An Error Detection and Correction Framework to Improve Large Vocabulary Continuous Speech Recognition

ZHOU, Zhengyu

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy

In

System Engineering and Engineering Management

August 2009

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

To my loving family

Acknowledgments

First, I thank my thesis advisor, Prof. Helen Meng. She has been very supportive throughout this research. She let me choose research topics that interest me, gave me insightful advice, and helped me improve both my writing technique and my presentation skills. Her trust and encouragement played an important role in increasing my confidence. Her understanding and patience gave me the strength to face difficulties during this period.

I owe enormous gratitude to Dr. Frank K. Soong at Microsoft Research Asia. Dr. Soong gave me the opportunity to perform many speech recognition experiments in addition to my formal assignment of audio retrieval during my internships. These experiments are the basis of this work. He has also offered me valuable suggestions and continuous encouragement throughout my Ph.D. study.

I would like to express deep gratitude to Dr. Jianfeng Gao in Microsoft Research Redmond. Since 2002, Dr. Gao has helped me greatly in language modeling, ranging from trigram modeling with heterogeneous corpora to discriminative n-gram modeling. Discussions with him have always been a great source for ideas. Without his guidance and support, the discriminative training part of this research could not have been performed so smoothly.

I would further like to thank many other researchers at Microsoft who contributed to this study in various ways. Thanks to Dr. Ye Tian for his help in acoustic modeling; to Dr. Yu Shi for language modeling; to Dr. Chao Huang for error detection; to Dr. Peng Liu, Dr. Patrick Nguyen and the late Dr. Jianlai Zhou for discriminative training; and to Dr. Frank Seide and Dr. Peng Yu for decoding.

I am grateful to Dr. Wai Kit Lo and all other members of the CUHK Human-Computer Communications Laboratory for their assistance in this work. I would like to offer my special thanks to Prof. Zongge Li at Fudan University, who opened the door of spoken language processing for me, and to Dr. Eric Chang at Microsoft Research, who led me into the field of large vocabulary continuous speech recognition. Thanks are also due to the professors at Fudan University and Nanjing University who helped develop my knowledge in the fields of computer science and mathematics.

I also thank Prof. Lori Lamel from the Computer Sciences Laboratory for Mechanics and Engineering Sciences (LIMSI), Prof. Qiang Huo from Microsoft Research Asia, Prof. Yu Xu and Prof. Kam-Fai Wong from The Chinese University of Hong Kong for their precious comments regarding this thesis.

Finally, I would like to deeply thank my husband, parents and friends. It is your love and support that made this work possible.

This project was partially supported by the HKSAR government under Central Allocation CUHK1/02C and also affiliated with the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies. The development of the baseline recognizer, all the decoding and the experiments on discriminative n-gram modeling were conducted in Microsoft Research Asia.

An Error Detection and Correction Framework to Improve Large Vocabulary Continuous Speech Recognition

by

Zhengyu Zhou

Submitted to the Department of Systems Engineering and Engineering Management
on August 28, 2009 in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Abstract

This thesis proposes an error detection and correction (ED-EC) framework to incorporate advanced linguistic knowledge sources into large vocabulary continuous speech recognition. Previous efforts that apply sophisticated language models (LMs) in speech recognition normally face a serious efficiency problem due to the intense computation required by these models. The ED-EC framework aims to achieve the full benefit of complex linguistic sources while at the same time maximize efficiency. The framework attempts to only apply computationally expensive LMs where needed in input speech. First, the framework detects recognition errors in the output of an efficient state-of-the-art decoding procedure. Then, it corrects the detected errors with the aid of sophisticated LMs by (1) creating alternatives for each detected error and (2) applying advanced models to distinguish among the alternatives. In this thesis, we implement a prototype of the ED-EC framework on the task of Mandarin dictation. This prototype detects recognition errors based on generalized word posterior probabilities, selects alternatives for errors from recognition lattices generated during decoding and adopts an advanced LM that combines mutual information, word trigrams and POS trigrams. The experimental results indicate the practical feasibility of the ED-EC framework, for which the optimal gain of the focused LM is theoretically achievable at low computational cost. On a general-domain test set, a 6.0% relative reduction in character error rate (CER) over the performance of a state-of-the-art baseline recognizer is obtained. In terms of efficiency, while both the detection of errors and the creation of alternatives are efficient, the application of the computationally expensive LM is concentrated on less than 50% of the utterances. We further demonstrate that the potential benefit of using the ED-EC framework in improving the recognition performance is tremendous. If error detection is perfect and alternatives for an error are guaranteed to include the correct one, the relative CER reduction over the baseline performance will increase to 36.0%. We also illustrate that the ED-EC framework is robust on unseen data and can be conveniently extended to other recognition systems.

In addition to the ED-EC framework, this thesis proposes a discriminative lattice rescoring (DLR) algorithm to facilitate the investigation of the extensibility of the framework. The DLR method recasts a discriminative n-gram model as a pseudo-conventional n-gram model and then uses this recast model to perform lattice rescoring. DLR improves the efficiency of discriminative n-gram modeling and facilitates combined processing of discriminative n-gram modeling with other post-processing techniques such as the ED-EC framework.

Thesis Supervisor: Professor Helen Mei-Ling Meng

摘　要

對於大詞彙量連續語音(LVCSR)識別來說，加入高級語言知識雖然可能提高識別的準確率，但是往往會顯著降低識別的效率。 這主要是因爲高級語言模型的使用會增加識別複雜度和計算量。本論文提出一個新的框架，錯誤檢測與糾正(ED-EC)框架，來利用高級語言知識進行大詞彙量連續語音識別。該框架試圖在最大化高級語言知識帶來收益的同時，盡可能地減少計算量來保證識別系統的效率。其主要思想是在識別語音信號時，只在需要的地方使用複雜的語言模型。該框架以一個高效的當代主流語音識別器為基礎，首先檢測識別結果中的錯誤，然後使用複雜的高級語言模型來糾正檢測到的錯誤。在糾正錯誤的過程中，該框架為每個找到的錯誤建立一系列候選，然後利用高級語言模型從這些候選挑出最可能對的那一個來作糾正的結果。在本論文中，我們針對普通話的聽寫開發了一個該框架的原型。該原型基於一般化的詞后驗概率(generalized word posterior probability, GWPP)來進行錯誤檢測，它從識別過程中生成的識別網絡中提取候選，並且使用了一個結合了交互信息模型(mutual information)、詞三元模型(word trigram)、詞性三元模型(POS trigram)的高級語言模型。在理論上，ED-EC 框架在使用高級語言模型時可以用低計算量達到最佳效果。實驗證明，該框架在現實中也是可行的。在一個普通領域的測試數據集上，相對于基礎識別器的識別結果而言，普通話原型的使用把識別錯誤率降低了 6.0%。言及識別效率，該原型的錯誤檢測與建立候選的過程都是高效率的，而計算量很大的高級語言模型的使用，則集中到了不到一半的數據上。我們進一步證明了 ED-EC 框架的提升空間非常大。如果錯誤能夠被完美地檢測到並且候選中總是包含正確結果，使用原型帶來的識別錯誤率降低將會擴大到 36.0%。我們還展示了 ED-EC 框架對於未見數據具有良好的魯棒性，該框架還可以方便地延展至其他識別基綫上去。

爲了協助研究 ED-EC 框架的可延展性，我們還額外提出了一個區別性網絡重評分(discriminative lattice rescoring, DLR)算法。該算法把區別性 n 元模型轉化成傳統 n 元模型的形式，然後進行網絡重評分。該算法在兩個方面提高了區別性 n 元建模技術。它提高了使用區別性 n 元模型的效率。同時，它方便了區別性 n 元建模與其他技術的結合使用。

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis focuses on large vocabulary continuous speech recognition (LVCSR), the purpose of which is to automatically transcribe natural speech. LVCSR technology has many uses. For example, it has been applied to facilitate communication between computers and users. Speech-based computer-human interfaces not only have high usability but can also enhance human productivity; people can control machines by talking to them, which is especially useful for hands-busy/eyes-busy applications. In addition, LVCSR is a key component of automatic speech translation (AST), which is meant to help remove the language barriers between people in cross-national communications. For speech translation, reliable speech transcription (i.e., speech-to-text conversion) is the basis of effective text-to-text translation and text-to-speech synthesis. Other applications of LVCSR include audio/video retrieval and assistance of disabled people.

LVCSR has been actively studied since the 1980s, and substantial improvements have been made. However, there is still a long way to go towards developing real-time recognition systems with satisfactory performance. A promising direction to further enhance LVCSR is the incorporation of advanced linguistic knowledge sources, such as parsing knowledge and word mutual information. Modern recognizers model local word-sequence dependencies by trigram language models (LMs). Adopting more sophisticated linguistic knowledge sources can introduce better constraints in recognition and thus benefit overall performance.

Since advanced linguistic knowledge sources normally require a long context span and relatively intense computation, the application of sophisticated sources greatly increases the decoding complexity and reduces the decoding speed. Given certain sophisticated linguistic knowledge sources, identifying a suitable way to incorporate these sources into an efficient

LVCSR system becomes a challenging task. This thesis proposes an error detection and correction (ED-EC) framework to address this challenge. The ED-EC framework reduces the complexity related to advanced language models by concentrating the application of these models on signal segments where the baseline recognizer is likely to make mistakes. The framework aims to take advantage of sophisticated linguistic knowledge sources to improve recognition accuracy as much as possible, while at the same time maximizing efficiency.

Most previous efforts that incorporate sophisticated linguistic knowledge in LVCSR process speech segments indiscriminately. The selectiveness in the application of sophisticated knowledge is what differentiates the ED-EC framework from these efforts. Among the works that aim to correct recognition errors, the ED-EC framework is unique because it involves decoupling the error detection and error correction procedures. This separation leads to high flexibility in selecting detailed algorithms for error detection, error correction, or both. In this work, we have developed an error correction procedure that does not rely on recognizer-specific information. Hence, the procedure is generalizable.

In this chapter, we introduce the history of research on speech recognition in Section 1.1. Then, state-of-the-art approaches for LVCSR are described in Section 1.2. Recent progress in LVCSR is discussed in Section 1.3. Our focus of investigation and the motivation for proposing the ED-EC framework are addressed in Section 1.4 and Section 1.5, respectively. Finally, the organization of the thesis is given in Section 1.6.

1.1 History of Automatic Speech Recognition

Active research in automatic speech recognition (ASR) started in the 1950s. Since then, ASR technology has evolved from digit recognition to robust, speaker-independent, large vocabulary continuous speech recognition.

In the 1950s, researchers attempted to exploit the fundamental ideas of ASR. The first word recognizer that could recognize the ten digits by approximating the formants was built at Bell Labs in 1952 [26]. A classifier that continuously evaluated the frequency spectra of speech signals was proposed in 1958 [27]. The usage of grammar probabilities and distinctive linguistic features, such as voiced/unvoiced [33] and turbulent/non-turbulent [135], in recognition was

investigated. In the 1960s, several major breakthroughs were made, e.g., Linear Predictive Coding (LPC) [3, 58] and Dynamic Time Warping [127]. Neural networks have also been deployed for phone recognition [83]. In the 1970s, small-vocabulary isolated word recognition matured. The first ARPA (Advanced Research Projects Agency) project was established in 1971 to build speech understanding systems that could perform 1000-word connected speech recognition with a few speakers and constrained grammar. To fulfill this goal, new techniques were introduced in speech recognition. These include LPC segments, high-level syntactic/semantic models, and hidden Markov models (HMMs) [4, 78].

In the 1980s, the focus of ASR research turned to continuous speech recognition, which transcribes fluently spoken utterances. ARPA sponsored a new program to develop a speaker-independent recognition system using a 1000-word lexicon. With intense research related to the ARPA projects, the HMM approach became prevalent in speech recognition, replacing the template-based approaches that did not facilitate the incorporation of knowledge and were difficult to generalize. Within the stochastic framework of HMM, detailed modeling and decoding techniques continue to become more sophisticated [138]. Acoustic modeling evolved from simple context-independent monophone modeling to cross-word triphone modeling. For language modeling, n-gram modeling techniques, along with various smoothing algorithms, were proposed and gained wide usage [46]. Decoding strategies that improve search efficiency were designed. The capability of automatic speech recognition was greatly enhanced by the development of these technologies. Today, the lexicon size has increased to hundreds of thousand words, and spontaneous speech can also be handled.

Modern systems for speaker-independent large-vocabulary continuous speech recognition adopt statistic frameworks, including HMMs, and alternative models based on artificial neural networks [41]. Recognition accuracy depends on various aspects, such as the availability of abundant training data and the disfluency level of the focused speech. For general-domain dictation tasks in clean environments, the recognition accuracies of state-of-the-art LVCSR systems are normally higher than 90% [20, 147]. Several companies, such as IBM and Microsoft, have commercialized various products to transcribe dictation [154, 155, 156]. For those relatively difficult tasks, such as the recognition of spontaneous speech and recognition in noisy environments, the accuracies are relatively low, depending on the characteristics of the input speech. For example, on the Switchboard corpora which are telephone conversations and widely

used in research institutions, the word error rates (WERs) normally range from 20% to 40% [54, 85, 106, 124].

1.2 LVCSR Basics

State-of-the-art LVCSR can be viewed as a Bayesian decision problem. Let $X=x_1x_2..x_T$ be the sequence of acoustic observations corresponding to a speech utterance and $W=w_1w_2...w_M$ be a sequence of words into which the utterance may be transcribed. The task of speech recognition seeks the most likely word string W^* such that

$$W^* = \arg\max_W p(W \mid X) = \arg\max_W p(X \mid W)p(W) \tag{1.1}$$

where $p(X|W)$ is the acoustic likelihood determined by acoustic models (AM) and $p(W)$ is the linguistic likelihood determined by a language model (LM).

Acoustic models are usually implemented by hidden Markov models (HMMs). To model the acoustic variability and co-articulation effects, context-dependent phone units, such as triphones and quinphones, are adopted [49]. Since there are many such context-dependent units, sparseness of training data becomes a problem. A widely used approach to solve this problem is parameter tying, that is, sharing parameters for models with similar contexts [15]. Normally, maximum likelihood estimation (MLE) is utilized to iteratively estimate the parameters of the HMMs with an efficient forward-backward algorithm [55].

The most prevalent language modeling technique is n-gram modeling, which predicts the next word by estimating its conditional probability after truncating the conditioning word history to *n-1* words:

$$p(W) = \prod_{k=1}^{K} p(w_k \mid w_1, w_2, ...w_{k-1}) = \prod_{k=1}^{K} p(w_k \mid w_{k-n+1}, w_{k-n+2}, ...w_{k-1}) \tag{1.2}$$

The n-gram probabilities can be estimated from training corpora by simply counting the occurrences of word sequences. However, as n increases, data sparseness becomes a more serious problem. To overcome this issue, various smoothing approaches have been proposed that range from simple methods, such as absolute discounting, to relatively complicated ones, such as Katz smoothing [46].

Decoding is the procedure of searching for the most likely utterance among all possible utterances based on acoustics and available linguistic knowledge sources. State-of-the-art recognizers use a pre-compiled static recognition network to store all utterances along with parameters of acoustic/linguistic models [152]. The concept of this network can be illustrated in Figure 1.1. Each word in the lexicon is represented as a sequence of context-dependent phone HMMs. The n-gram probabilities $p(w_i|h)$ are assigned to the links between words as the transition probabilities. The most likely word sequence W^* is thus identified by performing a Viterbi search [55] in the network search space. Pruning is performed during the search to keep computation load and memory consumption under control. In practice, decoding is a complex design problem, especially for n-gram models with $n>2$. Since the assignment of an n-gram probability to a word depends on the previous $n-1$ words, all possible $(n-1)$-word histories for each word need to be maintained in the static recognition network. While efficient decoding algorithms have been proposed for trigrams [13, 121], performing higher-order n-gram decoding remains a challenging task.

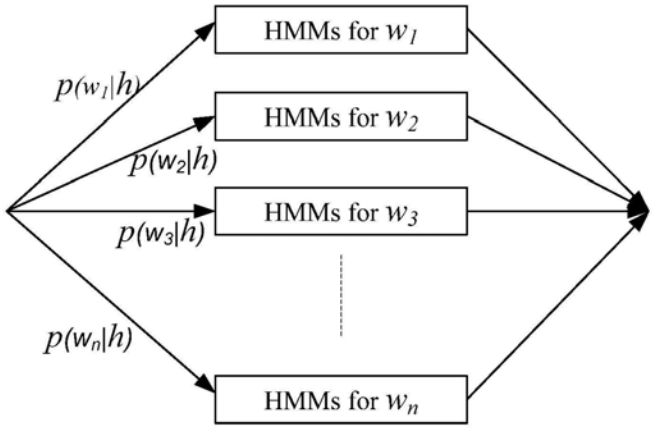

Figure 1.1: The concept of the recognition network

Given an input speech utterance, the decoder can be set to generate the recognition lattice and/or N-best hypotheses for post-processing. The recognition lattice is a compact representation of active hypotheses (i.e., those hypotheses that survive pruning during the search), as shown in Figure 1.2. In a recognition lattice generated by n-gram decoding, each word hypothesis has a unique $(n-1)$-word history that is needed to compute a unique LM likelihood. Some recognizers may merge two word hypotheses that have the same identified word and the same starting/ending times if (1) their LM likelihoods are the same and (2) merging will not cause ambiguities in the assignment of LM likelihoods for the subsequent word hypotheses. The

N-best hypotheses are the top *N* hypotheses contained in the recognition lattice, as illustrated in Figure 1.3. The recognized utterance is the top-scoring hypothesis contained in the *N*-best lists or the recognition lattice.

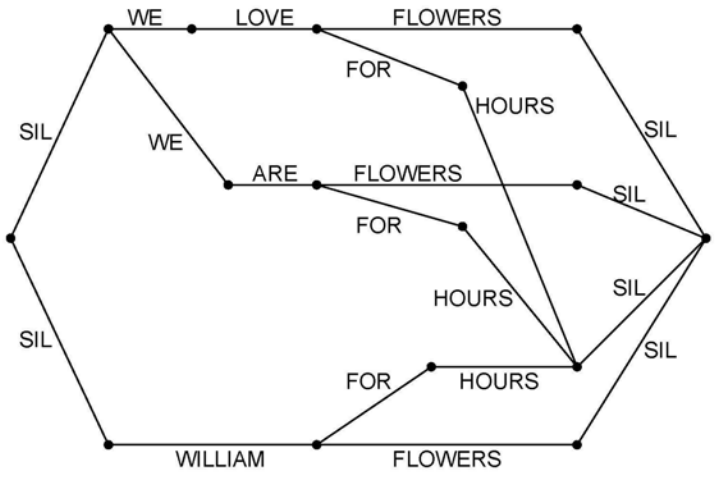

Figure 1.2: A sample recognition lattice generated by trigram decoding ("SIL" marks pauses)

Hypothesis 1: We love flowers.
Hypothesis 2: We are flowers.
Hypothesis 3: We love for hours.
...
Hypothesis *N*: William for hours.

Figure 1.3: An example of the *N*-best hypotheses

1.3 Previous Efforts to Improve LVCSR

This thesis proposes a novel framework to post-process the output of a baseline recognition system with advanced language models, as will be discussed in the subsequent sections. Theoretically, the baseline system of the framework could be any LVCSR system. In order for readers to get an overall understanding of the recent progress in LVCSR, as well as what the baseline system of the framework could be, this section briefly introduces the previous attempts

to improve state-of-the-art LVCSR. We will review the previous works that are related to the framework in details in Chapter 2.

The discriminative training technology has recently received increasing interest in LVCSR. State-of-the-art recognizers estimate the model parameters under the framework of maximum likelihood estimation. This may lead to suboptimal performance in terms of recognition error rate. Discriminative training approaches attempt to adjust the parameters with the aim of directly minimizing the recognition error rate. These approaches may bring lower recognition error rate but the discriminatively trained models tend to be less general than the models trained by maximum likelihood estimation. The discriminative training technology has been applied in different aspects of LVCSR. For acoustic modeling, various discriminative training criteria, such as maximum mutual information estimation (MMIE) and minimum classification error (MCE), have been adopted to adjust the parameters of state-of-the-art acoustic models or novel new acoustic models [8, 65, 87, 98, 137, 99]. For language modeling, n-grams have been adjusted using different discriminative training methods [20, 69, 71, 106]. For example, Roark et al. [106] adjusts n-grams discriminatively in a linear framework of *N*-best re-ranking; Kuo et al. [69] used the generalized probabilistic descent (GPD) algorithm to minimize string error rate. For the decoding procedure, algorithms have been proposed to discriminatively adjust the transition weights in the recognition network in a unified way [70, 76]. The results reported for the use of discriminative training in LVCSR show that this technology can be beneficial. For instance, training HMMs discriminatively using the Minimum Phone Error (MPE) criteria leaded to a 4.8% absolute reduction in WER on corpora of telephone conversations [99]. However, it is often observed that the effectiveness of discriminatively training relies on a good match between the training and testing conditions [147]. In addition, the discriminative training procedure can be computationally expensive, especially when iterative decoding is needed.

Modern recognizers use HMM acoustic models as the acoustic models and use the word trigram as the language model. In literature, many new acoustic/language models have been investigated. Among the new acoustic models, some models (e.g., the subspace precision and mean model (SPAM) and the extended maximum likelihood linear transformation (EMLLT) model) refine the details such as the precision matrices in the computation of the Gaussians in HMM to achieve better recognition accuracy and/or efficiency [35, 54, 97]. Some models (e.g.,

the segmental models and the switching linear dynamical models) extend the standard HMM with the aim of better modeling the dependency between successive speech frames [9, 36, 37, 53, 72, 73, 108, 120, 140]. Other models (e.g., the graphic models and template-based models) adopt novel modeling designs that are different from the HMM framework [10, 95, 128]. Encouraging LVCSR results have been obtained for some new acoustic models. For example, the EMLLT model achieves an absolute reduction in word error rate of around 1% over the baseline performance (from 23.6% to 29.1%) for the Switchboard task. On the other hand, many new acoustic models are still preliminary. Some novel models may even perform worse than the state-of-the-art acoustic HMMs, but they still have the potential to benefit LVCSR in certain way (e.g., providing complementary information) [128].

