, the DERs of all the test scenarios decrease. However, the degree of adaptation does not affect the supervised oracles as much as it affects the proposed system and the phoneme recognition baseline. This is because the supervised error candidate sets are fixed and contain less acoustically similar segments that would confuse the recognizer. On the other hand, the error candidates found by the proposed unsupervised error pattern discovery process are more acoustically similar to each other, since the first step of the process is based on analyzing acoustic distance between speech segments.

To analyze the source of mistakes on diagnostic feedback made by the proposed system, we pick the top 10 outputs from each speaker and compile a list of detected errors. Table 4.5 shows the top 10 most frequently detected triphone patterns in the list, with their diagnosis sorted based on number of occurrences. We can see that there are mainly two types of mistakes. First, many mistakes are due to the fact that the student mispronounced a consecutive sequence of phones. For example, the system detects a student mispronouncing both the $r$ and $th$ sounds in the word “north” into a $v$ sound (“$aa_r.th \rightarrow v$” and “$r.th.# \rightarrow v$”, where # stands for a word boundary), while in reality, the student deleted the $r$ sound and substituted the $th$ sound with the $v$ sound. The second source of mistake is due to recognition errors resulting from the imperfect acoustic model. For example, when examining the forced alignment result of the pronunciation rule “$aa_r.m \rightarrow l$”, we find that the segment of the $l$ sound takes up part of the $aa$ sound, since the ground truth error pattern is a deletion of the $r$ sound (“$aa_r.m \rightarrow \epsilon$”), and the acoustic model confuses between the semivowel $l$ and the back vowel $aa$. Although the proposed system makes mistakes in diagnosis, from the table we can see that the most frequent diagnosis of each triphone compiled from all the speakers is always correct. We believe a post-processing process can reduce the DER of the proposed framework, while this can be done for future work.

In fact, Table 4.5 also reveals an interesting characteristic of the proposed system. The number of occurrences of each triphone in the text scripts (shown in parentheses next to
Table 4.5: Top 10 most frequently detected mispronunciations from the top 10 outputs from each speaker in the Cantonese test set. The number of occurrences of each triphone pattern in the text scripts is listed in parentheses. The diagnosis is sorted based on frequency with correct system diagnosis (diagnosis that appear in both system output and ground truth) shown in **bold**. $\epsilon$ represents a deletion, and $#$ represents a word boundary.

<table>
<thead>
<tr>
<th>Triphone</th>
<th>Word Examples</th>
<th>System Diagnosis</th>
<th>Ground Truth Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa_r.th (4)</td>
<td>north</td>
<td>$\epsilon, v, ow, ng$</td>
<td>$\epsilon, n, d, b$</td>
</tr>
<tr>
<td>aa_r.m (2)</td>
<td>warm, warmly</td>
<td>$\epsilon, ng, l$</td>
<td>$\epsilon, f, ow$</td>
</tr>
<tr>
<td>ah_k_s (1)</td>
<td>succeeded</td>
<td>$\epsilon$</td>
<td>$\epsilon$</td>
</tr>
<tr>
<td>n_d_.# (8)</td>
<td>and, wind</td>
<td>iy, $\epsilon$, ih, ey</td>
<td>$\epsilon, t, ah, iy$</td>
</tr>
<tr>
<td>l_av_.j_h (1)</td>
<td>obliged</td>
<td>$\epsilon, r$</td>
<td>iy, ih, ey, uw, ah, ae</td>
</tr>
<tr>
<td>k_l_ow (5)</td>
<td>cloak, closely</td>
<td>$\epsilon, ow, ah$</td>
<td>$\epsilon, ah$</td>
</tr>
<tr>
<td>aa_r.# (2)</td>
<td>more</td>
<td>$\epsilon, l$</td>
<td>$\epsilon$</td>
</tr>
<tr>
<td>t_r_#_ (4)</td>
<td>traveler</td>
<td>$\epsilon, n$</td>
<td>$\epsilon, r$</td>
</tr>
<tr>
<td>ah_l_#_ (4)</td>
<td>traveler</td>
<td>ah, $\epsilon$, uh, ih</td>
<td>ah, ey, ay, aa</td>
</tr>
</tbody>
</table>

Table 4.6 lists the top 10 most frequently detected mispronunciations compiled from the top 10 outputs from all learners. Again, from the table we can see that many of the top ranked triphone patterns have high number of occurrences, and thus, the proposed system naturally puts more emphasis on them during the iterative decoding process. Also, many mistakes are due to recognition errors. For example, “aa_r.m $\rightarrow$ ng” comes from the ground truth error
Figure 4-5: Experimental results on mispronunciation detection (FAR vs. FRR) and diagnosis (DER vs. TR) on the Mandarin Chinese test set under three different degrees of speaker adaptation.
Table 4.6: Top 10 most frequently detected mispronunciations from the top 10 outputs from each speaker in the Mandarin Chinese test set. The number of occurrences of each triphone pattern is listed in parentheses. The diagnosis is sorted based on frequency with correct system diagnosis (diagnosis that appear in both system output and ground truth) shown in bold. $\epsilon$ represents a deletion, and $\#$ represents a word boundary.

“aa_r_m $\rightarrow \epsilon$” and a recognition error of mis-recognized a part of m as ng.

### 4.3 Summary

In this chapter, we have presented experimental results of the system proposed in Chapter 3 on nonnative English speech using the CU-CHLOE corpus. We examine the two components of the proposed system, unsupervised error pattern discovery and error pattern decoding, and demonstrate results on both the Cantonese test set and the Mandarin Chinese test set. First, we show that by combining DTW-based candidate selection process and ASR-based nbest filtering process, the two-pass error pattern discovery process can produce an error candidate set that is smaller, yet covers more ground truth errors than either of the single process. Second, we show that the proposed system detects a ranked list of errors that has a low false rejection error rate in its top ranked results, indicating its ability to prioritize corrective feedback. In the next chapter, we will demonstrate experimental results on nonnative Mandarin Chinese speech.
Chapter 5

Nonnative Mandarin Chinese
Mispronunciation Detection Experiments

In this chapter, we perform mispronunciation detection and diagnosis experiments on nonnative Mandarin Chinese speech using the iCALL corpus to demonstrate the proposed system’s portability to different L2s. Compared with the CU-CHLOE corpus, the iCALL corpus not only has more per-speaker data but also consists of students from a wider range of L1 backgrounds. In addition, the students are all beginning learners of Mandarin Chinese. Due to these characteristics, we focus on discussing the benefits of performing personalized error pattern discovery and the effectiveness of the template-based rescoring method for mispronunciation detection and diagnosis.

In the following sections, we start by introducing the dataset used for the experiments and the system implementation details given Mandarin Chinese as the target L2. Then, we focus on evaluating the system’s performance on English speakers, French speakers and Russian speakers, respectively, since these three groups are the top three largest L1 populations in the iCALL corpus. We focus on detecting phonemic pronunciation errors and leave tone errors aside. Although taking up only 20% of the pronunciation errors in the iCALL corpus¹,

¹Tone errors, which will be discussed in Chapter 6, are the main source of pronunciation errors in the iCALL corpus.
phonemic errors, especially phonemic errors on initials, are prone to cause miscommunication in Mandarin Chinese [67]. Experimental results show that the personalized error pattern discovery process generalizes well across speakers from different L1 backgrounds, and the personalized rescoring technique improves the performance on mispronunciation detection and diagnosis.

5.1 Experimental Setup

5.1.1 Dataset

We use the utterances spoken by the English, French and Russian speakers from the iCALL corpus for experiments, as they are the three largest L1 populations in the corpus (taking up 39.7%, 12.9% and 9.2%, respectively). All the utterances are transcribed in Pinyin by a human expert. Table 4.1 shows the division of the corpus for our experiments.

5.1.2 System Implementation

The Mandarin Chinese speech recognizer is trained on the GALE phase 2 Mandarin Chinese broadcast news corpus [4] with the Kaldi toolkit [80]. The corpus consists of 126 hours of Mandarin Chinese broadcast news speech, including recordings from Anhui TV, China Central TV, and Phoenix TV. All recordings are sampled at 16kHz and saved as single-channel 16-bit PCM. Transcripts in Mandarin Chinese characters are available for all the recordings. A 120-hr subset is randomly chosen as the training set, and the remaining 6-hr as the development set.
Table 5.2: The mapping from Pinyins ((a) initials and (b) finals) to phonemes for training the Mandarin Chinese speech recognizer. Note that tones should be included when a final is mapped to the corresponding phoneme sequence. For example, ANG4 should be transformed as “AE4 NG4”.

1The final i is realized as a continuation of the consonant when following initials z, c, s, zh, ch, sh or r. In other cases, i is realized as [i].
The GALE corpus consists primarily of Mandarin, while there are some English terms that are spoken. To cover the English terms, we use the CMU dictionary [1]. In order to handle English out-of-vocabulary (OOV) words, we also incorporate a pre-trained grapheme-to-phoneme model. For the Mandarin Chinese part, we use a 113k-word Mandarin lexicon from the MDBG Chinese dictionary [6], which provides mapping between words in Chinese characters to Pinyins. As there is no spacing that marks word boundaries between characters in the transcripts, the MMSEG toolkit [7], which carries out Chinese word segmentation based on maximum matching with words in the lexicon, is used to segment the transcripts into words. Finally, the set of 58 initials and finals, are further mapped to phoneme sequences (with tones) using the English phoneme inventory as shown in Table 5.2. The final 34k-word lexicon contains pronunciations in phonemes for both English and Chinese words.

All waveforms are transformed into 39-dimensional MFCCs plus three-dimensional pitch features every 10-ms, including first and second order derivatives. Cepstral mean normalization (CMN) is done on a per speaker basis, followed by LDA to reduce the feature dimension to 40 and feature-space maximum likelihood linear regression (fMLLR) for feature transformation. We build a subspace GMM (SGMM)-HMM-based triphone model trained with maximum mutual information (MMI) criterion [100]. The character error rate on the development set is 13.26% with a trigram word-based language model trained on the training set with Kneser-Ney smoothing.

In the following experiments, we carry out selective speaker adaptation for each learner in a heuristic manner and fix the degree of speaker adaptation. As students make fewer mistakes on shorter utterances [27], only utterances whose length is less than three characters and their canonical pronunciation from the Pinyin prompts are included into MMI training.

We focus on initials and finals, referred to as Pinyin units, as the target units for the mispronunciation detection task. Similar to the concept of context-dependent triphones used in English speech recognition, we use a concept of tripinyin, which consists of a Pinyin unit together with its left and right contexts. For instance, a tripinyin #_zh_ong is an initial zh under the context of a final ong, and # stands for the syllable boundary. Fig. 5-1 shows an example of an ERN of the character (in Pinyin) “zhong”, with one deletion on the initial and one substitution on the final.
Due to the characteristics of Mandarin Chinese phonology, during pronunciation rule generation, no deletion is allowed for the finals as they are the nucleus of a syllable, and we assume the students read all the words. A deletion error is considered for an initial when there is a final in its candidate set. We consider substitution and deletion errors only, as insertions are rare based on empirical analysis. In addition, we approximate the iterative decoding process by running the first iteration only ($L$ times of forced alignment, where $L$ is the number of pronunciation rules). This approximation improves the worst case time complexity from exponential to linear, while a pilot study did not show significant degradation in system performance due to the distinct syllable structure of Mandarin Chinese (an optional consonant initial plus a final).

5.1.3 Evaluation Metrics

As in the previous chapter, we evaluate the system’s performance in two stages. First, we compute the size and the error coverage of the error candidate set, which is the output from the two-pass personalized unsupervised error discovery process. In addition to the trade-off between size and error coverage, we also examine the worst case coverage, i.e., the minimum error coverage on an individual learner. In the second stage, we evaluate the system’s final output by computing FAR, FRR and DER, which are defined in Section 4.1.3.
5.2 Results and Analysis

5.2.1 Error Pattern Discovery

The proposed unsupervised error pattern discovery process is compared with a supervised baseline that compiles context-dependent error patterns from the English training set by aligning the surface Pinyin transcriptions with the underlying Pinyin prompts. By adjusting the distance threshold in the error pattern discovery process and the threshold on the number of occurrences of the error patterns in the training data, we can obtain trade-off curves between the error coverage and the set size as shown in Fig. 5-2. Results on the English, French and Russian test sets are shown separately.

From Fig. 5-2 we can see that when the set size is small, the proposed framework and the supervised baseline have similar behavior. As more error patterns are included in the error candidate set, the proposed framework achieves higher error coverage than the supervised error candidate set. When there are on average two error candidates per triphone for each learner, the proposed framework has an error coverage that is higher than the supervised error candidate set by 4.2% absolute on the English test set, 6.4% absolute on the French test set and 9.7% on the Russian test set. The larger difference in error coverage on the French and the Russian test sets demonstrates the limitation of the supervised approach, which requires a nonnative training set that consists of speakers from the same L1.

Although the supervised dataset consists of learners with English as their L1, the proposed framework still achieves higher error coverage on the English test set. This result is very different from the results on the CU-CHLOE corpus in the previous chapter. The main reason for this difference is the proficiency level of the learners. In the CU-CHLOE corpus, the speakers have higher proficiency levels in English. As a result, the error patterns are more related to their L1. Moreover, all the speakers in the training set and in the test set were reading the same set of scripts. Therefore, all the triphone patterns in the test set are covered, and the training set can capture 90% of the error patterns in the test set. On the other hand, the learners in the iCALL corpus are all beginners in Mandarin Chinese. Therefore, the error patterns are less related to learners’ L1s, but more related to each individual learners proficiency in Mandarin Chinese. As a result, even a training set consists of learners of the
Figure 5-2: Evaluation of the quality (error coverage vs. size) of the error candidate sets on (a) the English test set, (b) the French test set, and (c) the Russian test set. The proposed unsupervised error pattern discovery process is compared with a supervised baseline that compiles error patterns from the English training set.
same L1 cannot capture the error patterns made by another set of beginning learners well.

To support the above argument, we examine the similarity between the pronunciation patterns produced by the learners in the three test sets. We compute one confusion matrix for each individual learner. The $(i,j)$-th element in the confusion matrix represents how many times the $i$-th tripinyin pattern is realized as the $j$-th Pinyin unit based on ground truth human transcriptions. After normalizing and concatenating all the rows of the confusion matrix of each single speaker, Fig. 5-3(a) shows the results from all the test speakers projected onto the first two principal components after PCA. Different learners are labeled according to their L1s, i.e., English, French or Russian. From Fig. 5-3(a) we can see that there is no clear boundary between learners of different L1 backgrounds. This result, which is very different from the result in Fig. 4-3(a), indicates that L1 background is not the major factor that causes variance across the confusion matrices of different learners.

We further examine the similarity between the ground truth error patterns produced by the speakers in the English training set and the speakers in the three test sets. We consider only error patterns that appear more than once in the training set, which corresponds to the point where there are on average 1.5 error candidates per tripinyin per learner in Fig. 5-2. A confusion matrix is built for each learner. The $(i,j)$-th element in the confusion matrix records whether the error pattern of the $i$-th tripinyin mispronounced as the $j$-th Pinyin unit exists in the ground truth (0 or 1). After applying PCA, Fig. 5-3(b) shows the results projected onto the first two principal components. Learners are labeled in four groups: learners in the English test set, the French test set, the Russian test set, and the English training set. From Fig. 5-3(b) we can see that, the error patterns from the learners in the English training set only cover a small region in the subspace. There are many learners from all the three test sets outside the covered region, indicating that their error patterns are different from the error patterns in the training set. As a result, the supervised baseline only covers 60% to 65% of the error patterns in the test sets as shown in Fig. 5-2. This demonstrates the need of a personalized and unsupervised CAPT system, especially when the learners have lower proficiency levels.

In Fig. 5-4, we further examine the worst case coverage, i.e., the minimum coverage on an individual learner, with respect to different average error coverage across all learners. Having
Figure 5-3: Confusion matrices built based on the ground truth error patterns from each speaker in (a) the three test sets, and (b) the three test sets and the English training set, projected onto the first two principal components after PCA.
Figure 5-4: Evaluation of the minimum error coverage on individual learners of the two systems on (a) the English test set, (b) the French test set, and (c) the Russian test set.
Table 5.3: Top 10 output error patterns of one English L1 learner, one French L1 learner, and one Russian L1 learner. Correctly detected tripinyin patterns and diagnosis are shown in bold. ϵ represents a deletion error, and # represents a word boundary.

<table>
<thead>
<tr>
<th>English</th>
<th>French</th>
<th>Russian</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_ing.# → iang</td>
<td>#_b_u → p</td>
<td>sh.i.# → ui</td>
</tr>
<tr>
<td>sh_u.# → ui</td>
<td>#_ch_ang → sh</td>
<td>#_van.# → ian</td>
</tr>
<tr>
<td>#_z.i → zh</td>
<td>zh_e.# → i</td>
<td>#_v.# → i</td>
</tr>
<tr>
<td>#_z.ai → zh</td>
<td>#_d.i → j</td>
<td>#_iang.# → iao</td>
</tr>
<tr>
<td>q_v.# → u</td>
<td>t.a.# → e</td>
<td>q.i.# → vn</td>
</tr>
<tr>
<td>#_p_eng → f</td>
<td>h_ou.# → u</td>
<td>#_c ao → s</td>
</tr>
<tr>
<td>#_uai.# → ui</td>
<td>q_iu.# → v</td>
<td>#_b.en → ϵ</td>
</tr>
<tr>
<td>q_van.# → uan</td>
<td>q_iang.# → ian</td>
<td>l_iu.# → v</td>
</tr>
<tr>
<td>l_v.# → u</td>
<td>#_c.i → j</td>
<td>#_d.i → l</td>
</tr>
<tr>
<td>#_sh_ui → x</td>
<td>#_q ian → x</td>
<td>f.an.# → ang</td>
</tr>
</tbody>
</table>

5.2.2 Mispronunciation Detection and Diagnosis

We evaluate the effectiveness of the speaker-dependent template-based rescoring step by comparing the mispronunciation detection and diagnosis performance of the proposed system operating in two modes, with and without the rescoring step. The weight on the distance score for rescoring is empirically set to 60. By adjusting the thresholds on the likelihood scores or the weighted scores, we can adjust the number of the output error patterns, and DET curves can be formed for the two test scenarios. Table 5.3 shows examples of the top 10 outputs from an English L1 learner, French L1 learner, and a Russian L1 learner.

Fig. 5-5 shows the DET curves of the two scenarios on the three test sets, respectively.
Figure 5-5: Evaluation of FAR and FRR of the proposed framework operating with or without the speaker-dependent template-based rescoring process on (a) the English test set, (b) the French test set, and (c) the Russian test set.
The DET curves of three test sets all fall into the same region. From the figure we can see that the template-based rescoring process improves the system’s performance by reducing both FAR and FRR. When the FAR is at 60%, the rescoring process reduces FRR by 22.8% relative on the English test set, 6.0% relative on the French test set and 21.9% relative on the Russian test set. On the other hand, Fig. 5-6 shows the DER with respect to TR of the two scenarios on the three sets, respectively. The rescoring process also reduces the DER by as much as 12.2% relative on the English test set, 4.6% relative on the French test set and 11.5% relative on the Russian test set.