For new language models (LMs), some of them adopt Neural Network frameworks to estimate n-gram probabilities in a continuous space with the aim of solving the data sparseness problem [40, 113, 114]. Most of the new LMs attempt to capture relatively advanced linguistic knowledge sources. Note that the word trigrams in state-of-the-art recognizers only capture local word-sequence constraints. The usage of advanced LMs in LVCSR has great potential. Various advanced LMs have been proposed. Higher-order n-grams, which capture longer-distance word-sequence constraints, were used to post-process the recognizer output [85, 124]. Advanced LMs that model sophisticated syntactic/semantic knowledge sources, such as parsing information [125, 18], triggers extracted based on mutual information [107, 117], and Latent Semantic Analysis (LSA) [7, 21, 144], were also investigated. We will discuss the advanced LMs in more details in Chapter 2. Higher-level linguistic knowledge sources have been shown to be beneficial for LVCSR. For example, an advanced LM that linearly combines phonetic, lexicon, syntactic and semantic knowledge sources brought a 2.9% absolute WER reduction over the baseline performance of 30% for the Switchboard task. However, most of the advanced LMs are computational expensive, as will be discussed later. Incorporating them into LVCSR will lead to systems too slow for real-time applications. Thus, it is meaningful to find a suitable way to apply advanced LMs so that their optimal gains can be achieved at lowest computational load.

Efforts have been made to improve LVCSR in other ways. New decoding methods have been proposed to reduce memory usage and/or improve search efficiency. For example, an Order-Preserving LM Context Pre-computing (OPCP) method was proposed to reduce memory cost without slowing down the speed of LM lookup [75], an efficient likelihood ratio test

approach was developed to perform fast-matching [1] and virtual hypothesis copies were used to reduce the computational load [115]. Combining multiple recognition systems to achieve further improvement has been popular since the introduction of ROVER [31], an algorithm that aligns the recognition outputs from different systems into confusion networks and votes. Some extensions of ROVER such as *N*-best ROVER [29] and Confusion Network Combination (CNC) [29] have been proposed. Siohan et al. [118] attempted to use the randomized decision tree state tying procedure to systematically build multiple recognition systems preceding the usage of ROVER. Sankar [109] presented a Bayesian decision-theoretic algorithm, called BAYCOM, for system combination. The combination of multiple systems has been observed to be effective to further improve recognition accuracy, as long as the information provided by individual systems is complementary.

1.4 Problem Statement

State-of-the-art LVCSR systems involving HMM acoustic models and n-gram language models have been prevalent due to their effectiveness and efficiency. However, their detailed modeling techniques have evolved to a state of considerable sophistication whereby their performances are stabilizing at a local maximum [138]. Possible breakthroughs may come from various directions, such as designing new acoustic models, performing discriminative training, combining multiple systems and applying more advanced linguistic models. Among these options, the usage of advanced linguistic knowledge is especially attractive. Note that modern recognizers normally utilize trigrams which only capture local constraints. The potential benefit of incorporating higher-level information sources in LVCSR is large.

There are many works in the literature that attempt to model sophisticated linguistic knowledge to improve LVCSR, as mentioned in the previous section. However, most of these efforts face a serious efficiency problem. This is mainly due to the heavy computational load required for the application of advanced linguistic models. These models require relatively complex text processing procedures (e.g., parsing) to extract the needed information and/or intense calculations. The attempts to incorporate long-distance semantic/syntactic constraints into single-pass decoding or lattice-based post-processing may introduce additional complexity

in the search procedure and thus further decrease efficiency. The efficiency issue associated with knowledge application may be removed by enhancing the power of PCs or by inventing a novel LVCSR framework that simulates human perception. However, in the foreseeable future, this issue will have to be addressed for real-time applications. Given certain sophisticated linguistic knowledge sources, designing a suitable method to apply them in LVCSR becomes a challenging but meaningful task. The aim is to fully capture the benefit of the knowledge sources in focus to enhance recognition accuracy while maximizing efficiency.

1.5 Motivation

LVCSR is the task of transforming the input speech signals into corresponding text with the aid of various knowledge sources under a certain framework. Given a set of knowledge sources available for LVCSR, the application of these sources should consider their characteristics. When both acoustic and linguistic knowledge sources captured in LVCSR are elementary, it is attractive to adopt the island-driven approach [67] to decode speech. This island-driven approach first detects a keyword as a robust island based on acoustic similarity in continuous speech and then expands the island by verifying the neighboring words using acoustic features and linguistic rules. After the performance of hidden Markov acoustic models was raised to a higher level, bigram and trigram language models were developed to jointly work with acoustic models in a unified left-to-right HMM framework. Currently, after decades of evolution, the HMM framework with trigram models has already achieved good recognition accuracy, especially when the operating conditions are similar to the training conditions. For those utterances or speech segments where the current framework already works well, applying additional knowledge sources is unnecessary. Only those signals where efficient state-of-the-art decoding makes mistakes need to be processed by advanced knowledge sources. Based on these observations, we propose an error detection and correction (ED-EC) framework to focus the use of computationally expensive linguistic knowledge sources on correcting the errors made by a modern recognizer.

The ED-EC framework aims to only apply computationally expensive LMs where needed in the signals for LVCSR. The framework post-processes the output of a relatively efficient

baseline recognizer using two sequential procedures: error detection and error correction. The main structure of the framework is illustrated in Figure 1.4. The error detection procedure attempts to explicitly detect errors in each recognized utterance generated by the baseline recognizer. For each utterance containing one or more detected errors, the error correction procedure first expands each detected error into a candidate list of alternatives with the aim of including the correct transcription of the focused error. This results in a new search network. Then, sophisticated LMs are applied to rank the utterance hypotheses in the new search network. The candidate alternatives (e.g., Alternative *x* and Alternative *y*') in the top-ranking utterance hypothesis are the results of error correction for the target utterance. Generally speaking, the ED-EC framework attempts to use sophisticated linguistic constraints to re-decode the erroneous regions in the signals based on the surrounding utterance context.

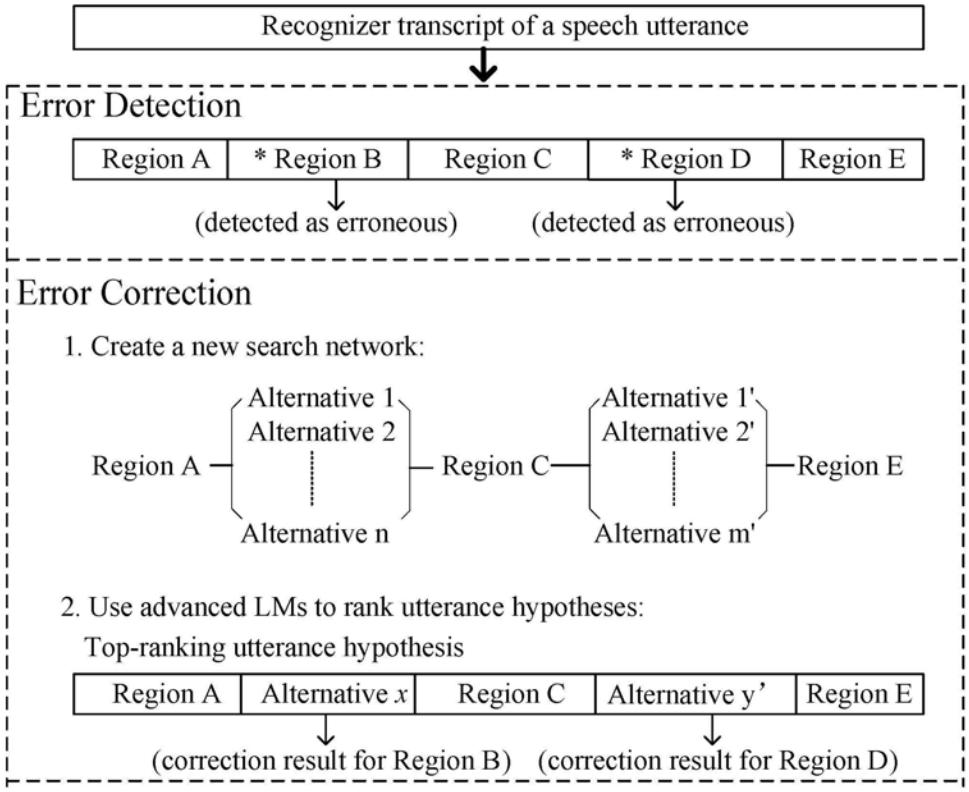

Figure 1.4: The structure of the ED-EC framework

Given a set of advanced LMs, the optimal gain of these models is theoretically achievable for the ED-EC framework. This is because the new search network will be guaranteed to contain the correct utterance hypothesis in the ideal case that (1) all errors can be correctly detected and (2) alternatives created for errors can always include the correct transcripts. In reality, both the

detection of errors and the creation of alternatives are difficult, if not impossible, to perfect. The correct utterance hypothesis may be absent in the new search network due to missed errors and/or candidate lists without correct answers. The ability of the framework to capture the benefit of focused LMs thus depends on the performance of both error detection and candidate creation.

The framework attempts to minimize the computational load of sophisticated LMs by applying these models to distinguish among alternatives only for the erroneous regions in signals. Note that the decoding search space and most post-processing search spaces (e.g., recognition lattices and *N*-best hypotheses generated during decoding) create alternatives for all signals indiscriminately. If advanced LMs are applied in such a space, these models will be used to score alternatives for correct utterances/segments. The framework aims to avoid this part of LM computation. To achieve this goal, additional computation, such as the detection of errors and the creation of alternatives, is needed. However, compared with the time saved in LM computation, the additional computational load can be relatively small or even negligible. A more detailed analysis of the framework efficiency will be provided in Section 5.3.

Given a set of computationally expensive LMs, it is suitable to use the ED-EC framework to apply these models if:

1) The baseline recognizer already provides good performance

Suppose that all errors can be correctly detected. When the baseline recognizer makes fewer errors, the framework is more efficient. In contrast, if the entire transcript for the input speech signal is erroneous, the framework actually applies sophisticated LMs to re-decode the entire speech signal.

2) An acceptable error detection method exists

The feasibility of the framework depends on the performance and efficiency of error detection. If errors cannot be detected, they cannot be corrected. If error detection is inefficient, the framework will be less efficient.

3) An acceptable candidate creation method exists

Candidate lists that include correct transcriptions are the basis for advanced LMs to correct errors. The efficiency of candidate creation as well as the sizes of candidate lists affects the overall efficiency of the framework.

The ED-EC framework is designed to be general. Both the error detection and the creation of the alternative lists for detected errors can be carried out by various techniques. It is convenient to incorporate new linguistic knowledge sources into the ED-EC framework.

The ED-EC framework has been proposed to improve the performance of a recognizer. However, the framework can also be applied to improve the performance of a multi-pass recognition system that already uses certain techniques (e.g., lattice rescoring) to post-process the output of a recognizer. In other words, the baseline recognition system of the ED-EC framework is not necessary to be a recognizer. Theoretically, any LVCSR systems can serve as the baseline recognition system for the framework.

1.6 Thesis Organization

In this work, we develop an initial prototype of the ED-EC framework for Mandarin LVCSR. The prototype attempts to detect the recognition errors in the output of a state-of-the-art baseline recognizer and then corrects the detected errors with the aid of an advanced language model that captures long-distance semantic constraints and local syntactic constraints. Based on this prototype, we aim to investigate the feasibility of the ED-EC framework, as well as the framework characteristics (e.g., factors that influence the capability of error correction, performance upper bounds and computational efficiency). To evaluate the effectiveness of the framework across various baseline recognition systems, we also analyze the performances of the ED-EC prototype on the multi-pass recognition baseline systems that use a discriminative lattice rescoring (DLR) technique to post-process the output of the recognizer. The DLR technique that we propose in this thesis is another contribution of this thesis. We refer to the multi-pass recognition baseline systems as discriminatively enhanced baseline systems, and to the original recognition baseline system of the state-of-the-art recognizer as the recognizer baseline system.

The remaining chapters are organized as follows.

Chapter 2 discusses the previous works related to the ED-EC framework. The previous efforts that inspired us to propose the framework are presented. Approaches that are relatively

similar to the framework are discussed and compared with the framework. Methods that are potentially useful in different aspects (e.g., error detection) of the framework are introduced.

Chapter 3 presents the initial prototype of the ED-EC framework for Mandarin in detail. The error detection procedure detects erroneous words and erroneous characters sequentially in an incremental way. For the error correction procedure, a two-part algorithm to correct detected errors along with an additional mechanism to handle the false alarms of error detection is proposed.

Chapter 4 describes the experiments to evaluate the ED-EC framework. A state-of-the-art recognizer is developed as the baseline recognition system. Based on this baseline system, the error detection and error correction procedures of the ED-EC framework are trained sequentially. The performance of the individual procedures and the overall framework performance are evaluated.

Chapter 5 analyzes the effectiveness of the ED-EC framework. Various factors that influence the error correction effect are investigated, including the selection of knowledge sources, the characteristics of search space and the usage of an additional mechanism to handle false alarms. Based on the observations, an equation is introduced to describe the overall effectiveness of the framework. The performance upper bounds and computational efficiency of the framework are also discussed.

Chapter 6 compares the ED-EC framework with previous approaches that utilize advanced linguistic knowledge sources in LVCSR. These previous approaches either incorporate the advanced models into single-pass decoding or apply those models in post-processing (e.g., *N*-best re-ranking). We analyze the differences between the ED-EC framework and each type of previous work. A detailed competitive analysis of computational expense is performed.

Chapter 7 proposes the DLR technique that will be used to provide discriminatively enhanced baseline systems for the ED-EC framework. DLR is an extension of discriminative n-gram modeling. We first briefly review discriminative n-gram language modeling. We prove that the discriminative n-gram model defined in a linear framework can be represented as a pseudo-conventional n-gram model under certain conditions. The DLR algorithm that uses the pseudo-conventional n-gram model to rescore recognition lattices is then discussed.

Chapter 8 evaluates the performance of the ED-EC framework on the discriminatively enhanced baseline systems. We post-process the output of the state-of-the-art recognizer with DLR. This multi-pass recognition procedure is then used as the new baseline recognition system (i.e., the discriminatively enhanced baseline system) for the framework. The effectiveness of the framework across various baseline recognition systems is analyzed by comparing the framework performances on the recognizer and discriminatively enhanced baseline systems.

Chapter 9 summarizes this thesis and discusses future research directions for the ED-EC framework. Our future work on discriminative n-gram modeling is also mentioned.

Chapter 2

Related Work

This chapter reviews the previous work related to the error detection and correction (ED-EC) framework proposed in Section 1.5. The ED-EC framework attempts to use advanced language models to correct errors that are detected in the output of a baseline LVCSR system. In this chapter, Section 2.1 reviews the previous efforts in error correction. We first introduce the context-sensitive spelling correction technology that initially inspired us with the idea for the ED-EC framework. We then discuss previous error correction efforts for LVCSR and compare the ED-EC framework with those methods that have relatively similar concepts. Section 2.2 describes previous works in the detection of recognition errors. Error detection for automatic speech recognition has been intensely investigated. Many different approaches have been proposed and have the potential to be used in the ED-EC framework to detect recognition errors. Section 2.3 compares the search spaces that were used for the application of advanced knowledge sources in the literature. For the ED-EC framework, the advanced language models are applied in a sausage-type search space, which is generated by creating candidate lists of alternatives for detected errors. Finally, Section 2.4 discusses the advanced language models that have been proposed for different tasks. These language models are potentially useful for the ED-EC framework.

2.1 Correction of Written/Recognition Errors

The ED-EC framework is inspired by the context-sensitive spelling correction technology [45, 64, 93] which attempts to correct written errors that result in valid, though unintended/misused words (e.g., *quiet* and *quite*; *among* and *between*) [45]. This technology predefines a collection of confusion sets such as {quiet, quite} and {among, between}. During the processing of a particular utterance, it views an occurrence of a word in one of the confusion sets as a potential error and attempts to verify/correct this potential error. In the verification/correction process, one or more linguistic knowledge sources are applied to distinguish among the words in the corresponding confusion set based on the context information. Linguistic knowledge sources that have been investigated in context-sensitive spelling correction include word trigrams [86], part-of-speech (POS) trigrams [45, 93], latent semantic analysis [64] and rules/features that capture the semantic/syntactic patterns in the neighborhood of the potential errors [45].

It is difficult, if not impossible, to directly apply the context-sensitive spelling correction technology in the field of LVCSR. The difficulty lies in the predefinition of confusion sets for potential errors in speech recognition. The decoding of speech is very complicated. Many acoustic and/or linguistic factors such as acoustic environment variation and speaking style variation can lead to recognition errors. A word may be recognized as various words under different situations. This makes the confusions between words inconsistent and hard to group into static confusion sets. The ED-EC framework circumvents this problem by (1) dynamically detecting recognition errors and (2) dynamically creating a confusion set (i.e., a candidate list of alternatives) for each detected error. By doing so, the framework adapts the context-sensitive spelling correction to speech recognition error correction.

There have been previous studies on error correction for LVCSR. Generally speaking, all algorithms that attempt to reduce recognition error rates can be viewed as error correction methods. Here we focus on those works that aim explicitly to correct recognition errors. These works can be categorized into two types: (1) methods to correct misrecognitions caused by a known mismatch between the training and testing conditions and (2) methods to correct misrecognitions caused by the defects of the recognizer.

The first type of error correction effort attempts to correct the recognition errors that regularly happen when the testing conditions (e.g., domain, acoustic environment) are known to differ from the training conditions in a certain way. For example, when a general-domain recognizer is applied to the medical domain, the word "cephalosporin" will always be misrecognized if this word is not included in the recognizer lexicon. Most of these works view the baseline recognizer as a "black box" and transfer each recognized utterance into a new utterance hypothesis through a channel model, which can either be a simple word-to-word transition model [102] or a "fertility" model that maps *n*-word sequences to *m*-word sequences [102, 103]. The channel models have also been used to generate *N*-best utterance hypotheses, among which advanced syntactic/semantic models are applied in distinguishing with the aim to achieve further improvement [61, 62]. The channel-model-based approaches have been applied to some small-scale dialogue tasks and achieved encouraging results (e.g., 24.0% relative reduction in WER). One advantage of these approaches is that they do not rely on recognizer-specific information. In this way, the error correction procedures can be conveniently adapted from one baseline recognizer to another. The main disadvantage is that these approaches are a kind of adaptation technology (i.e., adapting from the training conditions to certain testing conditions) and are inapplicable if the testing conditions are changed or unknown. With inconsistent testing conditions (e.g., general-purpose recognition with multiple users under various environments), recognition errors are highly unpredictable and are difficult to model as a statistical mapping of words.

The second type of error correction effort attempts to handle recognizer defects. Modern recognizers are imperfect in many aspects. The recognizer lexicon may not adopt a suitable set of words and/or include suitable pronunciations for each word entry. Both the acoustic Hidden Markov models and trigram language models are built based on simplifying assumptions (e.g., the frame-independence assumption and short-range linguistic dependencies) [138], leading to the fact that the correct hypothesis may not have the highest likelihood among the competing hypotheses. The current decoding strategy is suboptimal in terms of word error rate. This strategy searches for an utterance hypothesis having the highest utterance posterior probability instead of an utterance hypothesis having the lowest WER [80].

Many approaches have been proposed to correct the recognition errors that are due to recognizer defects. Wakita et al. [129] addressed the lexicon defect by selectively adding new

pronunciations to word entries based on POS-dependent HMM-state confusion characteristics. This approach reduced the WER from 13.1% to 11.8% on a Japanese spontaneous speech corpus. Roark et al. [105] attempted to reduce the errors that are due to imperfect language modeling. They discriminatively modeled n-gram features and used the resulting discriminative language models to post-process the recognizer output (e.g. the recognition lattice and the *N*-best hypotheses). This leaded to a 1.3% absolute reduction (from 39.2% to 37.9%) in WER on the Switchboard corpus, which are telephone conversations. Venkataramani and Byrne [126] attempted to handle the defects in acoustic modeling by adopting support vector machines (SVMs) to distinguish between acoustically confusing word pairs that were identified through a lattice pinching technique. This approach is still preliminary, only reducing the overall WER from 45.6% to 45.5% on a spontaneous Czech conversational speech corpus. Mangu et al. [81] focused on the suboptimal decoding issue that was mentioned in the previous paragraph and proposed a post-processing algorithm to minimize word error rate. This algorithm converts a recognition lattice into a sequence of confusion sets. The words having the highest word posterior probabilities in the confusion sets are then concatenated to form the output. Experimental results demonstrated that this approach was effective, reducing the WER from 38.5% to 37.3% on the Switchboard corpus and from 33.1% to 32.5% on the Broadcast News corpus. Nowadays, this technique has been widely utilized in LVCSR systems. Mangu and Padmanabhan [82] further extended this approach by introducing rules to make a second decision between the top two words in a confusion set in terms of word posterior probability. The rules were induced from the confusion sets based on context-independent features such as the difference in word posterior probability. This enhanced confusion-set method further brought relative WER reductions of around 1.8% over the performances of the original confusion-set method on the Switchboard corpus.

Error correction methods that were proposed to fix certain recognizer defects aim to generally improve the performance of the baseline recognizer and thus are not restricted to specific testing conditions. The ability to work under flexible testing conditions is a common advantage of this type of error correction. However, most of these algorithms rely heavily on side information (e.g., word posterior probability) derived from a specific baseline recognizer. This makes such error correction mechanisms less general; adapting them from one baseline recognizer to another may require repeating the whole training procedure.