The rescoring process may change either the ranking of the triipinyin rules or the diagnosis of each triipinyin, i.e., the ranking among the error candidates with respect to the same triipinyin. To analyze the effect that the rescoring process brings, we compare the output lists before and after rescoring. First, we compute the overlap between the two output lists and find that for the English test set, 29% of the errors detected by the system without rescoring no longer exist after rescoring. This percentage is similar across different test sets (28% for the French test set and 29% for the Russian test set). On the other hand, the rescoring process only introduces 1% of the new errors. As the rescoring process leads to lower FRR, we can conclude that the outputs removed by the rescoring process are indeed mainly correct pronunciations.

Second, within the triipinyin patterns that are detected as mispronounced under both scoring scenarios, more than 20% of the diagnosis on the triipinyins has been changed by the rescoring process. As the improvement on the DER is not as large, the rescoring process also makes errors. Last but not least, we examine the overlap between the two lists at different lengths in order to analyze how similar the rankings generated under the two scoring scenarios is. We focus on the top 100 outputs from the two lists, and Fig. 5-7 shows the percentage of overlap with respect to different lengths. The figures show that the change in the percentage of overlap with respect to the length of the lists is similar across the three test sets. In the top 10 results, the overlap is high, indicating the high agreement between the two lists on their top ranked output. After the top 10 results, the percentage of overlap starts to decrease and reaches a steady state below 70%. The decrease in overlap here results in the DET curves of the two scoring scenarios starting to diverge. As a result, we can conclude from the analysis.
Figure 5-6: Evaluation of DER and TR of the proposed framework operating with or without the speaker-dependent template-based rescoring process on (a) the English test set, (b) the French test set, and (c) the Russian test set.
Figure 5-7: Percentage of overlap with respect to different lengths between the output list from the system without rescoring and the output list from the system with rescoring on (a) the English test set, (b) the French test set, and (c) the Russian test set.
that the rescoring process effectively boosts the scores of some mispronounced segments and suppresses the scores of some correctly pronounced segments, and the changes brought by the rescoring process take place mainly in the top 20 to 50 outputs.

5.3 Summary

In this chapter, we have demonstrated the L2 language portability of the proposed personalized mispronunciation detection framework by empirical validation on nonnative Mandarin Chinese speech using the iCALL corpus. Experimental results on nonnative speech from speakers of English, French and Russian L1s show that the proposed personalized unsupervised error pattern discovery process is able to accommodate high variations of error patterns across learners. In addition, the proposed personalized template-based rescoring effectively reduces detection errors in an unsupervised fashion.
Chapter 6

CNN and Data Augmentation for Mispronunciation Detection

In this chapter, we tackle the second challenge in developing CAPT systems, the dependency on human-engineered features, by exploring convolutional neural networks (CNNs) for the task of mispronunciation detection. We treat the problem of mispronunciation detection as a binary classification problem and build a CNN architecture that predicts whether the input speech is correct or not directly from the speech and text. Given that there is a limited amount of nonnative training data, we propose a simple yet effective data augmentation method by generating mismatched text and native speech pairs to create a native training set. Experiments on the iCALL corpus show how deep learning approaches outperform other non-deep learning baselines, and model adaptation with a small amount of nonnative data further improves the detection performance. Moreover, we visualize the filter response and perform both qualitative and quantitative analyses to show that the automatically learned convolutional filters capture different tone patterns in Mandarin Chinese, which is the main source of errors in the iCALL corpus. In the end, we demonstrate that the convolutional filters can be used as feature extractors through a tone recognition task.
6.1 Background

6.1.1 Deep Learning Techniques for Computer-Assisted Pronunciation Training

Deep learning has brought dramatic improvements to the machine learning world with applications in image recognition, speech recognition, natural language processing [17, 44, 55, 57, 95], etc. The multilayer structure and nonlinearity introduced within each node in deep learning models enable them to discover complex underlying structures from raw data. Limited human feature engineering is required for training deep neural networks (DNNs) compared to conventional machine learning tasks.

For ASR, DNNs have beaten traditional GMM and HMM based methods, which have dominated the field for the past twenty years [44]. Multilayer feed-forward structures can learn multiple levels of feature representations that are less sensitive to speaker variability while having higher discriminative power. Convolutional structures can reduce both temporal variability (e.g., different durations of the same underlying phone) and tolerate small frequency shifts (e.g., realizations of the same underlying phone from different speakers). On the other hand, recurrent structures have made the modeling of sequential data possible.

A natural way of applying DNNs in CAPT is to replace the GMM-HMM-based recognizers with DNN-based recognizers. An initial attempt by Qian et al. [82] shows performance improvement from an ERN-based mispronunciation detection system with a deep belief network (DBN)-HMM-based acoustic model. Hu et al. [48, 49] demonstrate that the improved discriminative power of DNN acoustic models also leads to better GOP scores with experiments on both nonnative English and nonnative Mandarin Chinese. The work by Li et al. [67] shows that DNNs also improve speech attribute modeling, which can be used for mispronunciation detection and diagnostic feedback. Different DNN variations have also been proposed. Li et al. [66] directly perform free phone recognition on nonnative English speech with a multi-distribution DNN-based acoustic model, which integrates acoustic features and preceding phone states.

On the other hand, [50, 61, 78, 96] are able to extract features with higher discriminative power from the improved acoustic model for classifier training. While the mispronunciation
detection performance has been improved, the features, such as likelihood or posterior probability scores, fluency, rhythm, intonation and stress, still come from human engineering. Recently, Yu et al. [117] explore using a bidirectional long short term memory (BLSTM) model for automatic scoring of nonnative spontaneous speech. The LSTM model [45] is a type of recurrent neural network (RNN), which can model time sequences by incorporating the output of the current time step as a part of the input of the next time step. The model directly uses speech as input without relying on other ASR-based features and achieves good correlation with human scoring.

6.1.2 Convolutional Neural Network (CNN)

Convolutional neural networks (CNNs) [58] are biologically inspired neural networks where the convolutional filters resemble the receptive fields in the visual cortex. CNNs have been widely used in computer vision and achieve state-of-the-art performance on image recognition [55], etc. In recent years, CNNs have also been explored in ASR and shown improvement over fully-connected feed-forward DNNs [10, 11, 89].

A typical CNN consists of a number of convolutional layers and pooling layers, followed by one or more fully-connected multilayer perceptron (MLP) layers and a softmax layer or loss layer for classification or regression. Let \( x \in \mathbb{R}^{m \times n \times r} \) be the input to a convolutional layer, where \( r \) is the number of channels, and each channel has an “image” of size \( m \times n \). In speech, we often use the spectrogram as the single-channel two-dimensional input image, and thus we rewrite the input as \( x \in \mathbb{R}^{F \times T} \), where \( F \) indicates the height in frequency (number of frequency bins), and \( T \) indicates the width in time (number of frames).

Assume there are \( K \) filters, or kernels, in a convolutional layer. Fig. 6-1(a) illustrates the convolution process of a single filter. A filter \( f_k \) is associated with a weight matrix \( W_k \) and a scalar bias \( b_k \). The output from a single filter is

\[
h_k = \theta(f_k \ast x),
\]

where \( \ast \) denotes the two-dimensional convolution operation, and \( \theta(\cdot) \) represents a nonlinear activation function. The nonlinear functions that are commonly used are rectified linear units
(a) An example of a single filter in the convolutional layer with a 3×3 filter size.
(b) An example of the max-pooling layer with a 2×2 window size, and both the horizontal stride and the vertical stride equal to 2. (c) An example of how a feature map connects to an MLP layer.

(ReLUs) (i.e., \( f(x) = \max(0, x) \)) [75], sigmoid functions and \( \tanh(x) \).

After the convolutional layer, pooling is a common technique in CNN for nonlinear down-sampling. Fig. 6-1(b) illustrates an example of max-pooling in which the maximum value within a window is chosen to represent the whole window. With pooling, we can reduce the number of parameters in the following layers and thus avoid overfitting. After layers of convolution and pooling operations, the final feature map is connected to layers of MLPs as shown in Fig. 6-1(c). In the end, a softmax layer or a loss layer is added on top of the MLPs, and the whole model can be optimized with respect to some suitable loss function.

6.2 CNN for Mispronunciation Detection and Native Training Data Augmentation

6.2.1 Motivation

Compared with ASR, the power of deep learning may not have been fully explored in the field of CAPT research yet due to the limited amount of nonnative training data available. In this work we explore CNN models for mispronunciation detection, which is the first attempt
CNN models have several advantages that make them favorable over other deep learning models. First, in a CNN, one filter is associated with one weight matrix and one bias term, and this weight-sharing characteristic reduces the number of parameters. Models with fewer parameters are more preferable when there is a limited amount of training data. Second, work in computer vision has shown that through weight-sharing, filters seem to discover human-interpretable visual patterns, from low-level ones such as dotted patterns or stripes to high-level ones such as dogs, human faces, beds, lamps [56, 121]. Lee et al. [63] have also found that convolutional models trained on speech data learn features corresponding to phonemes. As the CNN filters are learned automatically from the data, no feature engineering is needed. We are interested in analyzing the CNN filters that are trained on nonnative speech data to see if meaningful patterns or high-level concepts can be found.

Last but not least, in [42], Harwath and Glass have used a CNN as a feature extractor to transform the spectrogram of a word into a single vector in an embedding space that can further be connected to an image embedding space, which is built from another CNN trained on the images of the words. We are interested in building a pronunciation embedding space for nonnative speech, where common pronunciation patterns may form clusters.

### 6.2.2 Model

Fig. 6-2 shows the CNN architecture we introduce for mispronunciation detection. We treat the problem of mispronunciation detection as a binary classification problem, where the classifier distinguishes between correct and incorrect pronunciations, and build a CNN motivated by the architecture in [42].