The ED-EC framework proposed in this thesis differs from previous error correction efforts in the de-correlation of error detection and correction. The proposed framework first uses an explicit error detection procedure to detect recognition errors. Then in a disjoint error correction procedure, a search space is generated by creating alternatives for the detected errors and advanced linguistic knowledge sources are applied to rank the hypotheses in the search space. The separation of error detection and error correction leads to high implementation flexibility for both procedures. By adopting general advanced linguistic knowledge sources in error correction, it is possible to design a generally applicable error correction procedure. Errors occurring in various testing conditions may be corrected by advanced linguistic constraints. If the creation of candidate alternatives for detected errors does not rely on the side information derived from the baseline recognizer, the error correction procedure will be totally independent from the baseline recognizer and thus can be conveniently adapted to other baseline recognition systems.

2.2 Detection of Recognition Errors

The ED-EC framework attempts to detect and correct the recognition errors contained in the output of a baseline recognition system. The error detection procedure serves as the basis for the error correction procedure. Since the error detection and correction procedures are relatively independent, the error detection procedure can be implemented by any error detection algorithm theoretically. In this preliminary work, we made no efforts to identify the best error detection method – we only want to choose a typical method as an example of possible error detection approaches. We thus choose the generalized word posterior probability (GWPP) [77, 122], which has been shown to be effective and efficient in error detection for both English and Mandarin LVCSR, as the feature to perform binary classification (i.e., correct or wrong) using the Naïve Bayes algorithm. However, we still introduce the previous error detection approaches in this subsection since they may be used in the error detection procedure potentially.

The technology to detect the recognition errors made by a recognizer has been intensely investigated and been used in a variety of tasks. These tasks include rejecting erroneous recognition results prior to speech understanding in spoken dialogue systems, filtering out those speech segments with wrong transcriptions in unsupervised training/adaptation, etc. [96, 130].

The task of error detection can be viewed as a classification problem. For each instance (e.g, phoneme, word, utterance) to be verified, one or multiple features are extracted and a certain classification algorithm is used to label each instance in focus as either correct or erroneous.

Most features utilized in error detection are based on the recognition information (i.e., the information contained in the output of the recognizer or obtained during decoding). The complexity of such recognition-based features varies. Some features are simply extracted from the information assigned by the recognizer to the target instance (e.g., a word hypothesis). These features include the number of component subunits (e.g., acoustic observations), the acoustic/language model scores and the score-based statistics (e.g., the standard deviation of the acoustic scores across all component acoustic observations of a word hypothesis) [16, 51, 111, 148, 149]. More complicated features attempt to capture the structures of search spaces. For example, one effective feature for word verification is calculated as the fraction of the hypotheses containing the target word in the same position among the *N*-best list [51, 148]. A similar feature is such a fraction computed among a different hypothesis list, that is, among a list of top-best hypotheses that are generated by changing the weight between the acoustic scores and language model scores in decoding [30, 111]. Other popular structural features include posterior probabilities. Posterior probability features have been shown to be effective for detecting erroneous words/utterances and can be efficiently computed from either *N*-best hypotheses or recognition lattices [77, 101, 122, 131, 133]. There are also recognition-based features that analyze elaborate recognition information, such as the rank of the best state (i.e., the one in the state sequence that corresponds to the best path in decoding) among the competing states [11] and the neighborhood location of the target instance in HMM model space [63].

Recently, features based on additional knowledge sources that are unavailable in baseline recognition have been introduced in error detection. Leung and Siu [74] presented two articulatory features that evaluate the match in articulatory properties (e.g., rounding and manner) between the target phoneme sequence and the corresponding speech segment using neural network classifiers. A myriad of works used various semantic and/or syntactic features to detect recognition errors for domain-specific dialogue systems [14, 48, 96, 100, 110, 143]. For a limited domain such as travel planning, it is feasible to design a grammar covering the task and perform robust parsing. This makes the extraction of syntactic/semantic features reliable. The syntactic/semantic features investigated for domain-specific dialogue systems include the

number of transitions between parsed and unparsed fragments [14], slot-based language model probabilities [100], semantic weights of word classes [96], etc. Several efforts have been carried out to use syntactic and/or semantic features in error detection for general-domain speech recognition. In this case, a covering grammar is absent and the performance of parsing is typically unsatisfactory. Zhou et al. [146] attempted to capture the left/right relationships between words as syntactic features using a link grammar. Semantic features that have been suggested for general LVCSR are extracted based on latent semantic analysis [25, 47] or inter-word mutual information [47]. Adopting features based on additional knowledge beyond the side information derived from a specific recognizer enhances the generality of error detection. However due to robustness concerns, these features are mainly used to complement the recognition-based features.

While using a single feature in error detection leads to a simple binary decision task, adopting multiple features triggers the involvement of complex classification algorithms. Many classification approaches have been utilized for error detection, including generalized linear models [43], neural networks [111, 131, 132, 143, 146], decision tree [143], linear discriminant projection [51, 96, 111], Gaussian mixture modeling [66] and support vector machines [79, 143, 146, 149]. Research in feature selection has been carried out. Zhou et al. [146] used maximum likelihood to assess the importance of each feature; Zhou and Meng [148] developed an iterative procedure to filter unnecessary features. There has also been works combining utterance- and word-level verification [51, 148].

2.3 Search Spaces for Knowledge Application

After detecting recognition errors, the ED-EC framework creates a candidate list of alternatives for each detected error in an utterance. Connecting the candidate lists with the utterance context leads to a sausage-type search space. Advanced LMs are then applied to correct the detected errors by ranking the utterance hypotheses in this search space. For LVCSR, different post-processing search spaces have been utilized to apply additional acoustic/linguistic knowledge sources. In this section, we introduce these search spaces and compare them with the sausage-type search space used in the ED-EC framework.

The *N*-best hypothesis list is the most frequently utilized search space for post-processing due to its simplicity [5, 147]. Applying knowledge models to score those hypotheses listed in parallel is straightforward. The *N*-best hypotheses can be those output by a baseline recognizer. They can also be created by a more complicated procedure. For instance, Jeong et al. [61] processed the recognized utterance produced by a baseline recognizer with a word-for-word channel model and used this channel model to create *N*-best hypotheses. Zhang and Rudnicky [145] adopted a number of acoustic model (AM) sets to construct a hypothesis list in which each hypothesis was a recognized utterance generated by decoding with an individual AM set.

The lattice search space is also widely used for applying additional acoustic/linguistic sources [28, 85, 124]. A lattice search space is normally more informative than the corresponding *N*-best hypothesis list because (1) it normally contains a larger number of utterance hypotheses and (2) it illustrates time relationships as well as the connections between word hypotheses. Similar to the search space of *N*-best hypotheses, a lattice search space can be either a recognition lattice generated during decoding or a lattice produced by post-processing. For the latter case, the most widely used representation is the confusion network (i.e., sequence of confusion sets) derived from the recognition lattices [82, 126]. A lattice search space can also be obtained by merging the recognized utterances that are generated by multiple recognizers for an input speech utterance into a single network [31].

The difference between the previous post-processing search spaces and the sausage-type search space used in the ED-EC framework lies in the distribution of competing hypotheses. In the sausage-type search space, competing hypotheses (i.e., alternatives in a candidate list) only exist for the signal segments that are detected as erroneous. In contrast, for previous post-processing search spaces, hypotheses are typically generated for all speech segments in a unified way. We will discuss the impact of this difference on the computational load of advanced LMs in Section 6.3.

The sausage-type search space is similar to the confusion network [82] in the sense that they both generate lists of alternatives (i.e., confusion sets) for speech segments. The confusion network is created by merging the hypotheses in the recognition lattice. A large confusion set indicates that the original recognition result corresponding to the confusion set is likely to be erroneous. At the same time, a small confusion set that contains few hypotheses indicates that the corresponding recognition result tends to be correct. The ED-EC framework detects

recognition errors in a more explicit way. Using a separate error detection procedure, information beyond the recognition lattice can be used and sophisticated classification methods can be adopted in error detection. With better error detection, the competing hypotheses can be more concentrated on the erroneous regions in the signals. In addition, for the sausage-type search space, the alternatives are not necessary to be the hypotheses in the recognition lattice.

2.4 Advanced Linguistic Knowledge Sources

The ED-EC framework applies advanced LMs to improve LVCSR. Theoretically, any advanced LMs that can be used to rank utterance hypotheses are readily to be applied in this framework. Similar to the choice of the error detection algorithm, in this preliminary work, we made no efforts to identify the best advanced LM and instead portrayed a typical advanced LM as a representative of all possible advanced LMs. This LM uses mutual information as an example of long-distance constraints and uses POS trigram as an example of syntactic constraints. In this section, we introduce previous advanced LMs which are potentially applicable for the ED-EC framework.

Many works have been proposed to model linguistic knowledge sources that are more advanced than trigram information. Some of these efforts attempted to extend trigrams to higher-order n-grams to capture longer-distance constraints [2, 19, 46]. To address the data sparseness problem, several variations of n-gram model have been investigated, including category-based n-gram models [38, 94] and skipping models (i.e., word probability models that condition the probability of word w_i on an incomplete $(n-1)$-word history such as $(w_{i-4},w_{i-2},w_{i-1})$) [56, 84, 119]. Although raising n for n-gram modeling is beneficial for improving recognition accuracy, the performance plateaus after n rises to a certain number (e.g., 6) [46]. To achieve breakthroughs, "high-level" syntactic and/or semantic constraints have been introduced in language modeling. By "high-level" we mean that the constraints cover complex linguistic knowledge beyond the word sequence information. High-level syntactic constraints capture the hierarchical characteristics of a language, while high-level semantic constraints capture long-distance semantic dependencies across the context of the entire utterance/document.

As mentioned in Section 2.2, extracting syntactic/semantic information is relatively convenient for domain-specific applications but is difficult for general-domain applications. For domain-specific speech recognition tasks, various high-level syntactic and/or semantic sources have been applied in different ways [12, 50, 57, 92, 116, 142]. For example, Mou et al. [92] incorporated phrase-based shallow parsing information in recognition using a layered Finite State Transducer framework. Seneviratne and Young [116] assigned parse probabilities to word sequences using a right branching stack automaton. Huning et al. [57] compiled small deterministic grammars into syntactically enhanced word bigrams, which are applicable without a parsing procedure.

For general-domain LVCSR, efforts have been made in utilizing high-level knowledge although the parsing procedure may not be robust in this case. Several semantic sources that do not rely on tagging/parsing have been modeled using data-driven approaches. These sources include topic information [42, 59, 134], latent semantic analysis (LSA) [7, 144] and word triggers selected based on mutual information [107, 117]. There are also semantic constraints (e.g., semantic tags and semantic arguments for predicates) extracted using a semantic tagger/parser [5, 141]. Syntactic sources that have been investigated range from POS tags to hierarchical structures such as linkage graphs (i.e., graphs that depict the link relationships between words) and parse trees [18, 52, 60, 104]. Syntactic information can be extracted by traditional text tagging/parsing techniques which process whole utterances [123]. A recent trend to capture structural knowledge in recognition is to design the parsing methods that are suitable for left-to-right decoding. These models include probabilistic shift-reduce paring [18], probabilistic top-down parsing [104] and probabilistic left-corner parsing [125]. Semantic and/or syntactic constraints have been applied in speech recognition with different frameworks. Wang and Harper [5] used a dependency-grammar almost parsing model to combine syntactic and semantic constraints. Bellegarda [6] integrated the LSA and n-gram paradigms into the calculation of conditional word probabilities. Multiple constraints have also been jointly applied with an *N*-best re-ranking mechanism [5], a Softmax network [144] or dynamic Bayesian networks [134]. A popular choice for constraint combination is to adopt the maximum entropy framework [17, 68, 107, 141, 142].

Most efforts to incorporate high-level syntactic/semantic knowledge into general LVCSR brought negligible or even no improvement compared with state-of-the-art trigram decoding.

This is partially because of the difficulty in robustly extracting and modeling syntactic/semantic knowledge. When applying an unreliable high-level linguistic model, the distinguishing power of advanced constraints may be greatly reduced by the misleading information introduced. Even so, several works that incorporated high-level constraints in LVCSR obtained encouraging improvements over the performance of trigram decoding [18, 125]. This demonstrated the power of sophisticated linguistic knowledge sources. Focusing the application of high-level linguistic models on those signals where an n-gram decoding procedure makes mistakes can relieve the unreliability problem of these models, because in this case the influence of misleading information on the correct regions in the signals can be eliminated.

Another thing worth noting is that advanced linguistic knowledge sources that are modeled for error detection are potentially beneficial for enhancing recognition performance. A linguistic knowledge source that can be utilized to determine whether a hypothesis is wrong should also be able to help identify the correct hypothesis among the competing ones.

2.5 Chapter Summary

This chapter presents the previous works that are related to the ED-EC framework. The ED-ED framework is inspired by a spelling correction technology, which attempts to correct potential written errors by applying additional linguistic constraints to distinguish among the alternatives in predefined confusion sets. For LVCSR, it is difficult to predefine confusion sets for all possible recognition errors. The ED-EC framework adapts the spelling correction technology to the field of LVCSR by dynamically detecting recognition errors and dynamically creating confusion sets (i.e., candidate lists of alternatives) for detected errors. Compared with the previous error correction efforts in LVCSR, the ED-EC framework is different in the sense that the error detection and error correction procedures are independent. The separation of error detection and error correction leads to high flexibility in the implementation of the error detection/correction procedure. This chapter also reviews the previous works that are related to different components of the ED-EC framework. The framework detects recognition errors, creates candidate lists for detected errors to generate new search spaces, and then applies advanced LMs in the new search spaces to correct errors. Thus, previous efforts in the detection

of recognition errors, the construction of search spaces and the use of advanced LMs are also discussed in this chapter.

Chapter 3

An Error Detection and Correction Framework

3.1 Overview

This thesis proposes an error detection and correction (ED-EC) framework to incorporate sophisticated linguistic knowledge sources into large vocabulary continuous speech recognition (LVCSR). The main idea of the ED-EC framework is to post-process the output of a state-of-the-art recognizer by (1) detecting recognition errors and (2) correcting the detected errors with the aid of advanced linguistic knowledge sources. This framework attempts to achieve the full benefit of advanced linguistic knowledge at minimal computational cost by applying computationally expensive language models only to regions of the signal where the state-of-the-art recognizer fails. To test the feasibility of the ED-EC framework, we propose a prototype for Mandarin LVCSR.

Although the ED-EC framework is conceptually language independent, the detailed design of the framework should consider the characteristics of the language of interest. For Mandarin, the recognition result of a recognizer for an input utterance (e.g., 在新闻中心拜会议长 Translation: Meet the prolocutor at the news center) is a word sequence (e.g., 在 at/新闻 news/中心 center/拜会 meet/议长 prolocutor), which can also be viewed as a character sequence (e.g., 在新闻中心拜会议长). One distinguishing characteristic of Mandarin LVCSR is that if the recognition results are viewed as character sequences, substitutions normally represent the majority of all character errors (i.e., substitutions, insertions and deletions) [34], [88]. This is because Mandarin has a relatively simple syllable structure. For Mandarin, all syllables share the

restricted form of consonant(optional)-vowel-consonant(optional) and consequently, the decoding procedure rarely makes mistakes in segmentation. On the basis of this characteristic, we assume that all character recognition errors are substitutions. Thus, the task of the ED-EC prototype is simplified to detecting and correcting erroneous characters in the output of the baseline recognition system. The complexity of handling insertions/deletions is avoided.

The basic structure of the Mandarin ED-EC prototype is as follows:

- ***Error Detection***: The error detection procedure attempts to detect erroneous characters in the output of a baseline recognizer. We first detect erroneous words by means of a word verifier based on Generalized Word Posterior Probability (GWPP), a probability scoring method that will be introduced later. We then detect erroneous characters within each erroneous word.
- ***Error Correction***: The error correction procedure attempts to correct the detected erroneous characters. For each erroneous character, we create a candidate list of character alternatives with the aim of including the correct character (i.e., the reference). The candidate creation algorithm will be presented later. Combining these candidate lists with the context in the utterance leads to new search networks. We formulate an advanced linguistic model by combining inter-word mutual information, word trigrams and part-of-speech (POS) trigrams. We then use this advanced model to re-rank the utterance hypotheses contained in the new networks. Character candidates in the top-ranking hypotheses are output as the results of error correction. The main handicap in this process lies in the fact that the previous error detection procedure is imperfect and "correcting" detected errors that are already correct may introduce new errors. To alleviate this problem, an additional mechanism is proposed to accept the correction results only when certain conditions are satisfied.

Figure 3.1 shows an example of the Mandarin prototype. In this example, the baseline recognizer transcribes the speech utterance "在新闻中心*拜会*议长" (translation: "Meet the prolocutor at the news center") as "在新闻中心*百位*议长" (translation: "Hundreds of prolocutor

at the news center"). 拜/bai4/[1] and 会/hui4/ are wrongly recognized as two acoustically similar characters, 百/bai2/ and 位/wei4/, respectively. The ED-EC framework first applies the error detection procedure to identify the two erroneous characters 百 and 位 as errors. The error correction procedure is then applied to correct the detected errors. In the top-ranking hypothesis, the two errors are successfully corrected into the characters 拜 and 会 respectively.

Input: < speech utterance> 在新闻中心拜会议长 (Meet the prolocutor at the news center.)
Baseline Recognition: 在新闻中心*百位*议长
Error Detection: *百*(hundred)，*位*(number) detected as errors
Error Correction: 1. Expand each error into a candidate list ► new search network 在-新-闻-中-心- (百 / 外 / 拜) - (位 / 会 / 回) -议-长 2. Rerank the hypotheses with the advanced language model (Sources: Mutual Information; Word Trigram; POS Trigram)
Output: <top-ranking hypo.> 在新闻中心拜会议长 (*Corrected*)

Figure 3.1: An illustration of the ED-EC prototype for Mandarin LVCSR

It should be noted that we focus on erroneous *characters* instead of erroneous *words* when performing error detection and error correction. This is to circumvent the segmentation problem of the Chinese language. In Chinese, a sentence is a character sequence without an explicit word delimiter, and the definition of words is debatable. The segmentation of a sentence into a string of words is not unique, decided by both the lexicon and the segmentation algorithm. For example, the sentence "在新闻中心拜会议长" may be segmented into various word sequences, including a meaningful word sequence "在(at)/新闻(news)/中心(center)/拜会(meet)/议长(prolocutor)" and a meaningless sequence "在(at)/新闻(news)/中心(center)/拜(bow)/会议(meeting)/长(long)". However, advanced linguistic models are normally based on words which

[1] /[syllable][tone]/ denotes the tonal syllable according to Chinese pinyin. There are in total five tones, denoted by 1, 2, 3, 4 and 5, respectively.

bear semantic information. By choosing the character as the basic unit of recognition error, the error correction procedure can be independent of the recognition baseline in terms of knowledge usage and will have more freedom in selecting suitable word lexicons to model linguistic knowledge sources. For instance, in this work, the lexicons utilized in error correction are different from the one incorporated in the baseline recognizer. The lexicon chosen by the POS trigram model is different from the lexicon adopted by the MI and word trigram models.

The main structure of the Mandarin ED-EC prototype can be illustrated as Figure 3.2. In the rest of this chapter, we describe the Mandarin ED-E prototype in detail. Section 3.2 discusses the input to the ED-EC prototype. Sections 3.3 and 3.4 present the error detection and error correction procedures, respectively.

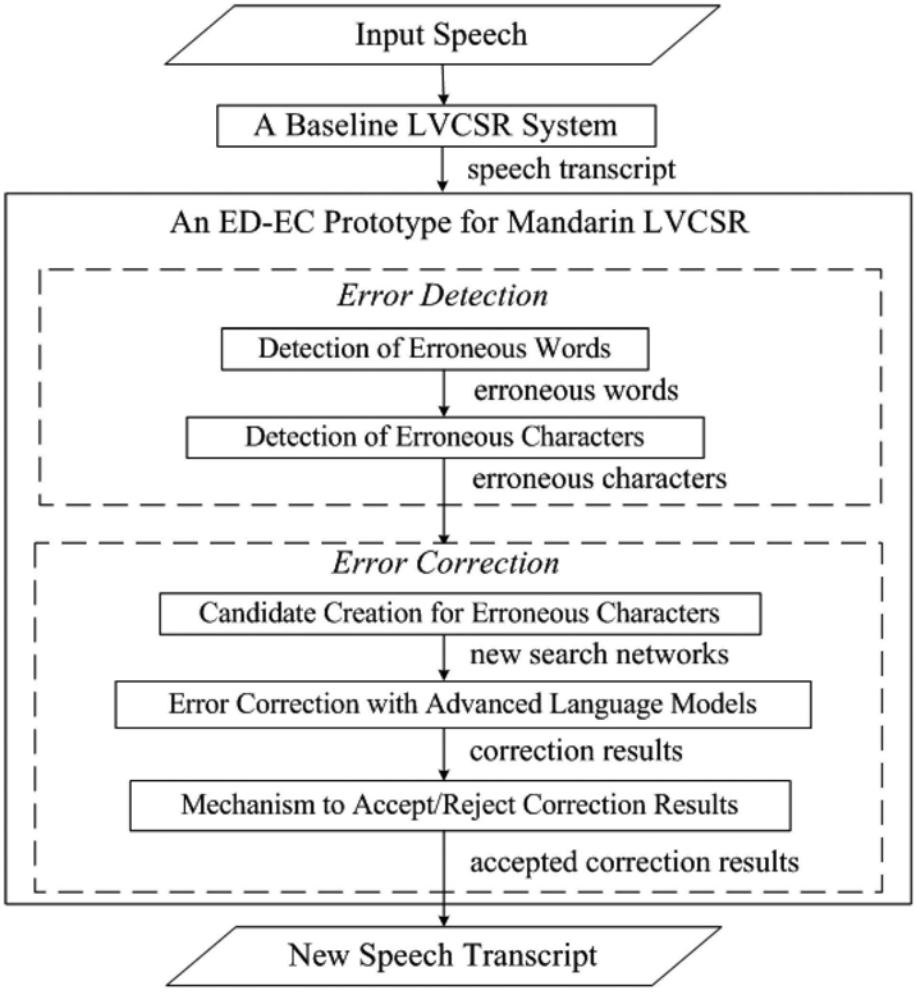

Figure 3.2: The structure of the ED-EC prototype for Mandarin LVCSR

3.2 Input to the ED-EC framework

The ED-EC framework has been proposed to improve the performance of a baseline recognizer. For each speech utterance, the baseline recognizer generates an utterance hypothesis, called a recognized utterance, from which the ED-EC framework detects and corrects errors. The baseline recognizer may also provide additional decoding information such as *N*-best hypotheses and recognition lattices as an extended input to the ED-EC framework if needed. For the proposed Mandarin prototype, recognition lattices are required for both the error detection and error correction algorithms, as will be discussed later in this chapter. Therefore, the inputs to the Mandarin prototype are the recognized utterances along with the corresponding recognition lattices generated by the recognizer.