The input to the CNN is a combination of speech and text. The speech signal is transformed to frames of feature representations such as MFCC, spectrogram or mel-filterbank. The text is transformed into a binary vector. We first define a vocabulary $V$ consisting of the basic units of which a word is composed. For example, for Mandarin Chinese we can define $V$ as the set of initials and finals with tones, while for English we can define $V$ as the English phoneme inventory or all triphone units. Then, we map the word into a sequence of the predefined basic pronunciation units based on a canonical lexicon. The word can then be
transformed into a binary vector of length $|V|$, where an element equals one if it appears in the mapped sequence. Assume the dimension of a single frame of the speech feature representation is $F$. We concatenate each frame with the $|V|$-dimensional binary vector and form the final input, which is a $(F + |V|) \times T$ matrix, and $T$ is the number of frames.

The proposed architecture has one convolutional layer with $K$ filters of size $(F + |V|) \times T_f$, where the width of the filter, $T_f$, corresponds to the duration in time (i.e., number of frames). The filters span across all frequency bins and all elements in the binary vector. Therefore, during the convolution process, the filters only move across the time axis. The idea is that if a segment in a word is mispronounced, the whole word should be marked as mispronounced. As a result, ideally the filters will be able to locate the mispronounced segment in time. After the convolution process, we apply nonlinear transforms with ReLUs.

Following the convention in computer vision, we perform local response normalization (LRN) [55] on the output from the convolutional layer. To perform LRN, each input value is divided by

$$
(1 + \frac{\alpha}{n} \sum_i x_i^2)^{\beta},
$$

where $n$ is the width of the local normalization window, and the sum is taken over the region centered at that value. After normalization, a max-pooling layer is added for nonlinear...
downsampling. Since the filters do not shift along the frequency axis, the output from a single filter from the convolutional layer is a vector instead of a two-dimensional matrix. Therefore, the max-pooling process is only carried out in the time axis, and the output from the max-pooling layer from a single filter is also a vector. After max-pooling, we concatenate the output vectors from all filters to form a supervector. The supervector is the input to an MLP layer with ReLU nonlinearity, and then the MLP layer is connected to a softmax layer with two output classes, i.e., correct versus incorrect pronunciations.

### 6.2.3 Data Augmentation

Data augmentation is a technique commonly used in machine learning to expand the training data set by incorporating transformed and unseen data in order to make a model generalize better [39]. In other words, data augmentation is to create simulated data that is still within the same class and add it to the training set. Commonly used methods include image rotation in computer vision, and noise injection such as adding Gaussian noise to the input for speech or time-scale modifications.

Borrowing the concept from data augmentation, we propose to take advantage of the large amount of native data and transform it into a training set with two classes to mitigate the problem of limited nonnative resources. Mispronunciation detection can be viewed from another perspective in which the goal is to decide whether the input speech and the text match or not. On the basis of this point of view, assume that native speakers always pronounce correctly, we propose to create negative training samples (i.e., mispronunciations) by modifying the text as the modified text forms a mismatched pair with its original speech. The modification process is based on a probability distribution over the vocabulary \( V \), which can be either a uniform distribution or a prior distribution computed from some training data or from expert knowledge. A balanced training set can easily be generated in this way.

### 6.3 Experiments on Nonnative Mandarin Chinese

We test the proposed framework on nonnative Mandarin Chinese speech using the iCALL corpus. We focus on detecting mispronunciation at the word level, i.e., each single Chinese
<table>
<thead>
<tr>
<th>Set</th>
<th># speakers</th>
<th># words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training (native)</td>
<td>N/A</td>
<td>227,486</td>
</tr>
<tr>
<td>Training (nonnative, for oracle)</td>
<td>177</td>
<td>227,486</td>
</tr>
<tr>
<td>Development</td>
<td>63</td>
<td>81,075</td>
</tr>
<tr>
<td>Test</td>
<td>55</td>
<td>69,547</td>
</tr>
</tbody>
</table>

Table 6.1: Division of the iCALL corpus for word-level mispronunciation detection experiments. Genders, L1 backgrounds, levels of proficiency and ages are all balanced across the three nonnative datasets.

character or syllable. The errors may be phonemic or prosodic, i.e., tone errors.

### 6.3.1 Experimental Setup

**Dataset**

The iCALL corpus is divided into training, development and test sets as shown in Table 6.1. Genders, L1 backgrounds, levels of proficiency and ages are all balanced across the three sets. The native dataset comes from the GALE phase 2 Mandarin Chinese broadcast news corpus [4]. We run forced alignment using the SGMM-HMM-based Mandarin speech recognizer built in Sec. 5.1.2 to segment the 120-hr training set into characters. A subset of 227k words is randomly chosen to be the native training set. We define the set of initials and finals with tones as the vocabulary for generating the binary text features. Half of the native training samples are randomly selected, and their texts are modified based on a uniform distribution over the vocabulary with constraints that initials and finals are modified into only initials or finals, respectively, to create a balanced training set.

**System implementation**

We use the Kaldi toolkit [80] to extract two types of speech features. The first type is 13-dimensional MFCCs plus three-dimensional pitch features together with the first and second order derivatives, resulting in a 48-dimensional feature vector for every 10-ms frame. We refer to the first type of feature as $mfcc$. The second type is the log energy from 40 filterbanks on the mel scale, together with log fundamental frequency and its time derivative, resulting in a 42-dimensional feature vector for every 10-ms frame, referred to as $40fbank$. Speaker-level
Table 6.2: Parameters used in CNN model training. All parameters are tuned using the development set except for the parameters for LRN, which are chosen following the experiment setting in [42].

We use the Theano toolkit [98] to train our network. As the implementation of CNNs requires that all input “images” should have the same size, we set the input width (i.e., duration in time) to be 100 frames and pad the shorter words with zeros or truncate the longer words equally from the beginning and the end. The network is trained with a cross-entropy criterion and updated with stochastic gradient descent (SGD) with momentum [88]. At each iteration in SGD, the current update is a convex combination of the previous update and the current gradient:

$$v = \gamma v + \eta \nabla_{\theta} J_i(\theta),$$

where $v$ is the update, $\gamma$ is a weight called momentum that controls how much information to be incorporated from the previous update, $\eta$ is the learning rate, and $\nabla_{\theta} J_i(\theta)$ is the current gradient on the $i$-th batch of the training samples. The parameters can then be updated as

$$\theta = \theta - v,$$

where $\theta$ represents the parameter set. After tuning the model’s performance using the development set, we summarize the final set of parameters in Table 6.2.

**Baselines and oracles**

We compare the system with three baselines. First, a support vector machine (SVM)-based binary classifier is trained using a set of GOP-based features [100] extracted from the nonnative training set. The GOP scores (see Eq. 3.2) are computed using the SGMM-HMM-based

<table>
<thead>
<tr>
<th>Convolutional layer</th>
<th># filters: 64, filter width: 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRN (Eq. 6.2)</td>
<td>$n$: 5, $\alpha$: $10^{-4}$, $\beta$: 0.75</td>
</tr>
<tr>
<td>Max-pooling</td>
<td>width: 4, horizontal stride: 2</td>
</tr>
<tr>
<td>MLP layer</td>
<td># nodes: 128, dropout ratio: 0.5</td>
</tr>
<tr>
<td>Training</td>
<td>learning rate $\eta$: $10^{-2}$, momentum $\gamma$: 0.9, # epochs: 50, batch size: 100</td>
</tr>
</tbody>
</table>
Table 6.3: Features for the SVM baseline for word-level mispronunciation detection.

| Feature                                                        | Dimension |
|                                                               |          |
| GOP score for each phone in the phone inventory               | 124      |
| Maximum GOP score across the word                             | 1        |
| Average GOP score across the word                             | 1        |
| Standard deviation of GOP scores across the word              | 1        |
| Average GOP score for initial phones                          | 1        |
| Average GOP score for final phones                            | 1        |

Mandarin Chinese speech recognizer built in Sec. 5.1.2. Table 6.3 lists the 129-dimensional handcrafted features. We use the LIBSVM toolkit [22] to train the SVM classifier with the radial basis function as kernel. The parameters are tuned through cross validation on the training set.

The second baseline takes advantage of the same Mandarin Chinese speech recognizer in Sec. 5.1.2, which is trained on 120 hours of native Mandarin Chinese speech without speaker adaptation on the nonnative speech. We perform free Pinyin unit recognition with a trigram Pinyin unit language model trained on the same 120-hr training set. The output from ASR is aligned with the Pinyin prompts, and a word is marked as mispronounced if there is mismatch between the two.

For the third baseline, we implement a bidirectional long short term memory (BLSTM) model trained on native data. The model architecture is similar to the one in [117]. Fig. 6-3(a) shows an illustration of an LSTM cell used in the study. An LSTM cell contains a memory cell that can remember a value for an arbitrary length of time. As shown in the figure, an LSTM cell has several gates that control its behavior. The input gate $i_t$ and the forget gate $f_t$ determine how much old memory to keep, and how much new input information to be stored, while the output gate $o_t$ controls how much information to output.