The ED-EC framework can also be applied to improve the performance of an enhanced LVCSR system that already adopts one or more techniques to post-process the output of a recognizer. In this case, the framework attempts to detect and correct recognition errors in the recognized utterances generated by the enhanced baseline system. The enhanced baseline system may provide the framework with some additional recognition information to facilitate the error detection and/or correction. To investigate the applicability of the framework to enhanced baseline systems, we (1) propose a discriminative lattice rescoring technique to rescore the recognition lattices generated by a recognizer, (2) use the new recognized utterances (i.e., the top-best utterance hypotheses in the rescored lattices) along with the rescored lattices as the input of the Mandarin ED-EC prototype and (3) apply the prototype to the discriminatively enhanced system in the same way as to the original recognizer system.

In the subsequent sections, we develop the Mandarin ED-EC prototype with the aim to improve the performance of a baseline recognizer. We will apply the proposed prototype to the discriminatively enhanced baseline systems in Chapter 8.

3.3 Error Detection

We propose a two-step procedure for error detection. This procedure first detects erroneous words (i.e., words containing erroneous characters) within the recognized utterances. The procedure then detects erroneous constituent characters from these words. The two steps for error detection are described in Sections 3.3.1 and 3.3.2, respectively. Missed detections and false alarms, which are typical errors in error detection, are discussed in Section 3.3.3.

3.3.1 Detecting Erroneous Words

The first step of the error detection procedure is to use a word verifier to classify each word in a recognized utterance as either correct or erroneous. In this study, we implement the word verifier as a binary classifier based on a Generalized Word Posterior Probability (GWPP) feature. This subsection first describes the concept of GWPP. Then, two related posterior probabilities (i.e., string posterior probability and word posterior probability) and GWPP are defined. The word verifier is then presented.

Given a sequence of acoustic observations, the String Posterior Probability (SPP) of a hypothesized word string that is generated by a recognizer can be expressed as,

$$P([w;s,t]_1^M \mid x_1^T) = \frac{p(x_1^T \mid [w;s,t]_1^M) \cdot p([w;s,t]_1^M)}{p(x_1^T)} = \frac{\prod_{m=1}^{M} p(x_{s_m}^{t_m} \mid w_m) \cdot p(w_m \mid w_1^M)}{p(x_1^T)} \tag{3.1}$$

where $x_1^T = x_1 x_2 \cdots x_T$ is the given acoustic observation sequence; $[w;s,t]$ denotes a word hypothesis for the word w, starts at time s and ends at time t; $[w;s,t]_1^M$ is a hypothesized word string $[w_1;s_1,t_1][w_2;s_2,t_2]\cdots[w_M;s_M,t_M]$; and $p(x_{s_m}^{t_m} \mid w_m)$ and $p(w_m \mid w_1^M)$ are the acoustic model probability and the language model probability respectively.

The Word Posterior Probability (WPP) for a word hypothesis $[w;s,t]$ is computed by summing the string posterior probabilities of all paths for an identical word hypothesis in the search space (e.g., the recognition lattices and the *N*-best hypotheses). It can be written as

$$P([w;s,t]\,|\,x_1^T) = \sum_{\substack{\forall M,[w;s,t]_1^M \\ \exists n,1\le n\le M \\ w=w_n,s_n=s,t_n=t}} P([w;s,t]_1^M \mid x_1^T)$$

$$= \sum_{\substack{\forall M,[w;s,t]_1^M \\ \exists n,1\le n\le M \\ w=w_n,s_n=s,t_n=t}} \frac{\prod_{m=1}^{M} p(x_{s_m}^{t_m} \mid w_m)\cdot p(w_m \mid w_1^M)}{p(x_1^T)} \tag{3.2}$$

GWPP is a generalization of WPP. GWPP extends the calculation of WPP by (1) relaxing time registration and (2) re-weighting acoustic and language model probabilities to achieve optimal performance in word verification. We follow [77] to define GWPP as

$$P_G([w;s,t]\,|\,x_1^T) = \sum_{\substack{\forall M,[w;s,t]_1^M \\ \exists n,1\le n\le M \\ w=w_n \\ (s_n,t_n)\cap(s,t)\neq\phi}} \frac{\prod_{m=1}^{M} p^{\alpha}(x_{s_m}^{t_m} \mid w_m)\cdot p^{\beta}(w_m \mid w_1^M)}{p(x_1^T)} \tag{3.3}$$

where α and β are acoustic and language model weights, respectively. As shown in Equation 3.3, we relax the time registration to allow overlapping. If an utterance hypothesis contains a word hypothesis for the word *w* and overlaps in time with [*w; s, t*], we include the posterior probability of this utterance hypothesis in the summation.

In this study, we utilize the recognition lattices that are generated during the decoding as the search space to compute GWPP. For each recognized word, we select its GWPP value as the feature of interest. Based on this GWPP feature, the word verifier uses the Naïve Bayes algorithm to perform binary classification [136]. The details for developing the word verifier, such as the tuning of α and β, will be presented in Section 4.3.1. The word verifier assigns each word a confidence score, which ranges from 0 to 1. This score indicates the confidence level of the judgment that the word is erroneous. If the confidence score of a word is less than 0.5, the word verifier labels this word as correct. Otherwise, this word is labeled as erroneous.

3.3.2 Detecting Erroneous Characters

Focusing on the words deemed erroneous, the error detection procedure proceeds to detect erroneous characters. It is possible to design a character verifier to classify each character in focus as either correct or erroneous. However, for simplicity and efficiency, this study makes the assumption that all the characters of an erroneous word are wrong. With this assumption, we label all characters in focus as erroneous.

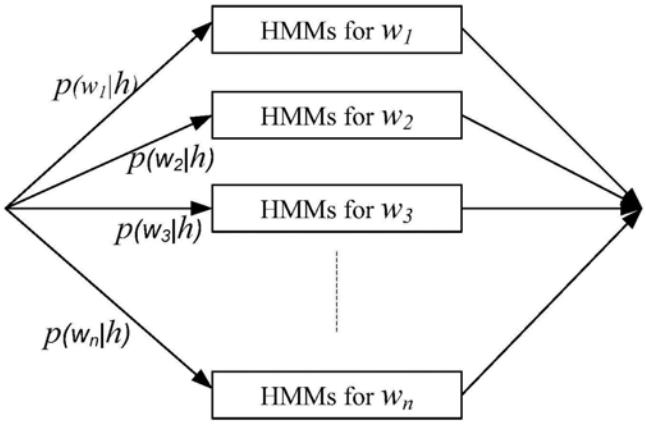

Figure 3.3: The concept of the recognition network. w_i is a word which contains one or more characters; $p(w_i|h)$ is the conditional probability of w_i given a history h

This assumption is plausible because state-of-the-art Mandarin decoding uses the word as the smallest linguistic unit (see Figure 3.3). If the speech segment corresponding to a word is wrongly transcribed as another word or word sequence, the correct characters may only appear by coincidence. More often, they are transcribed as other characters with same/similar pronunciations. Note that for each Chinese character, there are on average 30 other characters sharing the same syllable, as will be further discussed in Section 3.4.1. The number of characters with acoustically similar syllables is even larger. Given a long list of acoustic similar characters for each character, the occurrence rate of correct characters is low in erroneous words. For example, the word 幻境/huan4 jing4/ (fantasy) may be recognized as 欢迎/huan1 ying2/ (welcome), 安静/an1 jing4/ (peaceful), or 幻听/huan4 ting1/ (acouasm). Only the last word 幻听 coincidentally includes one correct character 幻. With low character accuracy of erroneous words, labeling all component character as erroneous leads to high computation efficiency and limited performance loss.

We assign a confidence score to each character labeled as erroneous. The confidence score of a detected erroneous character is set to be the same as the confidence score of the detected erroneous word that contains the character.

3.3.3 Missed Detections and False Alarms

The error detection procedure is imperfect. Error detection may produce missed detections and false alarms. False alarms involve correct characters wrongly labeled as erroneous. There are three types of missed detections. First, all detections are missed detections due to the usage of the simplifying assumption that all character recognition errors are substitutions. Second, those substitutions wrongly labeled as correct are missed detections. Third, those insertions that are deemed correct are also missed detections.

Both missed detections and false alarms have an impact on the error correction procedure. In error correction, an advanced linguistic model is applied to rank utterance hypotheses within a search space, in which detected errors have alternatives and the utterance context is unique. Missed detections lie in the utterance context in a search space. Their existence will bring misleading information into the discrimination of competing alternatives. False alarms will not only cause unnecessary computational load via candidate creation and linguistic ranking but may also introduce new misrecognitions. If a correct character is labeled as erroneous, applying an error correction procedure may actually turn it into a wrong character.

3.4 Error Correction

The error correction procedure attempts to use advanced linguistic knowledge sources to correct the erroneous characters detected by the previous error detection procedure. As described in Section 3.1, the error correction procedure first creates a candidate list of character alternatives for each detected erroneous character to construct new search networks. An advanced linguistic model is then used with the aim of correcting the detected errors. To handle false alarms in error detection, an additional mechanism is also applied.

We describe the method to create candidate lists for erroneous characters in Section 3.4.1. The application of an advanced linguistic model in error correction is described in Section 3.4.2. The special mechanism to handle error-detection false alarms is proposed in Section 3.4.3.

3.4.1 Candidate Creation

The task of candidate creation is to create a candidate list of alternatives for each detected error. Candidate creation is an important component of the error correction procedure since it is the basis for the subsequent knowledge application. If the alternatives created for each error are similar in pronunciation, we may simply apply linguistic models to re-rank the utterance hypotheses in the new networks, since all competing utterance hypotheses are acoustically similar. Otherwise, acoustic models have to be involved in hypothesis re-ranking. Additionally, if candidate creation fails to include the actual correct candidate for selection, the error will have no chance of being corrected. Our goal is to create candidate lists in which character alternatives are similar in pronunciation, and at the same time the correct characters are included.

Given a character error, the easiest way to create a list of acoustically similar character alternatives is to include all the homophonous characters. For Mandarin, characters are pronounced as tonal syllables (e.g., /ha1/), which are syllables with tone information. We refer to syllables without tone as base syllables (e.g., /ha/). On average, there are 8 characters sharing the same tonal syllable and 31 characters sharing the same base syllable in Mandarin, according to CCDICT (2000). Incorporating the characters that have the same tonal/base syllable as a given error into the candidate list seems possible. However, such candidate lists may often fail to include the correct characters. Due to the use of language model in decoding, correct characters are recognized as erroneous characters with different base syllables in many cases. For the example in Figure 3.1, the character 会/hui4/ is wrongly recognized as 位/wei4/. The two base syllables /hui/ and /wei/ are similar in pronunciation, but different. Thus, the correct character 会/hui4/ will not be found in the candidate list created based on the syllable /wei/. To handle this problem, one would think that homophonous characters for those base syllables that have similar pronunciation with the error should be added to the candidate lists. However, this is unrealistic. Since the decoding procedure is complex, it is hard to predict which acoustically

similar base syllables should be considered for candidate creation, and the number of such possible base syllables is large.

In this work, we attempt to seek help from recognition information for candidate creation. We propose an approach to select acoustically similar character alternatives for each erroneous character from recognition lattices that are generated during decoding. This approach creates a candidate list for each character error using three steps:

Step 1. From the recognition lattice, select character hypotheses with starting and ending times similar to the erroneous character. For each selected hypothesis, if the character carried by the focused hypothesis has not been included in the candidate list, add this character to the candidate list.

Step 2. Rank the character alternatives in the candidate list based on their Generalized Character Posterior Probabilities (GCPPs).

Step 3. Prune the candidate list by keeping only the top N (e.g., N=20) character alternatives.

In *Step 1*, we state that a character hypothesis has starting and ending times similar to the target erroneous character if these time boundaries encompass the midpoint of the target erroneous character. With this definition of time similarity, we further define GCPP in a similar way as GWPP: GCPP is the summation of the posterior probabilities of all utterance hypotheses in the lattice bearing the character in focus with similar starting and ending times. GCPP can be written as,

$$P_G(c_a \mid x_1^T) = \sum_{\substack{\forall K,[c;s,t]_1^K \\ \exists i, 1\le i\le K \\ c_i=c_a \\ s_i<\frac{s_e+t_e}{2}<t_i}} \frac{p(x_1^T \mid [c;s,t]_1^K)\cdot p([c;s,t]_1^K)}{p(x_1^T)} \tag{3.4}$$

where c_a is a character alternative; $[c;s,t]$ is a character hypothesis bearing character c, starting at time s and ending at time t; $[c;s,t]_1^K$ is an utterance hypothesis (i.e., a sequence of character hypotheses); and s_e and t_e are the starting and ending times of the target erroneous character.

The GCPPs of candidate characters can be computed efficiently during the creation of candidate lists. For each character hypothesis selected in *Step 1*, we record its posterior probability, which is equal to the posterior probability of the word hypothesis containing it. We then sum up the posterior probabilities for those selected character hypotheses with the same

character identity. The summations are the GCPPs of the candidate characters, as proven in the equation below,

$$
\begin{aligned}
\sum_{s_a<\frac{s_e+t_e}{2}<t_a} P([c_a;s_a,t_a]\,|\,x_1^T) &= \sum_{\substack{[c_a;s_a,t_a]\in[w;s,t] \\ s_a<\frac{s_e+t_e}{2}<t_a}} P([w;s,t]\,|\,x_1^T) \\
&= \sum_{\substack{\forall M,[w;s,t]_1^M=[c;s',t']_1^K \\ \exists i,j,1\le i\le K,1\le j\le M \\ c_i=c_a,c_i\in w_j \\ s'_i<\frac{s_e+t_e}{2}<t'_i}} P([w;s,t]_1^M\,|\,x_1^T) \\
&= \sum_{\substack{\forall K,[c;s,t]_1^K \\ \exists i,1\le i\le K \\ c_i=c_a \\ s_i<\frac{s_e+t_e}{2}<t_i}} P([c;s,t]_1^K\,|\,x_1^T) \\
&= P_G(c_a\,|\,x_1^T)
\end{aligned}
\tag{3.5}
$$

where $P(y|x_1^T)$ is the posterior probability of the item y in focus

In the last step, we prune the size of the candidate lists to N. This is to keep the computational load of the subsequent processing (i.e., ranking the utterance hypotheses in the new search) under control.

There are three underlying assumptions behind the proposed three-step algorithm. First, we assume that recognition lattices contain the correct transcriptions in similar time periods as the errors (i.e., that the correct transcriptions will be selected). This assumption is grounded on the fact that state-of-the-art recognizers have achieved high *lattice accuracies* (i.e., accuracies of the best paths in recognition lattices). For example, on the Switchboard telephone conversation corpus, Mangu et al. [6] reported a lattice accuracy of 90% while the recognition accuracy was only 62%. The high lattice accuracies indicate that recognition lattices have a good chance of including the correct transcriptions in similar time periods with the errors. Second, we assume that in a recognition lattice, all character hypotheses with starting and ending times similar to those of the erroneous character have similar pronunciations (i.e., the character alternatives selected are acoustically similar). This assumption is generally true since the character hypotheses selected for an error more or less match the corresponding signals in acoustics. Third, we assume that if the candidate characters selected include the correct transcription, the correct character has a relatively high GCPP (i.e., the correct transcription will not be pruned). As we

discussed previously, GCPP is defined in a similar way as GWPP, which has been shown to be effective to measure the correctness of word hypotheses. We expect that GCPP is effective at measuring the correctness of characters and that the correct character will have a GCPP high enough to be kept during the pruning.

3.4.2 Linguistic Scoring

After creating candidate lists of alternatives for detected errors, the ED-EC framework combines the candidate lists with the corresponding utterance context to form new search networks. Then, the framework attempts to correct the detected errors by using advanced language models (LMs) to rank the utterance hypotheses in the new search network. The underlying assumption is that among the competing utterance hypotheses, the advanced LMs will assign the highest score to the correct hypothesis or the best hypothesis with the highest accuracy. This assumption may be true if the chosen LMs can handle the operating conditions well.

For the Mandarin prototype, the utterance hypotheses in new search networks (see Figure 3.4) are scored and ranked by an advanced LM that combines three knowledge sources: inter-word mutual information, word trigrams and POS trigrams. The three sources aim to capture long-distance semantic constraints, local constraints and syntactic constraints respectively. Those candidate characters (e.g., 拜 and 会) contained in the top-ranking hypotheses (e.g., 在新闻中心**拜会**议长) are viewed as the error correction results.

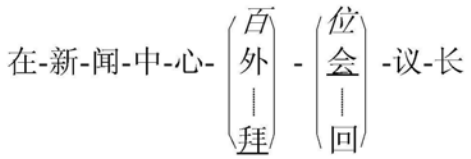

Figure 3.4: A sausage-type search network

This subsection presents the methodology to score an utterance hypothesis using the advanced LM. Given an utterance hypothesis, we first score it by the three individual linguistic knowledge sources separately and then calculate the overall score as a linear combination of the individual scores. The scoring method for each linguistic knowledge source and the score interpolation are described in this subsection. Note that all the three individual LMs are word-

based because the word is the basic unit carrying semantic information. For each individual LM, hypotheses need to be segmented into word strings by certain segmentation algorithm using the lexicon used for the modeling of the individual LM of choice.

- **Inter-word mutual information**

Mutual Information (MI) is used to capture long-distance semantic dependencies between words. The algorithm we propose to score utterance hypotheses by MI is adapted from [47], which applied MI in confidence measuring. We first follow [47] to define MI between two words *x* and *y* as,

$$MI(x,y) = \log(\frac{p(x,y)}{p(x)p(y)}) \tag{3.6}$$

where

$$p(x,y) = N(x,y)/(\sum_{x,y} N(x,y)); \tag{3.7}$$

$$p(x) = \sum_{y} p(x,y); \tag{3.8}$$

$$p(y) = \sum_{x} p(x,y); \tag{3.9}$$

N(x, y) refers to the times that both word *x* and word *y* appearing in an utterance; *p(x,y)* is the joint probability of co-occurrence of *x* and *y* in an utterance; and *p(x)* and *p(y)* are the two corresponding marginal probabilities.

Due to data sparseness, we smooth the MI model as follows,

$$N'(x,y) = N(x,y) + C \tag{3.10}$$

where *C* is a constant tuned by grid search.

Using the above definitions, we assign a score to each utterance hypothesis based on the mutual information between the candidate alternatives and their utterance contexts. To facilitate the discussion, we first introduce the concept of a *target word*. A target word refers to a word that contains one or more candidates (i.e., character alternatives in candidate lists that are created for detected errors) in an utterance hypothesis. For example, there is an utterance hypothesis 在新闻中心***白会***议长 in the search network shown in Figure 3.2, where 白 and 会 are two

candidate characters. After segmentation, the utterance was transformed into a sequence of words 在/新闻/中心/**白**/**会**议/长. Among these words, the two words 白 and 会议 contain candidate characters and are thus target words. With the target word defined, we proceed to score each utterance hypothesis by (1) assigning each target word a score that is the average MI value of the word with all the other words in this hypothesis, as shown in Equation 3.11, and (2) assigning the average score of the target words as the hypothesis's score, as shown in Equation 3.12.

$$AvgMI(w_k \mid w_1, w_2, ...w_m) = (\sum_{\substack{i=1,..m \\ i \neq k}} MI(w_i, w_k))/(m-1) \tag{3.11}$$

$$K_{MI}(w_1, w_2, ...w_m) = (\sum_{k=1,...n} AvgMI(w_k \mid w_1, w_2, ...w_m))/n \tag{3.12}$$

where $w_1, w_2, \ldots w_m$ is the word sequence of the utterance hypothesis; w_k is a target word; and n is the number of target words.

- **Word trigram**

Word trigrams capture local constraints. Although the baseline recognizer has already incorporated a word trigram model, we continue to apply word trigrams for error correction because the search networks constructed for error correction differ from the decoding search space. The word trigram model scores each utterance hypothesis as,

$$K_{WdTri}(w_1, w_2, ...w_m) = \sum_{i=1...m} P(w_i \mid w_{i-2}, w_{i-1}) \tag{3.13}$$

where $P(w_i|w_{i-2},w_{i-1})$s are the word-based trigram probabilities in the log domain.

- **POS trigram**

We adopt POS trigrams to capture syntactic constraints. The hypothesis scoring method for the POS trigram model is similar to that of the word trigram model. The only difference is that in this case, the trigram probabilities are computed based on the sequence of POS tags as follows,

$$K_{PosTri}(w_1, w_2, ...w_m) = \sum_{i=1...m} P(pos_i \mid pos_{i-2}, pos_{i-1}) \tag{3.14}$$

where pos_i is the corresponding POS tag of the word w_i and $P(pos_i|pos_{i-2},pos_{i-1})$s are the POS-based trigram probabilities in the log domain.

- **Linear combination of linguistic knowledge sources**

Combining complementary linguistic knowledge sources should be beneficial. There are many approaches, such as the maximum entropy framework, to combine multiple sources. In this study, as an initial prototype, we simply interpolate the individual sources linearly as,

$$K(h) = r_1 \cdot K_{MI}(h) + r_2 \cdot K_{WdTri}(h) + r_3 \cdot K_{PosTri}(h) \quad (3.15)$$

where h refers to the utterance hypothesis and r_1, r_2 and r_3 are the linear combination weights tuned by grid search.