Fig. 6-3(b) shows the BLSTM model structure. There are two LSTM models operating in forward and backward order in time respectively. The input is the same combination of speech frames and binary text features as the one we use to train the proposed CNN model. After the two LSTMs have processed through the sequence of frames in their respective order, the two hidden vectors from the last time step are concatenated to form a supervector, which is the input to a one-layer MLP with ReLU nonlinearity and then a two-class softmax layer.
\[ i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \]
\[ f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \]
\[ c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \]
\[ o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \]
\[ h_t = o_t \tanh(c_t) \]

Figure 6-3: (a) An LSTM cell and its mathematical formulation. \( \sigma(\cdot) \) stands for the sigmoid function. \( W \)'s and \( b \)'s are model parameters (weights and biases). (b) The baseline BLSTM model. In the reverse LSTM layer, the input time sequence is fed into the LSTM cells in reversed order. The output vectors, \( h \), from the last time steps from the two layers are concatenated and used as input to an MLP layer followed by a softmax layer for classification.
<table>
<thead>
<tr>
<th>Model</th>
<th>Training data</th>
<th>Accuracy (%)</th>
<th>FAR (%)</th>
<th>FRR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>proposed</td>
<td>CNN native ((mfcc))</td>
<td>71.5</td>
<td>60.1</td>
<td>13.8</td>
</tr>
<tr>
<td></td>
<td>CNN native ((40fbank))</td>
<td>70.8</td>
<td>64.1</td>
<td>13.1</td>
</tr>
<tr>
<td>Baseline</td>
<td>SVM nonnative</td>
<td>58.0</td>
<td>42.2</td>
<td>42.0</td>
</tr>
<tr>
<td></td>
<td>ASR native</td>
<td>60.2</td>
<td>14.5</td>
<td>51.5</td>
</tr>
<tr>
<td></td>
<td>BLSTM native ((mfcc))</td>
<td>70.9</td>
<td>56.0</td>
<td>16.7</td>
</tr>
<tr>
<td></td>
<td>BLSTM native ((40fbank))</td>
<td>68.0</td>
<td>65.0</td>
<td>16.7</td>
</tr>
<tr>
<td>oracle</td>
<td>CNN nonnative ((mfcc))</td>
<td>80.1</td>
<td>26.2</td>
<td>17.0</td>
</tr>
<tr>
<td></td>
<td>CNN nonnative ((40fbank))</td>
<td>79.7</td>
<td>29.5</td>
<td>16.0</td>
</tr>
</tbody>
</table>

Table 6.4: Word-level mispronunciation detection performance.

for binary classification. The difference to the model in [117] is that we do not incorporate a separate set of handcrafted features such as fluency, rhythm, intonation and stress. As the CNN filters can be viewed as a kind of feature extractor that is automatically learned to extract meaningful features for mispronunciation detection, the BLSTM baseline provides a comparison where the LSTM cells are used for automatic feature extraction. The parameters in the BLSTM model are tuned on the development set, resulting in 256-dimensional LSTM cells and the MLP layer with 512 nodes.

In addition to the three baselines, we also test an oracle system by training the proposed CNN model using the nonnative training set to show an upper bound on the best performance we can get.

### 6.3.2 Results

**The power of CNN**

We evaluate the system’s performance by computing the accuracy of the output, which is the number of correctly detected words divided by the total number of words in the test set, false acceptance rate (FAR) and false rejection rate (FRR) as defined in Section 4.1.3. Table 6.4 summarizes the detection results.

As Table 6.4 shows, the proposed CNN model outperforms both SVM and ASR baselines. These comparisons demonstrate the power of deep learning. The low accuracy and high FRR from ASR are due to high recognition errors, since the acoustic model is trained on native speech only. While the SGMM-HMM-based speech recognizer has decent character error rate
on the native test set (13.26%), the mismatch between the native speech and the nonnative speech results in a Pinyin error rate of 33.5% on the nonnative test set. We find that most of the recognition errors are phonetic errors and not tone errors. On the other hand, despite being trained on a smaller native training set, the CNN model improves the accuracy over ASR by more than 10% absolute and reduces the FRR by more than 37% absolute, since the model is trained for the purpose of mispronunciation detection, and the filters capture speech characteristics that are essential to this task. The SVM baseline has very high FAR and FRR, and this may be due to the fact that the GOP scores are computed from the same SGMM-HMM-based recognizer used in the ASR baseline.

Next, when trained on the same speech feature and the same native training set, the CNN model outperforms the BLSTM model. As the two models both have one MLP layer followed by the two-class softmax layer, the difference in their performance results from the intrinsic difference between a set of convolutional filters and a sequence of LSTM cells. While it is hard to analyze how long the “short term memory” in the LSTM cells is, the filters in the proposed CNN architecture focus on a 50-ms window during the convolution process, and this duration seems to be effective for the task of mispronunciation detection. Moreover, the CNN model has 466k parameters, while the BLSTM model has 1.7M parameters. The difference in the order of the number of parameters supports one of our motivations to adopting CNN models with limited training data. As a side note, both CNN and BLSTM models trained on mfcc produce better performance than models trained on 40fbank. This result could be due to the fact that the 48-dimensional MFCC features contain delta and delta-delta information, while we do not further expand the 42-dimensional mel log-filterbank features with delta information, since we would like to keep the input feature dimensions, and thus the number of model parameters, on the same order for comparison.

Last but not least, there is an absolute 8.5% gap in accuracy between the CNN model trained on samples generated by the proposed data augmentation approach and the oracle, while the former has lower FRR than the latter does (around 3% absolute). This indicates that the gap between the accuracy comes from the high FAR of the proposed approach. Nevertheless, when designing CAPT systems, we should pay more attention to lowering FRR, since the system output should be as precise as possible, otherwise high FRR may
discourage the users. In addition, we try generating negative training samples based on a prior distribution on the error patterns computed from the nonnative training data, but no significant difference is found between the two approaches. We also try enlarging the size of the native training set, while the accuracy does not improve. This result indicates that the gap between the oracle and the proposed model is mainly due to the mismatch between native and nonnative acoustic rather than the generated error patterns.

Model adaptation

To bridge the gap between the performance of the proposed model and the oracle, we perform model adaptation with different amounts of nonnative data. In the following discussion, we present results from models trained on MFCC features, but the overall trend of the performance is the same for models trained on mel log-filterbank features.

Fig. 6-4(a) shows the accuracy of the adapted model with respect to different amounts of nonnative data used. The accuracy of a CNN model only trained on the same amount of nonnative speech is also shown for comparison. When the model is adapted with only 10% of the nonnative training data, which is less than 2.4hr of speech, the accuracy increases by 6.5% absolute compared with the model trained with only native data. Moreover, the accuracy of the adapted model is consistently higher than the accuracy of a model trained only on nonnative speech. A breakdown into FAR and FRR in Fig. 6-4(b) shows that the adaptation mainly improves the FAR, while the FRR remains steady. The adaptation reduces FAR by 22% absolute when only 10% of the data is used.

The result is consistent with the concept of model or domain adaptation commonly used in speech processing. The adaptation we carried out is also similar to the concept of pre-training, which is a technique commonly used in training DNNs [61, 119]. While there is limited amount of labeled data, one can take advantage of a large pool of unlabeled data or data from a similar domain to initialize the parameters into a proper region and then fine tune the model using the in-domain labeled data.
Figure 6-4: (a) Accuracy, and (b) FAR and FRR with respect to different amounts of nonnative data (0%, 10%, 30%, 50%, 80% and 100%) used for model adaptation. Under the same amount of nonnative data used, all performance differences between the adapted model and a pure nonnative model are statistically significant ($p < 10^{-6}$ using McNemar’s test).


6.4 Filter Analysis

In this section, we analyze the filter activation response, which is the output from the max-pooling layer, using the best CNN model we obtained, which is the one that is pre-trained on native training data and adapted on full nonnative training data. In the following, we discuss both the filter activation with respect to different underlying Pinyin units and the filter activation with respect to different surface Pinyin units. The analysis shows that the filters capture different tones, which is the main source of errors in the iCALL corpus. In the end, we demonstrate that we can use the convolutional filters as feature extractors, go beyond binary mispronunciation detection and carry out multi-class tone recognition.

6.4.1 Filter Activation Map

In the following analyses, we build a $|V| \times K$ filter activation map, where $V$ is the Pinyin inventory and $K$ is the number of filters, which is 64 in our case. Only the nonnative training data is used for building the filter activation map. We first run forced alignment to obtain the timing information of the initials and finals within each word in the training set. Assume the output from the max-pooling layer of the word is a $K \times T$ matrix. We average the output along the time axis according to the timing information from forced alignment, and form a $K$-dimensional vector representing the initial and a $K$-dimensional vector representing the final. Each element $(i, j)$ in the $|V| \times K$ filter activation map is the average of the $j$-th filter response on segments labeled as the $i$-th Pinyin unit.

6.4.2 Filter Activation vs. Underlying Pronunciation

Fig. 6-5 shows the filter activation map built based on the underlying pronunciation. Each row in the filter activation map can be viewed as a 64-dimensional vector that represents the corresponding Pinyin unit. We perform hierarchical clustering on these vectors and sort the Pinyin units based on the clustering result [74]. Different categories of the Pinyin units, including initials, tone 1, tone 2, tone 3, tone 4 and tone 5 finals, are shown in different colors. From the figure we can see that the initials and each tone form separate clusters, indicating the segments with the same tone have similar filter responses, respectively. The
Figure 6-5: The filter activation map built on the nonnative training set and the underlying pronunciations. The Pinyin units are sorted based on the order in hierarchical clustering shown in the left. Initials are colored in black, tone 1 finals are colored in blue, tone 2 in green, tone 3 in magenta, tone 4 in red and tone 5 in yellow.
Figure 6-6: Three examples of the distribution of the filter activation values of samples of a target tone versus samples of the rest of the tones.
clusters are formed due to the fully data-driven CNN training process. Since the tone error is the major source of mispronunciation in the iCALL corpus (taking up 80% of the errors), a natural feature for the classifier to capture is a word’s underlying tone. Tone 5 finals are scattered in different clusters as they only take up 7.0% of the training samples, and thus we exclude them in the following discussion. In addition, initials also form one cluster, with “N”, “M” and “L” being the only exceptions as the nasal and semivowel initials may seem closer to the finals.

If we look into each individual filter, some filters response are specific to a tone cluster, while others do not show specific activation patterns. For example, filter 58 seems to be highly activated only to underlying tone 1, filter 55 seems to be responsive to underlying tone 2, and filter 43 for tone 4. On the other hand, tone 3 finals do not seem to have a filter that is specific to them. This may be due to that samples with underlying tone 3 have the highest tone error rate (47.2%) compared with other tones (tone 1: 33.4%, tone 2: 38.0%, and tone 4: 37.2%). If we view the samples of the target tone as one cluster and the remaining samples as the second cluster, Fig. 6-6 plots three examples of distributions of the filter activation values with respect to the respective target tone cluster and the non-target tone cluster. There is clearly a separation between the distribution of the target underlying tone and the distribution of the other underlying tones.

In order to quantify how specific a filter activates with respect to a particular tone,
or how well a filter response separates a tone from the other tones, we compute a metric that evaluates how far two distributions are. The metric measures the maximum sum of the percentage of the distributions that can be correctly identified when a threshold is set to separate the two distributions. Assume the set of filter activation values of the target tone is $S_t = \{v_{t1}, v_{t2}, \cdots, v_{t|S_t|}\}$, and the set of filter activation values of the other tones is $S_o = \{v_{o1}, v_{o2}, \cdots, v_{o|S_o|}\}$, the metric, which we call the degree of separation, is defined as

$$\text{degree of separation} = \max_\theta \sum_{i=1}^{|S_t|} \mathbb{1}[v_{ti} > \theta] - \sum_{i=1}^{|S_o|} \mathbb{1}[v_{oi} \leq \theta], \quad (6.5)$$

where $\mathbb{1}[\cdot]$ is an indicator function, which equals to 1 if the input statement is true, and 0 otherwise. If the two distributions do not overlap, there exists a threshold that separates the two distributions completely, and thus the degree of separation equals one. On the other hand, if the two distributions are exactly the same, the degree of separation equals zero.

Fig. 6-7 shows the best degree of separation of the 64 filters with respect to the four tones, together with the corresponding target tone category on top of the bars. If we set a threshold at 0.6, there are six filters with their degree of separation exceeding the threshold. The filters and the corresponding target tones are filter 24 and tone 4, filter 43 and tone 4, filter 44 and tone 4, filter 55 and tone 2, filter 56 and tone 2, and filter 58 and tone 1. The quantitative analysis results are consistent with the qualitative analysis results from Fig. 6-5.

### 6.4.3 Filter Activation vs. Surface Pronunciation

We replace the underlying Pinyin labels with the surface Pinyin labels from human annotation, use the same timing information from forced alignment and repeat the same process to build a filter activation map based on surface pronunciations. Again, we perform hierarchical clustering and sort the surface Pinyin units based on the clustering results. The final filter activation map is shown in Fig. 6-8. The horizontal order of the filters is the same as the order in Fig. 6-5.

Similar to the earlier finding, surface Pinyin units with the same tones are clustered together. As a result, we can interpret the CNN’s behavior in a human-like manner. The CNN captures a word’s surface tone with a subset of filters, compares it with the underlying
Figure 6-8: The filter activation map built on the nonnative training set and the surface pronunciations. The Pinyin units are sorted based on the order in hierarchical clustering shown in the left. Initials are colored in black, tone 1 finals are colored in blue, tone 2 in green, tone 3 in magenta, tone 4 in red and tone 5 in yellow.
Figure 6-9: (a) The degree of separation of each filter, together with the best target surface tone. (b) The distribution of the filter with the best degree of separation score, filter 7 with surface tone 4 as the target.
tone captured by another subset of filters and makes its final decision. We carry out the same quantitative analysis on the filter activation map. In addition to examining whether each filter can separate a single surface tone from other surface tones, we also consider multiple tones in one distribution, e.g., the distribution formed by samples of surface tone 1 and tone 2 versus the distribution formed by samples of surface tone 3 and tone 4.

The degree of separation of each filter is shown in Fig. 6-9(a). The top three filters with the highest degree of separation are filter 7 and tone 4, whose distributions are shown in Fig. 6-9(b), filter 11 and tone 2 and 3, and filter 58 and tone 1. Note that filter 58 also has high degree of separation with respect to underlying tone 1. As a result, we can view filter 58 as an indicator for deciding whether a tone 1 final has any mispronunciation in tone.

Compared with the previous analysis on the underlying tones in Section 6.4.2, the distributions of filter responses with respect to surface tones have lower degrees of separation in general. The larger overlap between the distribution of the target surface tones and the other surface tones is due to the fact that surface pronunciations are embedded in speech signals, while underlying pronunciations are embedded in the binary features generated from text. The binary vectors are the same across speakers if their underlying pronunciations are the same. As a result, it is easier for a single filter to capture one underlying tone. On the other hand, speech features have higher variation across speakers even if the surface pronunciations are the same. This variability makes it harder for a single filter to capture all the instances of the same surface tone. Nevertheless, if we consider all the activation responses from the 64 filters at the same time, the clusters formed by the filter activation maps show that the difference between the tones are well captured.

### 6.4.4 Application: Tone Recognition

To demonstrate that the convolutional filters are capturing speech characteristics related to tones, we perform tone recognition experiments using the filter response as features. With a tone recognizer, we can not only detect mispronunciation but also provide diagnosis feedback on the tones. Tone information is encoded primarily in the fundamental frequency (F0) in speech. Early work adopted an HMM-based framework to model the F0 transition of tones [115]. Subsequent work has explored the use of both prosodic and spectral features and
applied various machine learning techniques. For example, Lei et al. [64] adopt a dynamic Bayesian network using MFCCs and pitch features. More recently, Tong et al. [99] train SVM classifiers based on a set of confidence measures computed from a DNN-based speech recognizer.

The case of tone recognition for CAPT is special, since the underlying pronunciations of the words are known. The above filter analysis motivates us to take advantage of the set of filters as feature extractors for tone recognition. As a result, we build another CNN model for surface tone classification. The architecture of the tone classifier is basically the same as the proposed CNN architecture for mispronunciation detection shown in Fig. 6-2, which has one convolutional layer and one max-pooling layer followed by one MLP layer and a softmax layer for classification. The only difference is that the softmax layer is changed from two-class to five-class (i.e., five tones). While training, we initialize the filter weights using the weights in the CNN learned from the previous task, and the filter weights are kept fixed during the training process. Only the weights related to the MLP layer and the softmax layer are updated during training. In this way, the filters can be viewed as a fixed set of feature extractors, and we show that the tone recognizer learned in this way performs better than a tone recognizer learned without the pre-trained filter weights. Our approach is similar to the idea of transfer learning [50]. When there is not enough data in the target domain, transfer learning suggests that we borrow data from another domain to learn a set of initial weights and then use the limited amount of in-domain data to fine tune the weights.

**Experimental setting**

We use the same data augmentation technique to prepare the training data by modifying the input texts. Given a set of nonnative words labeled with their surface pronunciations, for every word, we generate five training samples by pairing the speech with text in five different tones. Fig. 6-10 shows an example of how modifying the input text into four main tones affects the filter response. While the speech input (i.e., surface pronunciation) in the four matrices are all the same (SH-AN1), they are paired with different text inputs (i.e., underlying pronunciation), and thus, the outputs from the filters are different. For example, filter 58 is highly activated only for underlying tone 1, filter 34 has zero response for underlying tone 2,
Figure 6-10: An example of how pairing different texts with the same speech input generates different filter response.
and filter 13 is highly activated with respect to underlying tone 4. As a result, this data generation method can effectively create different training samples for the same surface tone class.

We use the same set of parameters as in Table 6.2 to train the model. The results are evaluated based on tone prediction accuracy. The proposed tone detector is compared with a baseline, which is a tone classifier based on the same CNN architecture while trained on the same training set directly from scratch. In other words, the latter does not have a set of pre-trained filters that serve as feature extractors.

Results

Fig. 6-11 shows the results. As a model without the pre-trained filters has a much larger parameter set to optimize, it requires a larger number of training samples. As a result, given
Table 6.5: Confusion matrix between different surface tone groups from the model with the highest prediction accuracy.

<table>
<thead>
<tr>
<th></th>
<th>tone 1</th>
<th>tone 2</th>
<th>tone 3</th>
<th>tone 4</th>
<th>tone 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ground</td>
<td>79.8%</td>
<td>8.1%</td>
<td>7.9%</td>
<td>8.0%</td>
<td>3.6%</td>
</tr>
<tr>
<td>truth</td>
<td>7.0%</td>
<td>70.3%</td>
<td>18.1%</td>
<td>2.8%</td>
<td>3.8%</td>
</tr>
<tr>
<td>tone 3</td>
<td>3.2%</td>
<td>14.0%</td>
<td>61.9%</td>
<td>3.1%</td>
<td>3.5%</td>
</tr>
<tr>
<td>tone 4</td>
<td>8.4%</td>
<td>5.2%</td>
<td>9.8%</td>
<td>84.4%</td>
<td>5.4%</td>
</tr>
<tr>
<td>tone 5</td>
<td>1.5%</td>
<td>2.4%</td>
<td>2.3%</td>
<td>1.7%</td>
<td>83.6%</td>
</tr>
</tbody>
</table>

the same amount of training data, the model with pre-trained filters consistently outperforms the baseline. Table 6.5 shows the prediction accuracy of the five tone groups from the best model, the model with pre-trained filters and trained on full nonnative training set. We can see that surface tone 3 has the lowest accuracy (61.9%). Tone 3 being the hardest to detect may be due to not only its relatively complex pitch contour but also the relatively fewer number of training samples available (taking up 15.7%, compared with 23.7% for tone 1, 19.8% for tone 2 and 32.7% for tone 4), since tone 3 is harder for nonnative students to produce to begin with. Moreover, contextual effects such as third tone sandhi [24] also have impact on the actual realization of the pitch patterns. On the other hand, tone 5 is a special case. While it has the fewest number of training samples (taking up only 8.0%), the prediction accuracy is high (83.6%). Since it is an unstressed syllable, there may be some other acoustic cues, such as shorter duration, learned by the MLP layer. In conclusion, the above experiment demonstrates that the filters trained in a word-level binary mispronunciation detection task capture human-interpretable sub-word level speech characteristics, which is tone in this case.

### 6.5 Summary

In this chapter, we presented a CNN model for mispronunciation detection. As CNN models have been shown to be capable of learning feature representations from raw data in an unsupervised manner, this is our proposed solution to the challenge that classifier-based approaches to mispronunciation detection typically rely on a large set of human-engineered features extracted based on expert knowledge.