3.4.3 Handling False Alarms

The ED-EC framework is a two-pass post-processing system. The first pass is error detection and the second pass is error correction. A common characteristic of multi-pass systems is that the mistakes made in the preceding pass will propagate to the subsequent pass. Propagation of errors affects system stability and effectiveness. For the ED-EC framework, neither error detection nor error correction can be guaranteed to be perfect. The error detection procedure labels certain correct characters as erroneous and passes them to the subsequent error correction procedure. The error correction procedure may transform some of these false errors into further (new) misrecognitions. This reduces the benefit of applying the ED-EC framework. If the number of new misrecognitions introduced is larger than the number of recognition errors corrected, adopting the framework will degrade overall recognition performance.

We believe that for multi-pass systems, designing particular algorithms to handle the propagation of mistakes is meaningful and beneficial. In this study, we propose a special mechanism to deal with the propagation of error-detection false alarms for the ED-EC framework. As described in previous subsections, the error correction procedure of the framework corrects the detected errors by using the advanced LM to score and rank the utterance hypotheses in the new search networks. The special mechanism to handle false alarms accepts or

rejects the results of error correction based on the error-detection confidence scores and the LM scores (i.e., the scores assigned by the advanced LM to the utterance hypotheses in the new networks). Figure 3.5 illustrates the procedure of the special mechanism. Note the recognized utterance is one of the utterance hypotheses in the new network.

```
1 For each detected erroneous character (confidence score is x)
2       s1 = the LM score of the top-ranking utterance hypothesis in the network
3       s2 = the LM score of the recognized utterance
4       t = s1 -s2
5       For t > f(x)   ( f(x) is the threshold function)
6             Accept the correction result for the target error
7       Otherwise
8             Reject the correct result (i.e., keep the target character unchanged)
```

Figure 3.5: The special mechanism to handle error-detection false alarms

The main idea behind this special mechanism is to only accept an error correction result when the correction seems reliable. We define the threshold $f(x)$ as a function that depends on the confidence score x based on an intuitive expectation. That is, the smaller the x, the larger the threshold. In other words, the more likely the detected character is a false error, the more conservative we tend to be in accepting any modification.

We use a data-driven approach to estimate $f(x)$. This method attempts to find the suitable form of $f(x)$ by analyzing the relationship between the confidence scores and the optimal thresholds on development data, as shown in the steps below:

Given a set of detected erroneous characters along with their confidence scores,

1) Divide the full range of confidence scores (i.e., from 0.5 to 1) into a sequence of segments: $s_0, s_1, \ldots s_n$. Each segment is then represented by its middle point x_i. For example, if s_i is from 9.80 to 9.90, x_i is 9.85.
2) Create a bin b_i for each s_i to store those erroneous characters detected with confidence scores falling into the range of s_i.
3) For each bin b_i, use a grid search to find a threshold f_i that provides the optimal correction performance on the detected errors stored in b_i.
4) Analyze the relationship between f_i and x_i to estimate a simulation function $f(x)$.

The algorithm above groups detected errors with similar confidence scores and then finds an optimal threshold for each group. Thus, the dependency between the threshold and confidence score can be modeled. The experiments using the development data to estimate *f(x)* will be described later in Section 4.3.2.

3.5 Chapter Summary

This chapter proposes an ED-EC framework to improve Mandarin LVCSR with advanced linguistic knowledge sources. Based on the fact that substitutions constitute the majority of character recognition errors for Mandarin recognition, this framework adopts the simplifying assumption that all character errors are substitutions. The framework consists of two parts: (1) it uses an error detection procedure to label each character in the output of a baseline recognizer as either correct or erroneous and (2) it applies an error correction procedure to correct the detected erroneous characters. The error detection procedure involves a verifier based on generalized word posterior probabilities. The error correction procedure first creates a candidate list of character alternatives for each erroneous character. Connecting these candidate lists with utterance context leads to new search networks for error correction. An advanced linguistic model combining inter-word mutual information, word trigrams and POS trigrams is then adopted to rank all utterance hypotheses in the new search networks. The candidate characters in the top-ranking hypotheses are the error correction results. Since the previous error detection procedure is imperfect, correct characters may be wrongly labeled as erroneous and be "corrected" to new misrecognitions by the error correction procedure. To handle this problem, we introduce an additional mechanism to accept/reject error correction results based on the confidence levels of error detection and the scores from the advanced linguistic model.

Chapter 4

Experiments

This chapter presents the experiments used to develop and evaluate the Mandarin error detection and correction (ED-EC) prototype proposed in Chapter 3. We first describe the organization of the data in Section 4.1. Separate development sets are adopted for developing the baseline recognizer, the error detection procedure and the error correction procedure. Two independent test sets are used in evaluation.

We then present the baseline recognizer in Section 4.2. The baseline recognizer uses state-of-the-art techniques, including cross-word triphone acoustic modeling and trigram language modeling. We train this recognizer with an ample quantity of training data to ensure that any possible improvement brought by the ED-EC framework is meaningful. Otherwise, the same improvement may be achieved simply by using more data in acoustic modeling and/or language modeling.

We outline the development of the ED-EC prototype in Section 4.3. The development of the error detection procedure and the error correction procedure are addressed in turn. For error detection, we list the steps we used to develop the procedure, along with details such as the assignment of reference labels and the tuning of the word verifier. For error correction, we present various development issues including the pruning of candidate lists, the application scope of the error correction procedure, the training of the advanced linguistic model and the estimation of the threshold function in the mechanism to handle false alarms.

Finally, in Section 4.4, we present the evaluation results of the prototype in terms of the reduction in recognition error rate. The baseline performance, the error detection performance, the error correction performance and the overall performance of the framework are described.

4.1 Data Organization

We conduct experiments on the task of Mandarin dictation. We select the general domain as the target domain, attempting to (1) train a general-domain state-of-the-art baseline recognizer, (2) train a general-domain advanced LM and (3) develop the proposed ED-EC prototype with the aim of using the advanced LM to benefit general-domain LVCSR.

Two general-domain Chinese text corpora are used in this study (see Table 4.1). The first corpus (LM_Rec_Set) was collected by Microsoft Research. This corpus contains a total of 28 gigabytes of text and is well balanced across a variety of domains. We use this large corpus to train the trigram model for the baseline recognizer. The second corpus (Lm_Corr_Set) consists of the text from the People's Daily and Xinhua newswire in the LDC corpus *the Mandarin Chinese News Text corpus*. This corpus is a text set of about 340 megabytes. We train the advanced LM on this corpus. Note that we choose the relatively small corpus Lm_Corr_Set instead of the large corpus Lm_Rec_Set for training the advanced LM; this facilitates the modeling of sophisticated linguistic knowledge sources.

LM Training Corpora	Name	Domain	Number of Characters
For Baseline Recognizer	Lm_Rec_Set	General	$14 * 10^9$
For ED-EC Framework	Lm_Corr_Set	General	$151 * 10^6$

Table 4.1: Organization of text corpora

All speech data involved in this study are read speech recorded in a clean environment. There are two speech corpora available. One is a standard general-domain test set provided by Microsoft Research Asia [15]. This test set is referred to as TestSet_G in this study and is used to evaluate the effectiveness of the framework. The other is a separate large corpus originally recorded for acoustic modeling at Microsoft Research. This corpus is a mixture of novels, news and reports. Because novels constitute the majority of the content, we refer to this corpus as novels-domain data in this paper for simplicity. The reference text of this novels-domain corpus is included in the LM training corpus (Lm_Rec_Set) for the baseline recognizer, but it occupies only 0.1% of LM_Rec_Set. Note that this will not affect the generality of LM_Rec_Set since LM_Rec_Set is balanced across different domains.

We analyze the discrepancy in domain between the two speech corpora (i.e., the general-domain corpus TestSet_G and the novels-domain corpus). Using a general trigram trained on Lm_Rec_Set with a 60,606-word lexicon (i.e., the recognizer lexicon that will be described in the next section), the perplexity [22] of the general-domain corpus (TestSet_G) is 463, while the perplexity of the novels-domain corpus is 1528. The difference in perplexity between the two corpora is large, indicating that the domain mismatch between the novels-domain corpus and the target general application data is substantial.

Since we aim to use the ED-EC framework to benefit general-domain LVCSR, the development speech sets should preferably be in the general domain. However, owing to limited data availability, we had to choose development data in the domain of novels. We divide the large novels-domain speech corpus mentioned above into four disjoint subsets randomly. These subsets serve different purposes, as follows:

1) BaseRec_Set: the training data for the acoustic models in the baseline recognizer
2) ErrDect_Set: the development set for the error detection procedure in the framework
3) ErrCorr_Set: the development set for the error correction procedure in the framework
4) TestSet_N: an additional test set, which is adopted to evaluate the influence of using novels-domain speech during development on the generality of the framework

		Name	Domain	Size
Development Sets				
Baseline Recognizer		BaseRec_Set	Novels	about 700 hours (458 K utterances) (14 M characters)
ED-EC Framework	Error Detection	ErrDect_Set	Novels	2,000 utterances (31,490 characters)
	Error Correction	ErrCorr_Set	Novels	8,000 utterances (125,463 characters)
Testing Sets				
		TestSet_G	General	500 utterances (9,572 characters)
		TestSet_N	Novels	4,000 utterances (62,691 characters)

Table 4.2: Organization of speech corpora

The details of the four subsets are shown in Table 4.2. For the baseline recognizer, the development set BaseRec_Set is large enough for acoustic modeling. Compared with the error detection procedure, the error correction procedure is assigned a larger development set because there are more parameters to train in error correction than in error detection, as will be discussed later in this chapter.

For the baseline recognizer, adopting novels-domain acoustic training data (BaseRec_Set) will have little, if any, influence on the generality of the recognizer. This is because BaseRec_Set is sufficient to estimate acoustic parameters. With acoustic models that model pronunciation well, the domain of the recognizer is determined by the domain of the LM in the recognizer. For the ED-EC framework, using novels-domain development data (ErrDect_Set and ErrCorr_Set) instead of general-domain development data will have an impact on the generality of the framework. However, we estimate that the influence is limited. In the ED-EC framework, the power to correct recognition errors mainly comes from the advanced LM, which is in the general domain, in the sense that the detection of errors and the creation of candidate lists only provide the basis (i.e., the new search networks) for the application of the advanced LM.

4.2 Developing the Baseline Recognition System

In this chapter, we use a general-domain recognizer as the baseline recognition system for the ED-EC framework. This recognizer is trained with state-of-the-art techniques on abundant training data. The recognizer adopts a 60,606-word lexicon, in which 6,763 commonly used characters are included as single-character words.

For acoustic modeling, gender-independent cross-word triphone mixture Gaussian tied-state HMM models are trained on 700-hour speech (BaseRec_Set). The speech signal is sampled at a rate of 16 kHz. The analysis frames are 25 ms wide, with a shift of 10 ms. A 39-dimensional feature vector that includes log energy, 12-dimensional cepstral coefficients and their first- and second-order time differences is extracted for each framework. The HMM models are set to have three emitting states and left-to-right topology. We first train flat single Gaussian mixture component monophone models, obtaining 97 monophones in total. We then

create 5,000 speech tied-states using a decision-tree clustering method. Through standard iterative mixture splitting [15], 36 Gaussian mixture components with diagonal covariance matrices are obtained for each tied-state as the final acoustic models.

For language modeling, a word trigram model is trained on a 28G (disk size) text corpus (LM_Rec_Set). Since LM_Rec_Set is in the general domain, the trained word trigram model is also general. In the training of the word trigram model, the absolute discounting algorithm is chosen to perform smoothing.

The Hapivite decoder [91] is used to combine the acoustic/language models into the recognition network and to perform decoding. We use the decoder to recognize all the remaining speech corpora involved in this study. The decoder generates an automatic transcription, known as a recognized utterance, for each input speech utterance. The recognized utterance can be viewed as either a word sequence or a character sequence with word delimiters inserted. The decoder also delivers the recognition lattice generated during the decoding for each utterance. The lattice provides information for the subsequent procedures of error detection and error correction.

4.3 Framework Development

4.3.1 Developing the Error Detection Procedure

We develop the error detection procedure using the corresponding development set (ErrDect_Set) which has 2,000 utterances in total. The proposed error detection algorithm sequentially detects erroneous words and erroneous characters. We first verify the two underlying assumptions: (1) all recognition errors are substitutions and (2) all characters in an erroneous word are wrong. Experiments on ErrDect_Set show that both assumptions are plausible. Substitutions account for 93.5% of the recognition errors, and 88.4% of the characters in the erroneous words are in fact erroneous. These observations indicate that the two assumptions, which are made to reduce system complexity, will cause limited performance loss. Development of the error detection procedure is summarized in the following steps. Details about the first two steps will be presented later in this subsection.

Step 1. Assign each recognized word/character a reference label (i.e., "actually correct" or "actually erroneous") to facilitate the tuning and evaluation of the error detection schema.

Step 2. Tune the parameters of the word verifier through ten-fold cross validation on ErrDect_Set.

Step 3. Train a "final word verifier" on the whole ErrDect_Set, using the optimal parameters identified in *Step 2*.

Step 4. Implement the error detection procedure. For each input utterance, the error detection procedure first applies the final word verifier obtained in *Step 3* to detect erroneous words and then labels all characters in the detected erroneous words as erroneous.

The error detection procedure implemented in *Step 4* is used to fulfill all the error detection tasks that are involved in the subsequent experiments in this chapter. These experiments include the development of the error correction procedure and the evaluation of the ED-EC framework.

- **Assigning reference labels**

Based on the manual transcripts of speech utterances, we assign a reference label to each character in the recognized utterances. We use the Viterbi algorithm to identify the character errors by mapping every recognized utterance to the corresponding reference utterance. Based on the assumption that all misrecognitions are substitutions, we ignore the deletions and view insertions in the same way as substitutions. Accordingly, we label all substitutions and insertions as erroneous and label the remaining characters as correct. For example, the speech utterance "他清醒地意识到这一点" ("He clearly realized this point") is wrongly recognized as "他/警醒/地/意识/到/了/这/点". The mapping result from Viterbi is shown in Figure 4.1. There are three character recognition errors: one substitution 警, one insertion 了 and one deletion. We label the substitution 警 and the insertion 了 as erroneous. Other characters are labeled as correct. Note that with such a labeling method, an insertion becomes a kind of erroneous character that may be detected but will never be corrected. In error correction, insertions that are labeled as erroneous will not be deleted since they are viewed as substitutions.

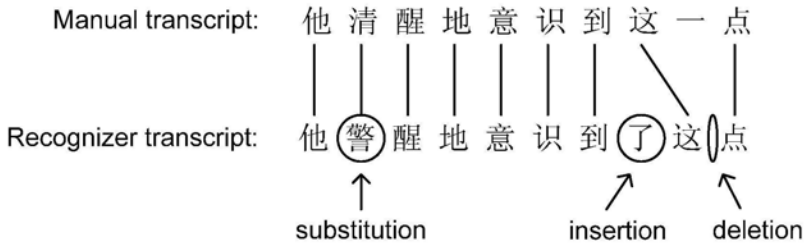

Figure 4.1: Identifying misrecognitions by the Viterbi algorithm

We then assign a reference label to each recognized word. This facilitates the tuning of the word verifier. If a recognized word contains no substitutions or insertions, we label it as correct. Otherwise, we label the word as erroneous. For the example above, the reference labels for words are shown in Figure 4.2. The words 警醒 and 了 are labeled as erroneous, since the two words contain one substitution and one insertion, respectively.

他/警醒/地/意识/到/了/这/点
C E C C C E C C

C=Correct E=Erroneous

Figure 4.2: Obtaining reference labels for the recognized words.

- **Tuning the word verifier**

The word verifier has been designed to classify each word as either correct or erroneous based on the Generalized Word Posterior Probability (GWPP) of the focused word using the Naïve Bayes algorithm [136]. The GWPP value of a word is calculated from the recognition lattice using two parameters: the acoustic model weight α and the language model weight β. The task of tuning is to identify the optimal α and β to calculate the GWPP. Once the optimal α and β are found, the word verifier can be trained on all words in the development set (ErrDect_Set) using the two weights.

We search for the optimal α and β by grid search. For every possible weight pair (α, β), we evaluate its performance on word verification by ten-fold cross validation on ErrDect_Set. The (α, β) having the best performance is then identified and used as the optimal weights.

4.3.2 Developing the Error Correction Procedure

The error correction procedure begins by creating a candidate list of character alternatives for every detected erroneous character. In this way, we build a new search network for each speech utterance. The utterance hypotheses in the new networks are then ranked by an advanced linguistic model combining inter-word mutual information (MI), word trigrams and part-of-speech (POS) trigrams. The candidate characters in the top-ranking hypotheses are the error correction results. To handle false alarms of error detection, an additional mechanism is also designed to accept/reject the correction results. This subsection presents the implementation details of the error correction procedure. All the experiments are performed on the 8,000 utterances of the corresponding development set (ErrCorr_Set).

- **Details of the candidate creation approach**

The algorithm to create a candidate list for each erroneous character has been proposed earlier in Section 3.4.1. For each error, the algorithm (1) selects candidate alternatives from the corresponding recognition lattice, (2) ranks the candidates based on their generalized character posterior probabilities, and (3) prunes the candidate list to only keep the top N candidates.

One problem is that the candidate creation algorithm requires the starting and ending times of character hypotheses in recognition lattices. However, the HapiVite decoder only annotates starting and ending times for word hypotheses. Forced alignment can be applied to identify the accurate times for character hypotheses, but the resulting computational load will be considerable. To maintain system efficiency, we simply assume that all character hypotheses contained in a word hypothesis have equal duration. Thus the starting/ending times of each character hypothesis can be derived from the starting/ending times of the word hypothesis containing it. This assumption is plausible because the speech rate is usually consistent over a short time period. The time expended to pronounce each character in a word (e.g., 苹果 "apple"; 珠穆朗玛峰 "Mount Everest") is normally similar.

With this problem addressed, we proceed to decide N (i.e., the scale of pruning) on the 23,346 character substitution errors in the ErrCorr_Set, which contains 125,463 characters in total. To facilitate the discussion, we define the reference coverage rate (R_L) of candidate lists as

the equation below. Reference characters refer to the correct characters given by the manual transcription.

$$R_L = \frac{\text{number of candidate lists that contain the corresponding reference characters}}{\text{total number of candidate lists}} \quad (4.1)$$

We create and rank candidate lists for the focused set of erroneous characters. The reference coverage rate R_L is 64.5%. Among the candidate lists, the largest one (i.e., {前,显,... 闲}) contains 52 characters. The average size of the candidate lists is 10.6. For all the candidate lists that contain the corresponding references, the reference character is ranked 4th on average, and the lowest position of a reference is 27th.

It is possible to use adaptive N in pruning. In this preliminary work, we adopt a constant N for simplicity. Based on the observations mentioned above, we set N=20. After pruning the candidate list size to 20, the R_L becomes 64.4%. The drop in R_L is only 0.1%.

- **Application scope of the error correction procedure**

Based on the error detection results, we cluster recognized utterances into three utterance subsets: a correct set, a lightly erroneous set and a seriously erroneous set. The correct set contains those utterances in which all characters are labeled as correct. The lightly erroneous set includes those utterances that contain one to four detected erroneous characters. The remaining utterances with more than four detected errors are assigned to the seriously erroneous set. Theoretically, the error correction procedure should be applied on both the lightly erroneous set and the seriously erroneous set to correct the errors detected. In this study, we focus the application of the error correction procedure on the lightly erroneous utterance set due to efficiency considerations.

The error correction method simply enumerates all the utterance hypotheses in the new search networks for ranking. When the number of detected erroneous characters for an utterance increases, the number of utterance hypotheses in the new search network grows exponentially. When the number of detected errors is larger than four, the computational load of ranking utterance hypotheses becomes overwhelming, making the correction prohibitive. Thus, this work only attempts to correct errors in the lightly erroneous utterance set. Note that the current framework is only an initial prototype. If a more efficient algorithm were to be developed to identify the top-ranking utterance hypothesis, processing utterances with more than four errors would also be possible.

We apply the error detection procedure on ErrCorr_Set and cluster the utterances in this development set into three sets based on the number of detected erroneous characters. The correct set, lightly erroneous set and seriously erroneous set contain 2724, 3592 and 1684 utterances respectively. The lightly erroneous set makes up nearly 50% of utterances. Among the utterances containing detected errors, more than 2/3 of these utterances are deemed lightly erroneous. Since the error correction procedure will focus on the lightly erroneous sets of utterances in application, the training of the error correction procedure concentrates on the lightly erroneous utterances in ErrCorr_Set.

- **Training the advanced linguistic model**

To fairly compare the error correction effects of the linguistic knowledge sources, we train all three individual LMs (i.e., MI, word trigram and POS trigram) on the same training corpus (Lm_Corr_Set). While the training data are identical, the lexicons used are different. The MI and word trigram models are trained with a LDC lexicon [153] that has 44,402 words. For the POS trigram model, we first use an 11,908-word lexicon to segment the training utterances into word sequences. A tagger [89] that maps the 11,908 words into 86 POS tags according to certain rules is then applied to transfer the word sequences into POS sequences. A trigram model is trained with the POS sequences using a lexicon that contains the 86 POS tags. The CMU LM toolkit [22] is adopted to perform the trigram training for both the word trigram and POS trigram models. The great flexibility in selecting lexicons for linguistic modeling is one advantage of choosing the character instead of the word as the basic unit of recognition errors. This allows linguistic knowledge sources to be modeled independently of each other.

With the individual models trained, we tune the interpolation weights r_1, r_2 and r_3 in the combination equation (i.e., Equation 3.15) that linearly interpolates the individual models. The optimal r_1, r_2 and r_3 are identified by grid search on the lightly erroneous utterances in ErrCorr_Set. The final advanced linguistic model is built with the optimal interpolation weights using the combination equation.