In the first part, we propose a data augmentation method that transforms native speech
into a training set for training a binary classifier. The method is language-independent such that it modifies the binary text feature vectors without any prior knowledge. Experimental results show the effectiveness of the data augmentation method and the power of adopting CNN models over non-deep learning models such as SVM. Moreover, we show that model adaptation with less than 2.4hr of nonnative speech can greatly improve the system’s performance.

In the second part, we visualize the automatically learned filters by building filter activation maps. Both the filter activation maps built based on underlying pronunciations and based on surface pronunciations are examined. We perform both qualitative study and quantitative study. From the filter activation map built based on the underlying pronunciations, we can see that the filters automatically learn that a group of underlying Pinyin labels share common acoustic characteristics, which is tone in this case. From the filter activation map built based on the surface pronunciations, we find that the filters capture the surface tone patterns embedded in the speech signals. This clustering phenomenon is closely related to the error patterns in nonnative speech. We also perform a tone classification experiment using the pre-trained filters as feature extractors. The results provide strong evidence that the filters are indeed capturing sub-word level speech characteristics related to error patterns despite being initially trained with only binary word-level labels.
Chapter 7

Conclusions and Future Work

In this chapter, we summarize the contributions of this thesis and discuss possible future work.

7.1 Summary and Contributions

In this thesis, we focus on developing computer-assisted pronunciation training (CAPT) systems and propose solutions to tackle the challenge of the mismatch between native and nonnative speech.

In the first part of the thesis, we take up the first challenge in developing CAPT systems: limited amounts of nonnative training data. A mispronunciation detection system that does not rely on nonnative training data is proposed. The system consists of two stages: personalized unsupervised error pattern discovery, and error pattern decoding. Inspired by the success of unsupervised acoustic pattern discovery techniques, the first stage of the system computes a DTW distance between speech segments from an individual learner, and generates a set of error candidates by identifying abnormally close phoneme pairs. Motivated by a pronunciation pattern analysis at an individual learner level, the second stage of the system imposes phonetic context constraints during the decoding process. In the end, a mispronunciation detection system can be built without expert knowledge on students’ L1s, and thus, the system is L1-independent. Moreover, the system can be portable to different target L2s as long as there is a speech recognizer available.
We perform mispronunciation detection experiments on two corpora, the CU-CHLOE corpus and the iCALL corpus. They differ in not only their L2s (English vs. Mandarin Chinese) and L1s (Cantonese and Mandarin Chinese vs. European languages), but also the amount of per-speaker data (small vs. large) and students’ proficiency levels (advanced vs. beginners). Therefore, our experiments cover a wide range of possible learners’ backgrounds. Experimental results show the effectiveness of the proposed unsupervised error pattern discovery process in capturing an individual learner’s pronunciation patterns, and the capability of the error decoding process in prioritizing feedback. As a result, the proposed framework not only resolves the technical challenge of applying ASR in CAPT systems but also enhances their pedagogical value.

In the second part of the thesis, we tackle the second challenge in developing CAPT systems: dependency on human-engineered features. We investigate the use of CNN models for mispronunciation detection. The model takes speech and text as input and makes a binary decision on the correctness of the input speech. Experimental results on predicting word-level mispronunciations in nonnative Mandarin Chinese demonstrate the feasibility of the data augmentation technique we propose to utilize only native speech for model training. Visualization and quantitative analysis on the automatically learned filters show that though trained with binary word-level labels, the CNN model captures the sub-word level tone characteristics in nonnative Mandarin Chinese, which is the main source of pronunciation errors. The filters can be viewed as a set of feature extractors that can be jointly optimized during the training process. No feature engineering is needed, and thus, no expert knowledge in how nonnative pronunciations deviate from native pronunciations is needed. As a result, we believe the proposed method also has the potential to be applied to different L2s.

7.2 Future Work

There are several directions that would be interesting to extend the current work.
7.2.1 Error Pattern Discovery in Spontaneous Speech

In this thesis, we have been focusing on read speech. The assumption is that the CAPT system will provide a set of pre-defined scripts for the students to read, and the student will follow the given script as much as possible. As a result, the system knows what the canonical pronunciations should be, and the scripts can be used for forced alignment. However, as the students improve, it would be better to let them speak freely and create sentences on their own as opposed to reading a given text for more advanced pronunciation learning activities.

Previous work on assessing nonnative spontaneous speech usually focus on holistic scoring. Up to now, there is limited work addressing the issue of assessing segmental pronunciation errors of spontaneous speech. In order to achieve a text-independent system, the challenge would be to recognize nonnative speech in the first place. Chen et al. [23] developed a two-stage method where the nonnative speech is first recognized and then the recognized text is used to perform forced alignment for pronunciation assessment. The acoustic model they use for recognition contains 30 hours of fully transcribed nonnative speech. Moustroufas and Digalakis [73] propose running recognition using two acoustic models: one trained from native speech of students’ L1 (Greek) and one trained from native speech of the target language (English), and the two recognition results are combined.

On the basis of the framework we proposed in this thesis, one possible solution is to iteratively perform recognition, forced alignment for phonemic error patterns decoding and update the lexicon. The idea is similar to data-driven lexicon pronunciation learning [70]. As the recognition result converges, we believe the common error patterns can also be discovered in spontaneous speech without the need for transcribed nonnative data or the knowledge of students’ L1s.

7.2.2 More Sophisticated Deep Learning Models

In Chapter 6, we have demonstrated the power of deep learning for mispronunciation detection by focusing on CNNs. Numerous variations of deep learning models, such as highway models and grid LSTMs, have been proposed for ASR. As the development of CAPT systems is closely related to the advancement of ASR, it is worth examining how these models
One variation that seems promising is neural attention models. First proposed for the task of machine translation [15], the attention-based framework automatically learns a set of weights over a source sentence emphasizing the parts that are relevant to predicting a target word. Later the attention mechanism has also been explored for end-to-end speech recognition [29]. Adopting an attention-based model for mispronunciation detection may generate a set of latent weights reflecting the sub-word segments that are problematic without the need of training data with phone-level annotations.

To investigate the use of attention models for the task of mispronunciation detection, two possible ways of incorporating the attention mechanism are shown in Fig. 7-1. Instead of concatenating all the outputs from the max-pooling layer as the input to the MLP layer, a set of attention weights, \( \alpha_i \)'s, can be added on top of the outputs. Assume the outputs from the max-pooling layer is an \( K \times T \) matrix, where \( K \) is the number of filters, and \( T \) is the duration in time after downsampling via max-pooling. The output matrix can be represented as either \([t_1 \, t_2 \, \cdots \, t_T]\), where \( t_i \)'s are column vectors representing all filters’ response at a single time step, or \([f_1; f_2; \cdots; f_K]\), where \( f_i \)'s are row vectors representing a single filter’s response after max-pooling. One possibility is to apply weights along the time axis and
compute the weighted sum on \( t_i \)'s to form a single vector as the input to the MLP. The idea is that the latent weights may be able to capture which sub-word segment is mispronounced. The second possibility is to apply different weights to different filters and compute the weights sum on \( f_i \)'s. The idea is that perhaps more weights should be put on several filters. The weights can be learned through a separate MLP layer, which takes \( t_i \) or \( f_i \) as input and generates \( \alpha_i \) as output [47]. After the model training is done, analysis on the weights should provide useful information for corrective feedback.

### 7.2.3 Towards a Complete CAPT System

In Chapter 2, we have introduced CAPT as an interdisciplinary research area. Building a CAPT system requires knowledge in not only ASR but also user interface and content design. While there has been research work focusing on creating a generic platform for CAPT developers [87], many aspects still have to be taken into consideration during the design.

We have built a prototype for a web-based English CAPT system using the Spoke framework [91]. The system provides scripts for students to read (Fig. 7-2(a)), and the students can record and listen to their voice until satisfied. Once they are done recording, the system carries out personalized mispronunciation detection and returns a list of results based on their priority. The syllables that are recognized as mispronounced are highlighted, and a speech synthesis module is added for audio feedback (Fig. 7-2(b)).

The system is our initial attempt towards a personalized CAPT system. More efforts can be made on improving the completeness of the system. For example, on the front-end user experience side, the system can allow learners to edit their own reading materials to make the learning experience truly personalized. On the back-end speech component side, prosodic analysis can be added to make the pronunciation assessment more complete. Moreover, engineering efforts have to be made on speeding up the iterative error decoding process.

With a more mature system, a user study can be carried out to examine the effectiveness of the system in helping students learn. In addition, it may be possible to apply the framework on massive open online course (MOOC) platforms. MOOC platforms are popular nowadays.
Figure 7-2: A prototype of an English CAPT system. (a) Students can read the provided scripts, record and listen to their voice repeatedly until satisfied. (b) The system provides a list of detected errors with the mispronounced segments highlighted. The students can listen to their own pronunciation and compare with the correct pronunciation produced by the speech synthesis module.
as a way of personalized learning. However, to date there are only few language learning classes available, while none of them have ASR capability to the best of our knowledge due to the heterogeneous backgrounds of the learners. A complete CAPT system powered by our proposed personalized L1-independent mispronunciation detection framework can potentially serve as a solution to language learning at the MOOC scale.

7.2.4 Beyond Nonnative Speech: Disordered Speech and Dialects

The mismatch between native and nonnative speech is only one example of the domain mismatch problem that is common in ASR. The unsupervised error pattern discovery approach we proposed takes advantage of the high consistency of an individual learner’s pronunciation patterns. The idea can be applied to other domains with similar characteristics.