- **Developing the mechanism to handle false alarms**

For a detected erroneous character with confidence score x (x ranges from 0.5 to 1), the mechanism to handle false alarms (see Figure 3.5) accepts the error correction result if t, the

difference in scores between the top-ranking utterance hypothesis and the recognized utterance, is larger than a threshold *f(x)*. Otherwise, the error correction result is rejected and the original recognized character remains unchanged. Using the data-driven approach proposed in Section 3.4.3, we estimate *f(x)* based on the lightly erroneous utterance subset in ErrCorr_Set. This focused utterance subset contains 8,443 detected erroneous characters in total. The experiments show that our expectation holds true: for most cases, the smaller x is, the larger the *f(x)*. However, the data set is not large enough to robustly estimate a smooth *f(x)*. Based on our observations, we adopt the equation below:

$$f(x) = \begin{cases} f_a & \text{if } x \geq 0.9 \\ f_b & \text{if } 0.5 \leq x < 0.9 \end{cases} \tag{4.2}$$

where f_a and f_b are two constants, and $f_a < f_b$. f_a and f_b are identified by grid search on the 8,443 detected erroneous characters on which we focus.

4.4 Framework Evaluation

We evaluate the ED-EC framework on the two test sets TestSet_G and TestSet_N. TestSet_G is in the general domain while TestSet_N is in the domain of novels. In this section, we first present the baseline performance in Section 4.4.1. We then discuss the error detection performance and the error correction performance in Section 4.4.2 and Section 4.4.3, respectively. Finally, the overall performance of the framework is given in Section 4.4.4.

4.4.1 Baseline Performance

To facilitate the discussion, we first define the Character Error Rate (CER) as follows:

$$CER = \frac{Num_{Substitution} + Num_{Insertion} + Num_{Deletion}}{Num_{all}} \tag{4.3}$$

where Num_{all} is the number of characters in the given recognized utterances; $Num_{Substitution}$, $Num_{Insertion}$ and $Num_{Deletion}$ are the numbers of substitutions, insertions and deletions respectively. All substitution/insertion/deletion errors refer to character recognition errors.

The baseline recognizer achieves 8.9% CER on TestSet_G and 19.9% CER on TestSet_N. For TestSet_N, the CER is relatively high due to the discrepancy in domain between the recognizer and the application data. The novels-domain utterances in TestSet_N are difficult to handle using the general language model in the recognizer.

The numbers of substitutions, insertions and deletions for the two test sets are listed in the table below. For both test sets, more than 90% of character errors are substitutions. This is consistent with the results reported in previous works [34], [88], further demonstrating that substitutions tend to occupy the majority of all character errors for Mandarin LVCSR. As discussed in Section 3.1, the reason is that Mandarin has a relatively simple syllable structure and the decoding procedure rarely makes mistakes in segmenting signals into syllable sequences. This observation also indicates the feasibility of the assumption "all character errors are substitutions" made for the ED-EC prototype in Section 3.1. With this assumption, less than 10% of character errors are missed, but the complexity of handling insertions/deletions is avoided.

	Number of Utterances	Number of Characters	**CER %**	Number of errors		
				Subt.	Inst.	Delt.
TestSet_G	500	9,572	**8.89**	766	15	70
TestSet_N	4,000	62,691	**19.86**	11,721	213	516

Table 4.3: Performance of the baseline recognizer. Subt. Inst. and Delt. refer to substitutions, insertions and deletions respectively

The proposed ED-EC prototype also assumes that all characters in an erroneous word are wrong, as discussed in Section 3.3.2. The experiments show that in erroneous words (i.e., the recognized words that contain substitutions and/or insertions), 84.5% and 88.3% of characters are wrong for TestSet_G and TestSet_N, respectively. This is consistent with our expectation.

In addition, we notice that when a substitution error occurs, the reference character (i.e., the correct character for the corresponding speech signal) is often recognized as a character with similar but different pronunciation. For example, the percentage of the substitutions that have the same tonal syllables (e.g., /ta2/) with the corresponding references is only 25.5% and 22.5% for TestSet_G and TestSet_N respectively. The remaining substitutions have different tones or even different base syllables (e.g., /ka/) with the corresponding references, as illustrated in the

table below. This phenomenon takes place because language models play an important role in state-of-the-art recognizers such as the baseline recognizer. A hypothesis with relatively low acoustic likelihood and high linguistic likelihood may be selected in decoding as the recognition result. Note that a character can have multiple pronunciations in Mandarin. In this section, for simplicity, we state that character A has the same tonal/base syllable with character B if one of the possible tonal/base syllables for A matches one of the possible tonal/base syllables for B. Since a character in the utterance context is pronounced as one tonal/base syllable only, the statistics reported in Table 4.4 for "same tonal/base syllables" are larger than the ones in reality.

	Number of Substitutions			
	Total	*Substitution and the corresponding reference character have:*		
		Same Tonal Syllable	Same Base Syllable and Different Tone	Different Base Syllable
TestSet_G	766	195 (25.5%)	149 (19.5%)	422 (55.1%)
TestSet_N	11,721	2,605 (22.2%)	2518 (21.5%)	6598 (56.3%)

Table 4.4: Acoustic similarity between the substitutions and the corresponding reference characters

Table 4.4 also shows that more than 55% of substitutions have different base syllables with the corresponding references for both test sets. This means that if we create a list for each substitution by including all characters sharing the same base syllable with the substitution, the chance that the list contains the reference character is less than 45% on average for both test sets. In other words, in the error correction procedure of the framework, creating a candidate list for each error by including all homophonous characters may not be a good idea. Note that the proposed candidate creation algorithm provided a reference coverage rate (i.e., the rate that a candidate list includes the corresponding reference character) of above 70% for misrecognitions, as will be discussed later in Section 4.4.3.

4.4.2 Performance of Error Detection

To evaluate the performance of error detection, we define the Detection Error Rate (DER) for a certain instance (character/word/utterance) as follows:

$$\text{DER} = \frac{\text{the number of incorrectly classified instances}}{\text{total number of instances}} \tag{4.4}$$

We apply the error detection procedure to label each character in the output of the baseline recognizer as either erroneous or correct. The detection error rate on TestSet_G is 6.9%. The DER on TestSet_N is much higher, at 14.5%. Details are in Table 4.5.

	TestSet_G (DER: 6.9%)		TestSet_N (DER: 14.5%)	
Classified as →	Correct	Erroneous	Correct	Erroneous
Correct	8,443	293	46,849	3,605
Erroneous	364	417	5,445	6,489

Table 4.5: Performance of detecting erroneous characters

Table 4.5 shows that for both TestSet_G and TestSet_N, more than half of the misrecognized characters are successfully detected as erroneous. At the same time, more than 1/3 of detected erroneous characters are actually correct for both test sets. The percentages of these false errors (i.e., false alarms) among all detected errors are 41.3% (293/710) and 35.7% (3,605/10,094) for TestSet_G and TestSet_N, respectively. These indicate that false alarms will have non-negligible influence on the subsequent error correction procedure.

We further measure the performance of error detection by the balanced F-measure, which can be written as follows:

$$F = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{4.5}$$

The balanced F-measures of the two test sets are relatively similar. On TestSet_G and TestSet_N, the balanced F-measures for correct characters are 96.3% and 91.2%, respectively, while the balanced F-measures for erroneous characters are 61.0% and 58.9%, respectively. Note that the error detection procedure is trained on ErrDect_Set, which is in the domain of novels. The observation that the error detection procedure performs better on the general TestSet_G than on the novels-domain TestSet_N shows that the proposed error detection algorithm is robust to domain shift.

We proceed to evaluate the ability of the error detection procedure to identify those utterances that have been recognized correctly. We label each utterance as either correct or erroneous based on the error detection results. If an utterance contains no detected errors, it is labeled as correct. Otherwise, it is labeled as erroneous. The performance in terms of detecting correct utterances is shown in Table 4.6. More than 4/5 of correct utterances are detected on both test sets. The DERs for utterances are 23.2% and 18.6% for TestSet_G and TestSet_N respectively.

	TestSet_G (DER: 23.2%)		TestSet_N (DER: 18.6%)	
Classified as →	Correct	Erroneous	Correct	Erroneous
Correct	165	27	794	175
Erroneous	89	219	568	2,463

Table 4.6: Performance of detecting correctly recognized utterances

We cluster the recognized utterances in each test set into three subsets. The utterances with 0, 1-4 and >4 detected erroneous characters are assigned to the utterance subsets of correct, lightly erroneous and seriously erroneous, respectively. This subset division is illustrated in Figure 4.3. Table 4.7 provides the details for the utterance subsets. We observe that in either test set, the seriously erroneous subset does have the relatively highest CER, while the correct subset has the relatively lowest CER. This reflects the effectiveness of the proposed error detection procedure.

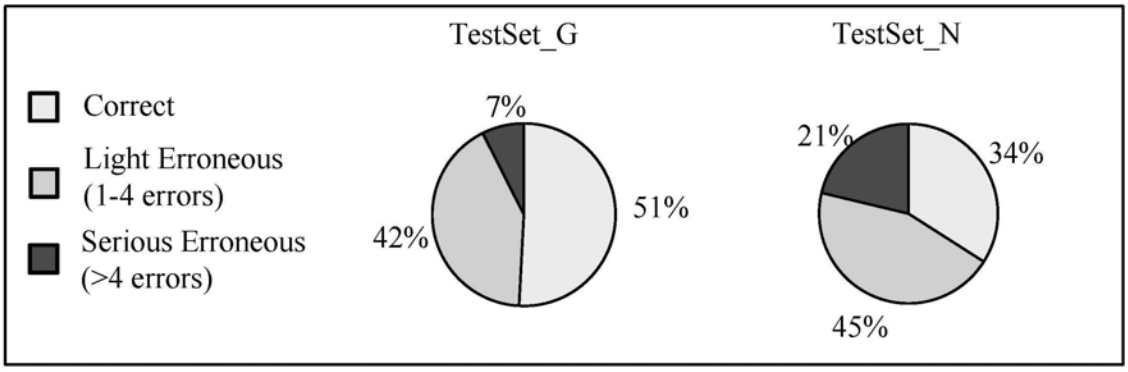

Figure 4.3: Division of each test set into three subsets

	In TestSet_G		In TestSet_N	
	Number of Utterances	CER %	Number of Utterances	CER %
All Utterances	500	8.9	4,000	19.9
Utterance Subset of Correct	254	3.3	1,362	6.4
Utterance Subset of Lightly Erroneous	209	11.9	1,790	20.4
Utterance Subset of Seriously Erroneous	37	27.7	848	37.1

Table 4.7: Details of the utterance subsets

4.4.3 Performance of Error Correction

We apply the error correction procedure on the lightly erroneous utterance subset for each test set. There are 209 and 1,790 such utterances for TestSet_G and TestSet_N respectively.

The error correction procedure first creates a list of candidate characters for each detected error. The reference coverage rates (RCRs) of the candidate lists are presented in the table below. For false errors, the reference coverage rates are 1. This is because, with the proposed candidate creation algorithm, the candidate list of a detected error always includes the corresponding recognized character. For true errors, more than 70% of candidate lists include the corresponding references on both test sets. This result is acceptable for a preliminary work. In addition, the RCR for true errors is relatively higher on the general-domain test set TestSet_G, although the development data (ErrCorr_Set) are in the domain of novels. This implies that the candidate creation procedure is insensitive to the difference in domain between the development data and test data.

	Number of Detected Errors	True Errors		False Error	
		Number	RCR	Number	RCR
TestSet_G	468	272	81.6%	196	100%
TestSet_N	4,385	2,793	73.3%	1,592	100%

Table 4.8: Reference coverage rates (RCRs) of the candidate lists that were created for the detected errors in the utterance subsets of lightly erroneous

We also analyze the acoustic similarity between the candidate characters in the created lists for errors. For clarity, we focus here on true errors. In a candidate list, the percentage of the characters that have the same pronunciation (i.e., the same tonal syllable) with the reference character is 21.0% on average for TestSet_G and 22.2% on average for TestSet_N. The percentage of the characters that have the same base syllable but different tone as the reference is on average 17.7% and 18.8% for TestSet_G and TestSet_N respectively. The remaining candidate characters have different base syllables as the reference character. However, the remaining candidate characters are more-or-less acoustically similar to the reference since they are all recognition hypotheses for the corresponding speech signals.

With the created candidate lists, the error detection procedure proceeds to correct errors by distinguishing among the competing candidate characters. We notice that the error correction performances on TestSet_G are consistently better than those on TestSet_N. Of the true errors, 34.9% are corrected on TestSet_G, while only 26.3% are corrected on TestSet_N. Of the false errors, 77.6% and 73.9% remain correct (i.e., avoid being "corrected" into new misrecognitions) on TestSet_G and TestSet_N respectively. Regarding the overall CER on the focused data sets (i.e., the lightly erroneous utterance subsets), the relative CER reduction on TestSet_G is almost twice that on TestSet_N, as shown in Table 4.9. The fact that the error correction procedure works better on the general-domain test set indicates that (1) the advanced linguistic model is critical for the effectiveness of error correction and (2) the domain of development speech has little influence on error correction. We will further analyze the performance of error correction in greater detail in Section 5.1.

	Baseline CER %	CER after Correction %	Relative Reduction %
TestSet_G	11.9	10.6	10.9
TestSet_N	20.4	19.2	5.9

Table 4.9: Error correction performance, evaluated on the lightly erroneous utterance subsets

4.4.4 Overall Framework Performance

In this subsection, we view the ED-EC framework as a black-box tool to improve the performance of the baseline recognition system. To analyze the overall effectiveness of the

proposed ED-EC prototype, we evaluate the CER reduction on all data for each test set. The results are included in the table below:

	Baseline CER %	CER after Correction %	Relative Reduction %
TestSet_G	8.89	8.36	6.0
TestSet_N	19.86	19.35	2.6

Table 4.10: Error correction performance, evaluated on the full test sets

As shown in Table 4.10, the relative CER reductions over the recognizer baselines are 6.0% and 2.6% for TestSet_G and TestSet_N, respectively. The good performance on the general-domain test set TestSet_G is especially encouraging since the objective of the ED-EC framework is to improve recognition performance on general-domain application data. Note that in this study, we adopt the novels domain for the development speech sets owing to limited data availability. Better performance can be expected on TestSet_G when general-domain development sets become available.

In this preliminary work, the proposed error correction procedure focuses on the lightly erroneous utterance subsets to correct errors. Thus, the correct and seriously erroneous utterance subsets remain unchanged after applying the ED-EC prototype. Designing a more advanced error correction algorithm that can also correct errors in seriously erroneous utterances may further improve the performance of the ED-EC framework. However, handling seriously erroneous utterances may introduce considerable computational load.

4.5 Chapter Summary

This chapter describes the implementation and evaluation of the proposed ED-EC prototype. We first train a state-of-the-art baseline recognizer on ample training data. We then develop the error detection and error correction procedures on separate development sets. For error correction, the size of the candidate lists created for detected errors is pruned to 20 based on our observations on the corresponding development set. As a consequence, the application scope of the error correction procedure is limited to those utterances with one to four detected erroneous characters to constrain the computational load. This limitation may be removed by designing a more

efficient algorithm to rank competing hypotheses during error correction. We evaluate the developed ED-EC framework on two independent test sets: a general-domain test set and a test set in the domain of novels. The results are encouraging. Non-trivial CER reductions are observed on both test sets. For the general-domain test set, a 6.0% relative CER reduction over the recognizer baseline is observed on the whole test set, whereas on the focused utterances of error correction, the CER reduction is as high as 10.9%. In the next chapter, we will analyze the effectiveness of the ED-EC framework in detail.

Chapter 5

Analyses of Framework Effectiveness

Previous chapters have described a prototype of the error detection and correction (ED-EC) framework for Mandarin LVCSR. This chapter examines this Mandarin prototype in detail to analyze the effectiveness of the ED-EC framework. We first analyze the ability of the ED-EC framework to reduce the baseline recognition error rate in Section 5.1. In the framework, while error detection provides the basis for error correction, error correction is the step that reduces the error rate. The effect of error correction is influenced by three major factors: (1) the linguistic knowledge sources utilized, (2) the search space created and (3) the mechanism adopted to handle false alarms of error detection. The last two factors depend on the error detection performance. This chapter demonstrates that different linguistic knowledge sources have different error correction capabilities. The error correction effect also relies on search space properties, such as the number of competing hypotheses and the amount of misleading information (e.g., unexpected erroneous characters in the hypotheses). Handling false alarms of error detection by an additional mechanism substantially enhances the framework performance, indicating that designing special strategies to control error propagation between procedures is beneficial for the overall system. In Section 5.2, we proceed to propose a formula to describe the ability of the ED-EC framework in error rate reduction. We also present the performance upper bounds in Section 5.3. The upper bounds for improving error detection or candidate creation or both are estimated separately. Finally, we discuss the computation efficiency of the framework in Section 5.4. While the error detection procedure is very efficient, the time spent on error correction to process an utterance varies greatly, as it depends on the number of detected errors in the target utterance.

5.1 Factors Affecting Error Correction

5.1.1 Linguistic Knowledge Sources

Different linguistic knowledge sources capture different types of information and thus have different capabilities to correct recognition errors. The proposed Mandarin ED-EC prototype uses mutual information (MI), word trigrams and POS trigrams to capture long distance semantic constraints, local constraints and syntactic constraints, respectively. This subsection compares the error correction capabilities of the three individual linguistic models and their linear combination.

To facilitate the discussion, we first define the Correction Rate (CR) as follows:

$$CR = \frac{Num_{ErrorCorrected}}{Num_{allError}} \tag{5.1}$$

where $Num_{allError}$ is the number of character errors; $Num_{ErrorCorrected}$ is the number of the character errors that are successfully corrected by the error correction procedure.

To clearly analyze the power of these linguistic models in error correction, we assume both error detection and candidate creation are perfect in this set of experiments. We apply the error correction procedure to correct the substitutions in those utterances that contain one to four substitutions. If a created candidate list fails to include the corresponding reference character, we insert this reference into the candidate list by replacing the lowest-ranking candidate with the reference. In this way, all erroneous characters in focus are correctable true errors. The experimental results are as follows:

Test Set	Number of Errors	Correction Rate %			
		MI	Word Trigram	POS Trigram	Combination
TestSet_G	529	46.3	47.6	19.3	57.8
TestSet_N	4,626	44.8	39.9	24.3	54.5

Table 5.1: The abilities of various linguistic knowledge sources to correct erroneous characters, evaluated on the lightly erroneous utterance subsets. Both error detection and candidate creation are assumed to be correct in this experiment.

Table 5.1 shows that the use of MI achieves high correction rates. With the MI model alone, about 45% of errors in focus are successfully corrected for both test sets. Note that the MI model captures the co-occurrence information of an error with all other words in the utterance context. The effectiveness of MI for error correction demonstrates the power of long-distance constraints.

As illustrated in Table 5.1, the word trigram model adopted in error correction is also effective to correct recognition errors, despite the fact that the baseline recognizer has already employed a well-trained word trigram model (Rec_Trigram). One possible reason is that the training corpus (Lm_Corr_Set) for the LMs in error correction is "closer" to the testing data than the training corpus (Lm_Rec_Set) for the baseline trigram. To verify this, we train a new word trigram model (Verify_Trigram) on the corpus Lm_Corr_Set using the same lexicon and training techniques as the baseline word trigram model. In other words, the training procedures of Very_Trigram and Rec_Trigram only differ in the training data. If Lm_Corr_Set is closer to the testing data than Lm_Rec_Set, decoding with Verify_WdTrigram instead of Rec_WdTrigram would lead to better recognizer performance. In this work, for simplicity, we re-rank the 1000-best hypotheses generated by the original baseline recognizer using the Verify_WdTrigram to simulate the Verify_WdTrigram decoding. The results show that the re-ranking hurts the recognition performance on both test sets. The CER increases from 19.9% to 25.4% on TestSet_N and from 8.9% to 16.5% on TestSet_G. This shows that the possible reason stated above does not hold. The corpus LM_Corr_Set may contain complementary information but is not "closer" to the testing data. We believe that the benefit of word trigrams for error correction is mainly due to the new search space for error correction, which may include more alternatives for erroneous regions in the signal. We will discuss the search space in detail in the next subsection.

For the POS trigram model, the correction rates are relatively low, as shown in Table 5.1. This is partially because different utterance hypotheses may degenerate to the same tag sequence. We further evaluate the rate that the reference hypothesis is among the top-ranking tag sequences. The rates are 20.6% and 27.6% for TestSet_G and TestSet_N respectively, which are still lower than the correction rates of the MI and the word trigram. The POS trigram model is relatively ineffective for error correction possibly because local syntactic dependencies (i.e., those captured

by POS trigrams) are weak compared with either local word dependencies (i.e., those captured by word trigram) or long-distance constraints (i.e., those captured by MI).

The combined model achieves the highest correction rate at around 55%. On both test sets, the combination brings about a 21.5% relative increase in correction rate over the performance of the best individual model. This shows the advantage of knowledge combination. The current prototype simply uses linear interpolation to combine individual linguistic models. Adopting more advanced combination techniques, such as the maximum entropy approach, may bring further improvement.

We also notice that if we exclude the POS trigram model from the combination, the correction rate of the combined model (i.e., the linear interpolation of the MI and word trigram models) will only drop less than 1% for both test sets. This implies that for the application of the POS trigram model, the benefit in accuracy may not compensate for the related computational load. Whether or not to adopt linguistic knowledge sources in error correction should be a decision that considers the balance between accuracy and efficiency. We will further discuss this point in Section 5.4.

Finally, we compare the performance differences between the two test sets. Except for the POS trigram model, the correction rate of a model for TestSet_G is consistently higher than that for TestSet_N. Note that the individual models in error correction are trained on a general corpus. The semantic knowledge captured by the MI or word trigrams is more effective in correcting errors in a general context. On the other hand, the syntactic information captured by the POS trigram is relatively domain-independent. Since MI and word trigrams play major roles in the linearly combined model, the combined model also performs better on TestSet_G.

5.1.2 Search Space

For each utterance that contains detected erroneous characters, the error correction procedure expands every detected character error into a candidate list of alternative characters and thus generates a sausage-type search space as Figure 5.1. The search space is the basis for the subsequent application of linguistic models (i.e., correcting errors by ranking the competing utterance hypotheses with the aid of linguistic models). The effect of error correction is thus

influenced by the properties of the search space. In this subsection, we analyze two major search space properties: the number of competing hypotheses and the amount of misleading information.

Figure 5.1: A sausage-type search space

The first property, *the number of competing hypotheses*, has an impact on error correction effect since a greater number of competing hypotheses lead to more confusion. When the number of erroneous characters in a recognized utterance increases, the number of utterance hypotheses in the search space grows exponentially and the confusion level increases substantially. To analyze this phenomenon, we conduct a series of experiments. For clarity, we assume that both error detection and candidate creation are perfect, as in Section 5.1.1. Without loss of generality, we utilize the combined model to correct errors in those testing utterances containing one to four substitutions. The results are included in Table 5.2.

		Number of Possible Hypotheses in the search space	Correction Rate %	
			On TestSet_G	On TestSet_N
Errors in those Utterances that Contain *N* Errors	*N*=1	20	63.4	57.7
	N=2	20^2	60.2	56.4
	N=3	20^3	57.1	54.8
	N=4	20^4	50.0	51.6
Total Errors		/	57.8	54.5

Table 5.2: The impact of the size of the search space on error correction, evaluated on the lightly erroneous utterance subsets. Both error detection and candidate creation are assumed to be correct in this experiment.

If an utterance has *N* erroneous characters, the search space created for this utterance contains 20^N utterance hypotheses at most. When *N* grows from 1 to 4, the number of possible utterance hypotheses in the search space increases from 20 to 160,000. The complexity of distinguishing among utterance hypotheses also increases greatly. As a consequence, the

correction rate decreases, as shown in Table 5.2. However, compared with the increase in complexity, the drop in correction rate is limited. The relative reductions in correction rate are only 21.1% and 10.6% on TestSet_G and TestSet_N, respectively. This indicates that the capability of the advanced linguistic knowledge sources to identify the correct utterance hypothesis from the competing ones is robust for the number of competing hypotheses.

The second search space property is *the amount of misleading information.* Misleading information in the search space is introduced by imperfect candidate creation and/or imperfect error detection. The search spaces are created by connecting the candidate lists for detected errors with the corresponding utterance context. If error detection is imperfect, the utterance context may contain missed detections and be misleading. If candidate creation is imperfect, a candidate list may fail to include the corresponding reference character, making all areas related to this candidate list in the search space misleading. The existence of misleading information is unavoidable in real applications, since it is very hard (if not impossible) to achieve perfect performance for both error detection and candidate creation.

The amount of misleading information will affect error correction. When both error detection and candidate creation are perfect, reference utterances are always contained in search spaces, and the task of error correction is to score the reference utterance highest among its competing utterance hypotheses for each utterance. With imperfect error detection or imperfect candidate creation or both, reference utterances may not be included in the search spaces. The error correction task thus reduces to selecting the *best hypothesis* (i.e., the one with the lowest CER) from the competing utterance hypotheses for each utterance. Note that the *best hypotheses* may contain misrecognitions in either utterance context or areas related to the candidate lists. This affects the effectiveness of linguistic models in error correction because linguistic models are trained on an error-free text corpus.

We analyze the impact of misleading information by comparing the error correction effects under scenarios with perfect/imperfect error detection and candidate creation procedures, as shown in Table 5.3. Perfect error detection refers to the procedure that manually labels all character substitutions as erroneous. Perfect candidate creation refers to the procedure that manually inserts reference characters into the candidate lists if they are absent. Imperfect error detection refers to the application of the proposed error detection procedure. Imperfect lattice creation refers to using the proposed candidate creation procedure as is without manually

inserting reference characters (i.e., without any guarantee that the reference characters are present). S1, S2, S3 and S4 are the four scenarios using different error detection and candidate creation procedures, as illustrated in Table 5.3.

	Error Detection	Candidate Creation	TestSet_G			TestSet_N		
			R_T %	R_{T_C} %	R_F %	R_T %	R_{T_C} %	R_F %
S1	Prefect	Prefect	57.8	57.8	/	54.5	54.5	/
S2		Imperfect	39.9	54.5	/	32.8	49.9	/
S3	Imperfect	Prefect	55.5	55.5	59.7	46.2	46.2	58.2
S4		Imperfect	43.8	53.6	58.2	33.5	45.8	56.9

Table 5.3: The influence of misleading information in search space on error correction. Under each of the four scenarios (S1, S2, S3, S4), the error correction procedure was applied on the corresponding lightly erroneous utterance subsets.

We evaluate the effectiveness of error correction in terms of three rates: (1) the correction rate for all true errors, denoted R_T, (2) the correction rate for the correctable true errors, denoted R_{T_C}, and (3) the rate of uncorrected false errors (i.e., those that have not been turned into new errors), denoted R_F. We state that a true error is correctable if the corresponding reference is contained in the candidate list. Otherwise, we refer to the true error as an uncorrectable true error. For false errors, reference characters are always included in candidate lists based on the current candidate creation algorithm. When candidate creation is perfect, all errors are correctable. When error detection is perfect, all detected errors are true errors. In this set of experiments, without loss of generality, we use the advanced language model that combines three individual sources to correct errors. Since this set of experiments concentrates on the impact of search space, the special mechanism to handle false alarms of error detection is not involved. The experimental results are shown in Table 5.3.

Table 5.3 shows that on both test sets, R_{T_C} is the highest under S1 and is the lowest under S4. This demonstrates that the ability to select correct answers from candidate lists decreases when search spaces contain more misleading information, such as misrecognitions in the utterance context and erroneous areas related to those uncorrectable true errors. Similarly, R_F under S3 is always higher than that under S4 because only S4 contains uncorrectable true errors, which are misleading. We can also see that the quality of candidate creation greatly affects the

error correction effectiveness. Applying perfect candidate creation instead of imperfect candidate creation always increases the rates under both perfect and imperfect error detection.

One observation that seems counterintuitive in Table 5.3 is that the R_T under S2 is lower than under S4 for both test sets, although S2 has perfect error detection. For S2 and S4, while the error correction procedures are the same, different error detection procedures lead to different sets of detected errors. In each set of detected errors, true errors are composed of correctable true errors and uncorrectable true errors. For S2, although the correction rate for correctable true errors (i.e., R_{T_C}) is higher, the overall correction rate for true errors (i.e., R_T) is lower because of the higher percentage of uncorrectable true errors (i.e., the lower reference coverage rate of candidate lists), as shown in Table 5.4. This observation further proves that improving the candidate creation algorithm is a worthwhile direction to enhance the error correction effect.

	Error Detection	Candidate Creation	Reference Coverage Rate % for True Errors	
			TestSet_G	TestSet_N
S2	Perfect	Imperfect	73.2	65.7
S4	Imperfect	Imperfect	81.6	72.2

Table 5.4: Reference coverage rates of candidate lists, evaluated on true errors in the corresponding lightly erroneous utterance subsets under a certain scenario (S2 or S4).

The above discussions show that both R_{T_C} and R_F are influenced by misleading information contained in search spaces. R_T is jointly decided by R_{T_C} and reference coverage rate R_L as:

$$R_T = R_{T\_C} * R_L \tag{5.2}$$

5.1.3 False Alarms

The special mechanism to handle false alarms of error detection selectively accepts error correction results to avoid turning false errors into new misrecognitions. This subsection analyzes the benefit of this mechanism for error correction. We apply the proposed error detection procedure to detect errors in the test sets. For the detected lightly erroneous utterances, the error correction performances before and after applying this mechanism are included in Table 5.5. M* refers to the special mechanism. In this set of experiments, we use the proposed

candidate creation algorithm, which is imperfect. The observations under perfect candidate creation are similar.

	Number of Utterances	Number of Characters	Baseline CER %	Error Correction CER %	
				Before M*	After M*
TestSet_G	209	4,098	11.9	11.0	10.6
TestSet_N	1790	28,456	20.4	19.5	19.2

Table 5.5: CER before and after applying the mechanism (M*) to handle false alarms, evaluated on the lightly erroneous utterance subsets

Table 5.5 shows that the mechanism to handle false alarms brings substantial improvements for error correction. If we compare with the correction performance before applying this mechanism, the relative CER reductions are 3.6% and 1.5% for TestSet_G and TestSet_N respectively. These demonstrate that it is feasible to design special strategies to handle error propagation between the subsequent procedures in a multi-procedure system. We further illustrate the details of error correction on the TestSet_G and TestSet_N in Figures 5.2 and 5.3 respectively.

From these two figures, we can see that the overall performance of error correction is determined by two character sets: (1) those true errors that are corrected and (2) those false errors that are "corrected" to new misrecognitions. For the remaining characters, the error rate remains unchanged after the error correction procedure. The first character set (Set_P) represents the positive effect of error correction, while the second set (Set_N) represents the negative effect. The overall performance is the competing result of these two conflicting effects. The strategy for handling false alarms is to substantially reduce the size of Set_P and at the same time to keep the reduction of Set_N relatively small. This enlarges the size difference between the two competing sets, making the overall error correction performance improved.

Given a specific automatic error detection procedure, the sizes of Set_P and Set_N are respectively determined by R_T (i.e., the correction rate for all true errors) and R_F (i.e., the rate of uncorrected false errors). We further analyze the impact of applying the mechanism to handle false alarms on these two rates. The result is shown in Table 5.6.

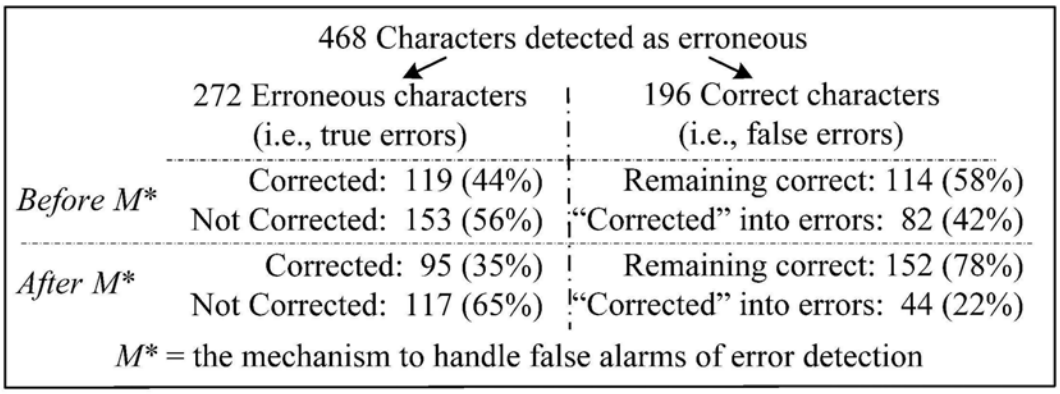

Figure 5.2: Error correction details on TestSet_G,
evaluated on the lightly erroneous utterance subset

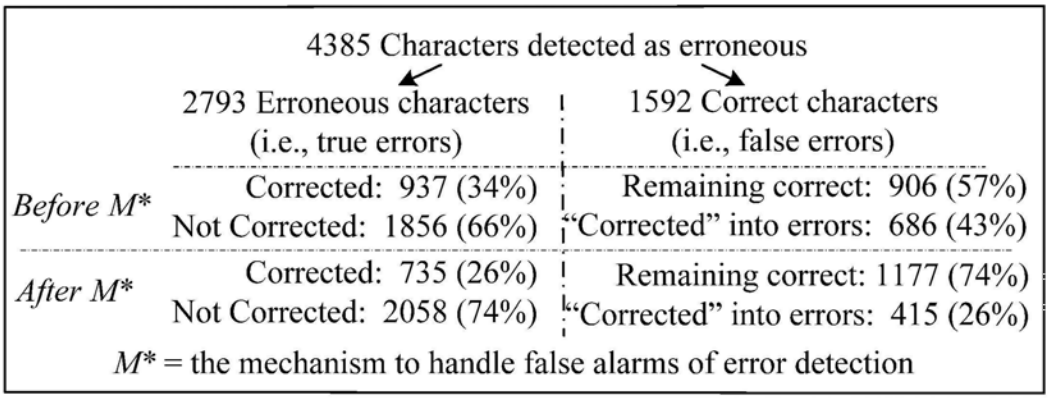

Figure 5.3: Error correction details on TestSet_N
evaluated on the lightly erroneous utterance subset

		TestSet_G		TestSet_N	
		R_T %	R_F %	R_T %	R_F %
Error Correction	without M*	43.8	58.2	33.5	56.9
	with M*	34.9	77.6	26.3	73.9

Table 5.6: The influence of the mechanism (M*) to handle false alarms on R_T and R_F, evaluated on the lightly erroneous utterance subsets. R_T refers to the correction rate for all true errors and R_F refers to the rate of uncorrected false errors.

In Table 5.6, we observed that after applying this mechanism, R_F is substantially increased. This means the ratio of turning a false error into a new misrecognition is greatly reduced. Although R_T is dropped at the same time, the reduction scale for R_T is relatively much smaller. Thus the overall CER is reduced, as illustrated in Table 5.5.

5.2 Formulation

In this subsection, we summarize the previous discussions by describing the overall effectiveness of the ED-EC framework in a formal way. We evaluate the benefit of applying the ED-EC framework in terms of the reduction in CER over the baseline recognizer performance. Based on the proposed error detection and correction procedures, the ED-EC framework only attempts to detect and correct substitution errors. Thus after applying the framework, the total character number and the number of insertions/deletions remain the same. The impact lies in the change of the number of substitutions. Some substitutions are corrected, and at the same time, some correct characters are "corrected" into new substitutions. The effectiveness of the framework can be described as the formula below:

$$C_{ED-EC} - C_{ori} = \frac{N_E * R_{NN} * R_{L\_T} * R_{T\_C} - N_C * R_{PN} * (1 - R_F)}{N_{all}} \tag{5.3}$$

In Equation 5.3, $C_{ED\text{-}EC}$ is the CER after applying the ED-EC framework, while C_{ori} is the CER of the baseline recognition. N_E and N_C are the number of substitutions and the number of correct characters, respectively in the recognized utterances. N_{all} is the total number of characters. R_{NN} and R_{PN} are two rates related to the performance of error detection. R_{NN} is the percentage of substitutions that are detected as errors. R_{PN} is the rate at which a correct character is wrongly detected as an error. R_{L_T} is the reference coverage rate of the candidate lists created for the true errors. For false errors, the reference coverage rate is 1. R_{T_C} and R_F are the two error-correction related rates described in Section 5.1.2. As discussed previously, both R_{T_C} and R_F are affected by a series of factors: the linguistic knowledge sources utilized, the number of detected errors in an utterance, the performances of error detection and candidate creation, and the special mechanism to handle false alarms. For the numerator of Equation 5.3, the first part is

the number of corrected substitutions while the second part is the number of newly introduced misrecognitions.

5.3 Performance Upper Bounds

We also estimate the performance upper bounds of the ED-EC framework, as shown in Table 5.7. Imperfect/perfect error detection and candidate creation are defined in the same way as in Section 5.1.2. The current framework uses proposed procedures (i.e., imperfect procedures) of error detection and candidate creation. Up-bound 1, Up-bound 2 and Up-bound 3 refer to the three upper bounds that are achieved by using perfect candidate creation, perfect error detection, and both, respectively (see Table 5.7). In this subsection, *CER reduction* refers to the relative CER reduction over the recognizer baseline. All CERs are evaluated on the whole test sets.

	Error Detection	Candidate Creation	TestSet_G		TestSet_N	
			CER%	Reduction%	CER%	Reduction%
Baseline	/	/	8.89	/	19.86	/
Framework	Imperfect	Imperfect	8.36	6.0	19.35	2.6
Up-bound 1	Imperfect	Perfect	8.01	9.9	18.83	5.2
Up-bound 2	Perfect	Imperfect	6.69	24.7	17.44	12.2
Up-bound 3	Perfect	Perfect	5.69	36.0	15.83	20.3

Table 5.7: Performance upper bounds for the ED-EC framework

The results show that the potential benefit of the ED-EC framework is substantial. On TestSet_G and TestSet_N, improving error detection alone can achieve maximal CER reductions of 24.7% and 12.2%, respectively, while improving candidate creation alone can obtain maximal CER reductions of 9.9% and 5.2%, respectively. The upper bounds in CER reduction for refining both candidate creation and error detection are as high as 36.0% and 20.3% for TestSet_G and TestSet_N, respectively.

Another observation is that the CER reductions on TestSet_G are consistently larger than the corresponding ones on TestSet_N. While the current framework achieves better performance on TestSet_G, the improvement space is also bigger for TestSet_G. This indicates that the ED-

EC framework is more promising for general-domain LVCSR. One reason is that the framework adopts general-domain linguistic models for error correction and thus is more effective at correcting recognition errors in the general-domain context. Another reason is that the general baseline recognizer performs better for general-domain LVCSR, leading to less misleading information (e.g., misrecognition) in the search space for error correction.

Note that in Table 5.7, only error detection and candidate creation are included. The performance upper bound of the ED-EC framework may further be enhanced in other directions, such as adopting better knowledge (linguistic and/or acoustic) model in error correction and refining the special mechanism to handle false alarms of error detection.

5.4 Computation Efficiency

For the proposed ED-EC prototype, the error detection procedure is very efficient, taking about 0.2 seconds to handle a test utterance. The efficiency of the current error correction procedure varies for utterances with different numbers of detected errors. This study applies the error correction procedure to utterances with one to four detected erroneous characters. Table 5.8 shows the average time for applying error correction. The overall average correction time for an utterance is also listed. In this study, computation times are estimated on a server with Pentium 4 CPU of 3.20 GHz. All data in Table 5.8 are presented in seconds as the time unit.

Number of detected erroneous characters	Candidate Creation	Segmentation	POS Tagging	Linguistic Scoring	Total Time Cost
One	0.18	0.074	0.057	0.0046	0.32
Two	0.36	1.5	1.1	0.091	3.1
Three	0.54	28.5	22.6	1.8	53.5
Four	0.72	589.6	452.8	36.5	1079.6
Overall	0.45	135.1	103.8	8.4	247.7

Table 5.8: The average time to apply the error correction procedure on an utterance, evaluated on the lightly erroneous utterance subsets. The time unit is second.

Table 5.8 shows that the processing time for an utterance grows dramatically when the number of contained errors increases. For an utterance with one or two errors, the error correction procedure is efficient. Processing a single-error utterance uses less than 0.5 second, and processing a two-error utterance normally takes several seconds. For those utterances with three errors, nearly one minute is needed to handle an utterance. For an utterance with four errors, the average computation time is 18 minutes. These observations suggest that with the proposed algorithms, the ED-EC framework may adjust the application scope of the error correction procedure to meet the requirement of system efficiency. For real-time recognition, processing utterances with four detected errors is obviously inappropriate.

The results in Table 5.8 also show that the current search approach of enumerating all utterance hypotheses in a search space has a direct impact on computational load. When the error number increases in a target utterance, the number of hypotheses in the search space grows exponentially from 20 to 160,000. As a consequence, the computational load on segmentation, POS tagging and linguistic scoring also increases dramatically, occupying the majority of computation for utterances with more than one error. These indicate that designing a more efficient algorithm to identify the top-scoring path in search space should be beneficial. With an efficient search method, application of the error correction procedure to utterances with more than four errors would be feasible. Efficient search algorithms such as A* search can usually work directly over search networks. The segmentation problem is the major obstacle for the design of such network-based search algorithms. Sacrificing some accuracy may be necessary to achieve efficiency. We will further the investigation in this direction in the future.

The efficiency of error correction may also be improved in some other ways. First, it is possible to reduce or even eliminate segmentation cost. The current linguistic scoring algorithm requires each utterance hypothesis to be segmented into two word sequences using two lexicons (i.e., a lexicon used for the mutual information and word trigram models and a lexicon used for POS tagging). If all lexicons used are identical, the segmentation cost will be substantially reduced. If we chose a word instead of a character as the smallest unit for error detection and correction, it would be unnecessary to perform segmentation, and the related cost will be eliminated. Second, better efficiency can also be achieved by abandoning the usage of some knowledge sources. For example, if we exclude the POS trigram model, whose contribution for error correction is relatively small, the requirement for POS tagging will disappear, and the cost

for segmentation and linguistic scoring will decrease accordingly. Whether or not to adopt a knowledge source is a decision that must take into account two conflicting factors, the improved accuracy and the decreased efficiency. Other possible approaches to enhance efficiency include optimizing detailed implementations, such as the tagging method.

5.5 Chapter Summary

This chapter analyzes the effectiveness of the ED-EC framework. We first demonstrate that the error correction effectiveness of the framework is determined by multiple factors. The effectiveness of error correction varies from one linguistic knowledge source to another. Properties of the sausage-type search spaces created for error correction, such as the number of utterance hypotheses and the amount of misleading information, also influence the effect of error correction. The special mechanism designed to handle false alarms of error detection is shown to be effective. We then evaluate the overall framework performance in terms of CER reduction, performance upper bounds and computational efficiency. We describe the ability of the ED-EC framework to reduce recognition error rate by a single equation (i.e., Equation 5.3). Experiments to estimate the performance upper bounds show that the potential benefit of the ED-EC framework is big. Concerning the computational efficiency of the framework, while the error detection procedure is efficient, the computational load of the error correction procedure to process an utterance depends heavily on the number of detected erroneous characters.

In the next chapter, we will further the analysis by comparing the ED-EC framework with other methods that incorporate advanced linguistic knowledge in LVCSR.

Chapter 6

Competitive Analyses

This chapter compares the error detection and correction (ED-EC) framework with other approaches that use advanced linguistic knowledge sources to benefit large vocabulary continuous speech recognition (LVCSR). Methods to incorporate additional linguistic knowledge into LVCSR can be categorized into two classes: single-pass strategies and multi-pass strategies. Algorithms of single-pass strategies incorporate advanced linguistic models into a single-pass decoding procedure. Implementation of multi-pass strategies applies sophisticated knowledge models to post-process the recognizer output. The ED-EC framework adopts a multi-pass strategy. In Section 6.1, we compare the properties of single-pass strategies and multi-pass strategies. These properties include the application of knowledge sources and the usage of context information.