One possible domain is disordered speech. The mismatch between healthy speech and disordered speech is very challenging. For example, dysarthria is a motor speech disorder resulting from impaired movement of the muscles used for speech production, such as the lips, tongue and vocal folds [2]. While it is commonly acknowledged that dysarthric speech has slow speaking rate, monoloudness and mono-pitch, the actual articulation of the phonemes varies across individual patients. Similar to our analysis on pronunciation patterns on an individual learner level, data analysis on dysarthric speech can also be done to first confirm whether the same phenomenon can be observed. If there also exists high consistency under the same phonetic contexts for a single patient, the unsupervised error pattern discovery process can also be carried out to discover sounds that are problematic for a particular patient. The results can be used not only for recognizing dysarthric speech but also for aiding speech pathology treatments.

Another possible domain is dialects. Dialects can be characterized by speech patterns specific to a region. In ASR, it is necessary to adapt models to overcome the mismatch, or difference, between a national standard language and a regional dialect. Chen et al. [26] proposed a framework that automatically discovers dialect-specific phonetic rules at both the phonetic and acoustic level in a supervised manner. We may be able to view the difference as a type of mispronunciations, and apply the proposed technique to discover pronunciation patterns in an unsupervised manner to complement the existing supervised methods.
The inferred pronunciation variations can be incorporated into a lexicon for dialectal speech recognition.
Appendix A

Symbols for English Phonemes

Table A.1 lists the ARPAbet symbols and their corresponding IPA symbols for the English phonemes used in the TIMIT corpus. Table A.2 lists the 39 TIMIT phoneme groups used in the experiments.
<table>
<thead>
<tr>
<th>ARPAbet</th>
<th>IPA</th>
<th>Example</th>
<th>ARPAbet</th>
<th>IPA</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa</td>
<td>a</td>
<td>lock</td>
<td>ix</td>
<td>i</td>
<td>roses</td>
</tr>
<tr>
<td>ae</td>
<td>iæ</td>
<td>bat</td>
<td>iy</td>
<td>iy</td>
<td>beat</td>
</tr>
<tr>
<td>ah</td>
<td>A</td>
<td>but</td>
<td>jh</td>
<td>j</td>
<td>judge</td>
</tr>
<tr>
<td>ao</td>
<td>A</td>
<td>bought</td>
<td>k</td>
<td>k</td>
<td>kit</td>
</tr>
<tr>
<td>aw</td>
<td>α^w</td>
<td>down</td>
<td>kcl</td>
<td>-</td>
<td>k closure</td>
</tr>
<tr>
<td>ax</td>
<td>æ</td>
<td>about</td>
<td>l</td>
<td>l</td>
<td>let</td>
</tr>
<tr>
<td>ax-h</td>
<td>æʰ</td>
<td>suspect</td>
<td>m</td>
<td>m</td>
<td>met</td>
</tr>
<tr>
<td>axr</td>
<td>œ</td>
<td>butter</td>
<td>n</td>
<td>n</td>
<td>net</td>
</tr>
<tr>
<td>ay</td>
<td>œ</td>
<td>buy</td>
<td>ng</td>
<td>ng</td>
<td>sing</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>bet</td>
<td>nx</td>
<td>nx</td>
<td>winner</td>
</tr>
<tr>
<td>bcl</td>
<td>-</td>
<td>b closure</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ch</td>
<td>c</td>
<td>church</td>
<td>ow</td>
<td>ow</td>
<td>boat</td>
</tr>
<tr>
<td>ch</td>
<td>c</td>
<td>church</td>
<td>oy</td>
<td>oy</td>
<td>boy</td>
</tr>
<tr>
<td>d</td>
<td>d</td>
<td>debt</td>
<td>p</td>
<td>p</td>
<td>pet</td>
</tr>
<tr>
<td>dcl</td>
<td>-</td>
<td>d closure</td>
<td>pau</td>
<td>pau</td>
<td>pause</td>
</tr>
<tr>
<td>dh</td>
<td>d</td>
<td>that</td>
<td>pcl</td>
<td>pcl</td>
<td>p closure</td>
</tr>
<tr>
<td>dx</td>
<td>r</td>
<td>batter</td>
<td>q</td>
<td>q</td>
<td>glottal stop</td>
</tr>
<tr>
<td>eh</td>
<td>ε</td>
<td>bet</td>
<td>r</td>
<td>r</td>
<td>rent</td>
</tr>
<tr>
<td>el</td>
<td>l</td>
<td>battle</td>
<td>s</td>
<td>s</td>
<td>sat</td>
</tr>
<tr>
<td>em</td>
<td>m</td>
<td>bottom</td>
<td>sh</td>
<td>sh</td>
<td>shut</td>
</tr>
<tr>
<td>en</td>
<td>n</td>
<td>button</td>
<td>t</td>
<td>t</td>
<td>ten</td>
</tr>
<tr>
<td>eng</td>
<td>n</td>
<td>Washington</td>
<td>tcl</td>
<td>tcl</td>
<td>t closure</td>
</tr>
<tr>
<td>epi</td>
<td>-</td>
<td>epenthetic silence</td>
<td>th</td>
<td>th</td>
<td>thing</td>
</tr>
<tr>
<td>er</td>
<td>r</td>
<td>bird</td>
<td>uh</td>
<td>uh</td>
<td>book</td>
</tr>
<tr>
<td>ey</td>
<td>e</td>
<td>bait</td>
<td>uw</td>
<td>uw</td>
<td>boot</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>fat</td>
<td>ux</td>
<td>ux</td>
<td>too</td>
</tr>
<tr>
<td>g</td>
<td>g</td>
<td>get</td>
<td>v</td>
<td>v</td>
<td>vat</td>
</tr>
<tr>
<td>gcl</td>
<td>-</td>
<td>g closure</td>
<td>w</td>
<td>w</td>
<td>wit</td>
</tr>
<tr>
<td>hh</td>
<td>h</td>
<td>hat</td>
<td>y</td>
<td>y</td>
<td>you</td>
</tr>
<tr>
<td>hv</td>
<td>h</td>
<td>ahead</td>
<td>z</td>
<td>z</td>
<td>zoo</td>
</tr>
<tr>
<td>ih</td>
<td>i</td>
<td>bit</td>
<td>zh</td>
<td>zh</td>
<td>pleasure</td>
</tr>
<tr>
<td>h#</td>
<td>-</td>
<td>utterance boundary</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table A.1: Mapping between the ARPAbet and IPA symbols of the English phonemes used in this thesis (adapted from [9]).
<table>
<thead>
<tr>
<th>Phoneme groups</th>
<th>Phoneme groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>1   aa ao</td>
<td>20  k</td>
</tr>
<tr>
<td>2   ae</td>
<td>21  l el</td>
</tr>
<tr>
<td>3   ah ax ax-h</td>
<td>22  m em</td>
</tr>
<tr>
<td>4   aw</td>
<td>23  n en nx</td>
</tr>
<tr>
<td>5   ay</td>
<td>24  ng eng</td>
</tr>
<tr>
<td>6   b</td>
<td>25  ow</td>
</tr>
<tr>
<td>7   ch</td>
<td>26  oy</td>
</tr>
<tr>
<td>8   d</td>
<td>27  p</td>
</tr>
<tr>
<td>9   dh</td>
<td>28  r</td>
</tr>
<tr>
<td>10  dx</td>
<td>29  s</td>
</tr>
<tr>
<td>11  eh</td>
<td>30  sh zh</td>
</tr>
<tr>
<td>12  er axr</td>
<td>31  t</td>
</tr>
<tr>
<td>13  ey</td>
<td>32  th</td>
</tr>
<tr>
<td>14  f</td>
<td>33  uh</td>
</tr>
<tr>
<td>15  g</td>
<td>34  uw ux</td>
</tr>
<tr>
<td>16  hh hv</td>
<td>35  v</td>
</tr>
<tr>
<td>17  ih ix</td>
<td>36  w</td>
</tr>
<tr>
<td>18  iy</td>
<td>37  y</td>
</tr>
<tr>
<td>19  jh</td>
<td>38  z</td>
</tr>
<tr>
<td>39  sil bcl dcl epi gcl h# kcl pau pcl tcl</td>
<td></td>
</tr>
</tbody>
</table>

Table A.2: The 39 TIMIT phoneme groups used in the experiments. The phoneme symbols that are chosen to represent the whole group are shown in bold.
Appendix B

Mapping from Standard Pinyin to Initial and Final Pinyin Units

Pinyin is the official romanization system for Standard Chinese. In this thesis, we define a set of initials and finals shown in Table 5.2 as the target Pinyin units for mispronunciation detection. Below we describe the mapping from standard Pinyin to the set of initials and finals that we defined.

1. General rule: [initial][final] → [initial] + [final]

The general rule applies to Pinyin words that do not start with y or w, and Pinyin words that do not end with u, ü, ue, üe, uan, un. For example, ba → b + a, zhou → zh + ou, and jiong → j + iong.

2. Pinyin words starting with y (represented as y[X]):

   • If [X] starts with i: y[X] → [X]. For example, yi → i and ying → ing.
   • If [X] starts with u: y[X] → [X] with u substituted with v. For example, yu → v, yue → ve, yuan → van, and yun → vn.
   • If [X] does not start with i or u: y[X] → i[X]. For example, ya → ia, ye → ie, and yong → iong. There exists one special case, you → iu.

3. Pinyin words starting with w (represented as w[X]): w[X] → u[X]

   For example, wa → ua and wang → uang. The rule can be applied to all Pinyin words
starting with \( w \) except \( wu \rightarrow u \), \( wei \rightarrow ui \), and \( wen \rightarrow un \).

4. Pinyin words containing \( ù \): \([\text{initial}]ù[X] \rightarrow [\text{initial}] + v[X]\)
   For example, \( nù \rightarrow n + v \) and \( lüè \rightarrow l + ve \).

5. Pinyin words ending with \( u \), \( ue \), \( uan \), \( un \): \([\text{initial}]u[X] \rightarrow [\text{initial}] + v[X]\)
   For example, \( ju \rightarrow j + v \), \( xue \rightarrow x + ve \), and \( qun \rightarrow q + vn \).
Bibliography