Section 6.2 describes the specifics of multi-pass strategies. Previous post-processing approaches applied linguistic models to re-rank the *N*-best hypotheses, to rescore recognition lattices, or to process other post-processing search spaces, as discussed in Section 2.3. These previous efforts processed all signals in an indiscriminate way. In contrast, the ED-EC framework attempts to selectively apply computationally expensive models only to the necessary parts of the signal. Section 6.2 compares these two different ways of processing. *N*-best re-ranking is selected as a representative of post-processing methods that treat all signals equally. A similar advanced linguistic model is used to implement the ED-EC framework and *N*-best re-ranking. The performance of the ED-EC framework and *N*-best re-ranking are compared and analyzed.

Finally, we analyze the computational expenses of different strategies in Section 6.3. Single-pass strategies and most post-processing methods evenly distribute the computational load for all signals. We compare the ED-EC framework with such methods in four aspects: computation saved for correct utterances, computation saved for correct characters in erroneous utterances, computation expended for erroneous characters and computation expended on error detection.

6.1 Multi-Pass Strategy vs. Single-Pass Strategy

Single-pass strategies are theoretically more advantageous than multi-pass strategies, because single-pass strategies apply advanced linguistic knowledge sources in the decoding search space. Since this space contains all possible hypotheses, the optimal benefit of introducing certain linguistic knowledge sources is theoretically achievable. In practice, various factors (e.g., pruning that has to be performed to keep computational load and memory consumption under control) will make the performance suboptimal.

Multi-pass strategies use advanced knowledge sources to distinguish among hypotheses in post-processing search spaces, such as *N*-best hypotheses and recognition lattices. Since a post-processing search space is a subset of the decoding (full) search space and may not include the reference utterance, the performance of multi-pass strategies can be suboptimal. Given certain advanced knowledge models, the performance of a post-processing approach depends on how well the search spaces cover reference utterances. For the ED-EC framework, when the performance of error detection and candidate creation procedures improve, the rate of inclusion of reference utterances in the sausage-type search spaces will increase. As a result, the gain of the knowledge sources approaches optimal.

Although single-pass strategies may provide better performance, there are two practical issues making single-pass strategies unfavorable. First, it is difficult to apply advanced linguistic knowledge sources in the decoding procedure. Since high-level linguistic models (e.g., the inter-word mutual information model) require long-distance dependency, incorporating these models into the decoder will greatly increase the decoding complexity, leading to expensive computation and high memory consumption. Second, context information is not fully utilized in single-pass

strategies. In conventional left-to-right decoding, only the left context is available. This may reduce the effectiveness of the advanced knowledge. In the rest of this subsection, we address these two issues in detail, in Section 6.1.1 and Section 6.1.2, respectively.

6.1.1 Difficulty in Incorporating Knowledge

Incorporating advanced linguistic knowledge sources into single-pass decoding is challenging. Using a pre-compiled static recognition network to store linguistic models that capture long-distance dependencies leads to a major difficulty in network design, as discussed in Section 1.2. Even if the recognition network can be successfully designed, the size of the network would be prohibitive since all partial paths covering the required dependencies have to be included in the network. Besides, for complicated linguistic knowledge sources, such as syntactic knowledge, redesigning a recognizer to capture structural information may be necessary [116].

An alternative approach to applying additional linguistic models in single-pass decoding is to dynamically interpolate linguistic scores during decoding. This method keeps the original static recognition network in the baseline recognizer and adopts an additional scheme to use advanced linguistic models to score hypotheses. Searching is then performed by interpolating the scores from the original recognition network and the scores from the additional scheme of active hypotheses in the decoding search space. This approach circumvents the difficulty of designing a static recognition network. However, heavy computation has to be spent on exchanging information between the additional scheme and the recognition network constantly throughout the decoding procedure. This may result in serious system inefficiency.

Compared with incorporating sophisticated linguistic knowledge sources into single-pass decoding, applying advanced models in a post-processing procedure is much more convenient, as a post-processing search space normally contains only a limited number of hypotheses. For the example of the inter-word mutual information (MI) model, which evaluates the average MI of a target word with all other words in the utterance context (see Section 3.4.2) the ED-EC framework applies this model to score hypotheses in sausage-type search spaces without difficulty. In contrast, incorporating this MI model into single-pass decoding would be very difficult because the calculation of MI scores depends on the whole left and right utterance context.

6.1.2 Usage of Context

State-of-the-art recognizers decode speech in a left-to-right style. When processing a specific speech segment, only the left context is available for computing the acoustic and linguistic likelihoods. Although the right context of the target speech segment will be taken into account when the recognizer processes the subsequent segments of the signal, the correct path may have already been wrongly pruned and thus be unrecoverable. One advantage of the multi-pass strategy is that both the left and right context are available during post-processing. For the ED-EC framework, if the sausage-type search spaces always contain reference utterances, the framework will outperform the single-pass strategy, since the framework can set better linguistic constraints by considering the right context. In other words, when the error detection and candidate creation procedures approach perfection, this framework will eventually outperform the single-pass strategies.

We evaluate the benefit of utilizing the right context for the example of the ED-EC framework. In this framework, the scoring of MI, word trigrams and POS trigrams are based on both left and right context as reflected in Equation 3.11, 3.13 and 3.14 respectively. This set of experiments modifies the scoring by limiting the calculations to only one side of the context, as illustrated in Equation 6.1 to 6.6. For simplicity, we assume error detection is perfect and apply the error correction procedure on each test utterance that contains a single substitution. In the equations below, w_k refers to the word containing the single substitution in a target utterance w_1 $w_2 \ldots w_m$. The correction rates for the substitutions in focus are shown in Table 6.1.

$$K_{MI\_left}(w_1, w_2, \ldots . w_m) = \frac{\sum_{i=1,..k-1} MI(w_i, w_k)}{k-1} \tag{6.1}$$

$$K_{MI\_right}(w_1, w_2, \ldots . w_m) = \frac{\sum_{i=k+1,..m} MI(w_i, w_k)}{m-k} \tag{6.2}$$

$$K_{WdTri\_left}(w_1, w_2, \ldots w_m) = K_{WdTri}(w_1, w_2, \ldots w_k) \tag{6.3}$$

$$K_{WdTri\_right}(w_1, w_2, \ldots w_m) = K_{WdTri}(w_k, w_{k+1}, \ldots w_m) \tag{6.4}$$

$$K_{PosTri\_left}(w_1, w_2, \ldots w_m) = K_{PosTri}(w_1, w_2, \ldots w_k) \tag{6.5}$$

$$K_{PosTri\_right}(w_1, w_2, ...w_m) = K_{PosTri}(w_k, w_{k+1}, ...w_m) \quad (6.6)$$

	Correction rate % under various context usage					
	TestSet_G			TestSet_N		
	Both	Left	Right	Both	Left	Right
MI	49.5	48.4	45.2	49.2	46.0	48.0
Word Trigram	55.9	40.9	44.1	42.7	33.3	34.5
POS Trigram	21.5	14.0	12.9	25.0	22.6	22.4

Table 6.1: Error correction based on various context usages. Corrections rates are evaluated on the utterances that contain a single substitution.

From Table 6.1, we can see that (1) the right context is almost equally as effective as the left context and (2) utilizing both contexts increases the correction rate for all the three linguistic knowledge sources. These observations indicate that the two sides of the context provide complementary information and are beneficial for improving LVCSR performance. If a post-processing search space is a network (e.g., the sausage-type search space in the ED-EC framework), methods that search within the network based on both contexts should be an interesting research topic. The current ED-EC prototype simply enumerates all hypotheses in a network. Developing more efficient searching algorithms is also possible.

6.2 ED-EC Framework vs. *N*-Best Re-Ranking

There are various post-processing techniques such as *N*-best re-ranking and lattice rescoring. In contrast with the ED-EC framework, which selectively applies sophisticated linguistic knowledge sources to parts of the signal, existing post-processing techniques normally process all signals equally. This subsection selects *N*-best re-ranking as a representative of equal-processing methods and compares it with the ED-EC framework. Given a knowledge model, while the ED-EC framework ranks utterance hypotheses in the created sausage-type search spaces, *N*-best re-ranking ranks *N*-best hypotheses generated during baseline decoding.

6.2.1 *N*-Best Re-Ranking

The *N*-best re-ranking approach uses a knowledge model to score and re-rank *N*-best utterance hypotheses generated by the baseline recognizer. Recognizer scores (i.e., scores assigned to utterance hypotheses by the baseline recognizer) are also included in hypothesis scoring to enhance robustness. We define the scoring method of the *N*-best re-ranking algorithm as linear interpolation of recognizer scores and knowledge model based scores as:

$$S(h) = \mu \cdot D(h) + (1-\mu) \cdot K(h) \tag{6.7}$$

where *h* refers to an utterance hypothesis; *D(h)* and *K(h)* refer to the recognizer score and the knowledge-model based score assigned to the hypothesis, respectively; and μ is the interpolation weight. *D(h)* is computed as follows:

$$D(h) = \sum_{i=1}^{n} (\alpha_0 \cdot P_{AM}(w_i) + \beta_0 \cdot P_{LM}(w_i)) - n \cdot r \tag{6.8}$$

where $w_1 w_2 \ldots w_n$ is the word sequence corresponding to *h*; $P_{AM}(w_i)$ and P_{LM} *Score*(w_i) are the acoustic likelihood and language model (LM) likelihood in the log domain for the word w_i, respectively; α_0 and β_0 are the acoustic and LM weights adopted by the recognizer, respectively; and *r* refers to the word insertion penalty.

To fairly compare the behavior of the ED-EC framework and the *N*-best re-ranking, we use the same knowledge model for both techniques. In the ED-EC framework, we adopt an advanced linguistic model that linearly combines MI, word trigrams and POS trigrams, as shown in Equation 3.15. To facilitate reading, we rewrite this equation as follows:

$$K(h) = r_1 \cdot K_{MI}(h) + r_2 \cdot K_{WdTri}(h) + r_3 \cdot K_{PosTri}(h) \tag{6.9}$$

where $K_{MI}(h)$, $K_{WdTri}(h)$ and $K_{PosTri}(h)$ refer to the scores computed based on MI, word trigrams and POS trigrams, respectively; and r_1, r_2 and r_3 are the interpolation weights. For *N*-best re-ranking, we also use this advanced model to score corresponding utterance hypotheses, or in other words, to compute *K(h)* in Equation 6.7. In this case, all words in *h* are viewed as *target words* for the computation of $K_{MI}(h)$. In addition, since the ED-EC framework attempts to only modify those lightly erroneous utterances (i.e., utterances with one to four detected character errors), we focus comparison experiments on the lightly erroneous utterance sets only.

With the definitions and settings above, we conduct a set of experiments on the lightly erroneous utterances in the development set ErrCorr_Set to identify the optimal number of *N*-best hypotheses along with the optimal interpolation weight μ for *N*-best re-ranking. To facilitate the discussion, we refer to the maximum number of *N*-best hypotheses allowed for an utterance as *M*. Given *M*, the performance of the *N*-best re-ranking (i.e., the CER of the top-ranking hypotheses) depends on the value of μ. For each *M*, we identify the optimal μ by grid search and record the corresponding optimal performance of *N*-best re-ranking in Figure 6.1. We notice that when *M* increases to 60, the CER drops to its minimum point. After that, including a greater number of the *N*-best hypotheses slowly decreases the re-ranking performance. Based on this observation, we adopt the 60-best hypotheses along the corresponding optimal μ in the task of *N*-best re-ranking.

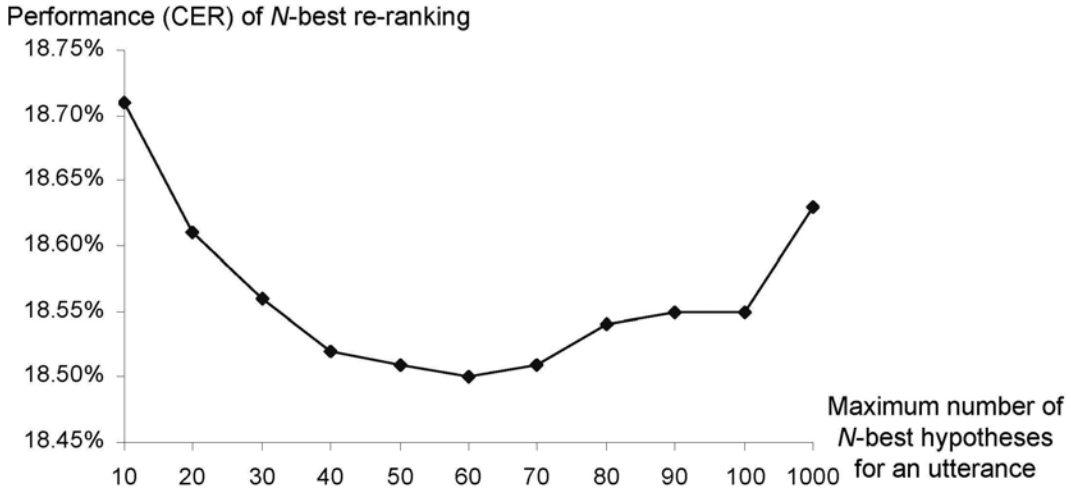

Figure 6.1: The change of the performance of *N*-best re-ranking when the maximum number of *N*-best hypotheses that is allowed for an utterance increases. Performances are evaluated on the lightly erroneous utterance subset in the development set ErrCorr_Set.

6.2.2 Comparison

We evaluate the *N*-best re-ranking on the lightly erroneous utterance sets of the two test sets based on the 60-best hypotheses. The experimental results are included in Table 6.2. In the table,

"With Recognizer Score" refers to the *N*-best re-ranking that considers recognizer scores as specified by Equation 6.7; "Knowledge Only" refers to the case in which an hypothesis is scored/ranked by *knowledge(hypo)* only for *N*-best re-ranking; and "With Perfect Candidate Creation" refers to the ED-EC framework using a perfect candidate creation procedure with 100% reference coverage rate. This perfect procedure of candidate creation is implemented by manually replacing the lowest-ranking candidate with the reference if the reference is absent in the created candidate list.

		CER %	
		TestSet_G	TestSet_N
Baseline		11.9	20.4
N-best Re-ranking	**With Recognizer Score**	**10.2**	**18.9**
	Knowledge Only	11.9	20.1
ED-EC framework	**Proposed Framework**	**10.6**	**19.2**
	With Perfect Candidate Creation	9.8	18.1

Table 6.2: Comparison of the *N*-best re-ranking and the ED-EC framework on the lightly erroneous utterance subsets.

Table 6.2 shows that the *N*-best re-ranking considering recognizer scores outperforms the current ED-EC prototype. However, given a specific knowledge model, room for improvement in *N*-best re-ranking is very limited, but in the ED-EC framework, it is big, as discussed in Section 5.3. The ED-EC framework is expected to outperform the *N*-best re-ranking quickly when error detection and/or candidate creation performance improves. As shown in Table 6.2, the ED-EC framework performs better than the *N*-best re-ranking on both test sets if candidate creation is perfect (i.e., if candidate lists always include reference characters). For the current candidate creation algorithm, the reference coverage rates (R_L) of candidate lists for true errors are 81.6% and 73.3% for TestSet_G and TestSet_N respectively. Assuming that CER decreases linearly when R_L increases, the ED-EC framework will outperform the *N*-best re-ranking when the R_L for true errors rises above 90.8% and 80.6% respectively. Note that the benefit of improving error detection is much larger than that of enhancing lattice creation, as discussed in Section 5.3. The ED-EC framework is potentially more promising than the *N*-best re-ranking.

Another interesting observation in Table 6.2 is that incorporating recognizer scores is critical for effective *N*-best re-ranking. If only the advanced linguistic model is used to score/rank hypotheses, as in the ED-EC framework, *N*-best re-ranking performs worse than the ED-EC framework. This result indicates that the sausage-type search space of the ED-EC framework is more advantageous than the search space of *N*-best hypotheses. It also illustrates the usefulness of acoustic likelihood and LM likelihood assigned by the baseline trigram model. Including the two likelihoods to rescore hypotheses in the ED-EC framework may be beneficial too. However, additional computational overhead is incurred to calculate the two likelihoods for the ED-EC framework, whereas the *N*-best re-ranking can directly obtain these two likelihoods from the recognizer output.

6.3 Differences in Computational Expense

Previous efforts to incorporate advanced linguistic knowledge in LVCSR applied related models to process all signals in an indiscriminate way, by using either the single-pass decoding strategy or a post-processing technique such as *N*-best re-ranking. The difference of the ED-EC framework is that it attempts to apply sophisticated models only to the necessary parts of the signal. This section discusses four aspects of how computation is saved and expended in the ED-EC framework.

- **Computation saved for correct utterances**

State-of-the-art recognizers can already achieve high recognition accuracy for dictation speech. The baseline recognizer correctly recognizes 38% of the utterances and 24% of the utterances for TestSet_G and TestSet_N respectively. For these correctly recognized utterances, applying additional knowledge models in either decoding or post-processing wastes computation and may even decrease recognition accuracy. The ED-EC framework attempts to use an error detection procedure to filter out these correct utterances from the processing of sophisticated linguistic models.

With the proposed error detection procedure, more than 80% of the correct utterances are successfully detected as correct for both test sets. For these truly correct utterances, unnecessary

computation is saved and possible accuracy degradation is avoided. This is the positive effect of filtering out detected correct utterances from further processing. However at the same time, the negative effect is that a number of erroneous utterances are wrongly labeled as correct, as shown in Table 3.4, and not "correcting" these false correct utterances leads to a performance loss. The influence of filtering out those utterances labeled as correct thus becomes the competing result of the positive and negative effects. We analyze this phenomenon with the example of *N*-best re-ranking. We apply the *N*-best re-ranking algorithm described in Section 6.2.1 to the correct utterances detected in the two test sets. The interpolation weight μ is re-tuned on all the utterances in the development set ErrCorr_Set. The results are illustrated in the figure below.

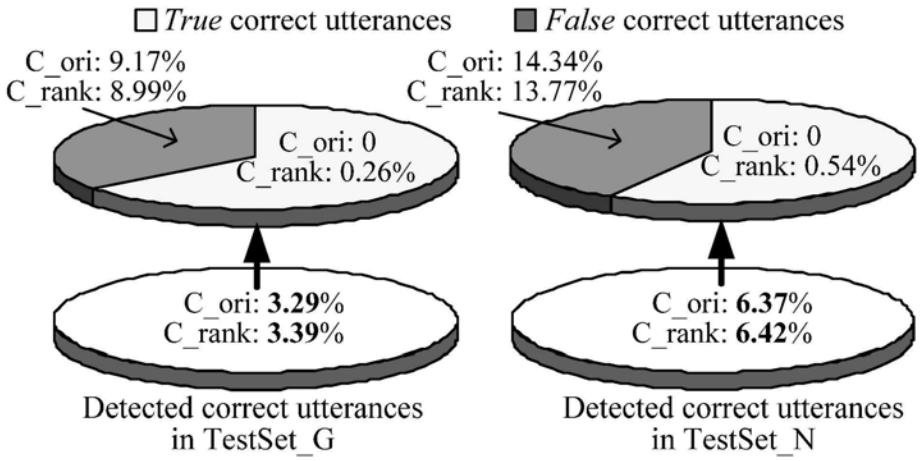

Figure 6.2: Performance of the *N*-Best re-ranking on detected correct utterances. C_ori refers to the baseline CER; C_rank refers to the CER of the re-ranking.

Figure 6.2 shows that although the application of *N*-best re-ranking reduces the CERs on false correct utterances, it introduces errors into the true correct utterances, and the overall performance with regard to the utterances labeled as correct is hurt. This observation supports the decision of filtering detected correct utterances from further processing. Note that the error detection procedure labels about half of the utterances as correct for TestSet_G and more than a third of the utterances as correct for TestSet_N. The reduction in computation load is substantial.

- **Computation saved for correct characters in erroneous utterances**

Given a recognized utterance, the EC_EC framework attempts to create alternatives only for misrecognitions in the sausage-type search space. The application of advanced linguistic

knowledge is then focused on distinguishing among those alternatives. Compared with the efforts that equally process all signals with sophisticated LMs in decoding or post-processing, the ED-EC framework is computationally economical in the sense that it is unnecessary to distinguish among alternatives for detected correct regions in signals.

With an automatic error detection procedure, some recognition errors are missed, and alternatives may be created for correctly recognized characters. With the proposed error detection algorithm, 40.0% and 49.5% of the character errors are missed for TestSet_G and TestSet_N, respectively. Not applying advanced linguistic knowledge to these undetected errors may lead to performance loss. However, 94.5% and 93.0% of the correct characters are successfully detected for TestSet_G and TestSet_N, respectively. The LM computation saved on these correct characters is substantial. This may compensate for the possible loss related to the undetected misrecognitions.

- **Computation expended for erroneous characters**

The ED-EC framework applies advanced linguistic models to errors. Given a recognition error, in most search spaces, such as recognition lattices and *N*-best lists, the number of alternatives for this error is limited, and these alternatives may not include the corresponding reference. Enlarging the alternative lists for errors to enhance the coverage rate of references should be beneficial. The ED-EC framework attempts to fulfill this objective by introducing a separate candidate creation scheme. As a result, the computation on the detected erroneous regions in signals for the ED-EC framework may be heavier than that for other post-processing techniques, since it is more focused.

The ED-EC framework places more computational emphasis on those utterances containing multiple errors. When the number of errors in an utterance increases, the size of the sausage-type search space grows, making the subsequent processing slower. We believe that this is reasonable since the worse the recognizer performs, the larger the effort needed to compensate. For a state-of-the-art recognizer, heavily erroneous utterances that require intensive computation normally only occupy a small portion of recognized utterances.

With the proposed error detection algorithm, about 40% of the detected errors are false errors. While the computation expended on true errors is worthwhile, the computation on false errors is wasted and can harm performance.

- **Additional computation expended on error detection**

As shown in Section 5.4, compared with the heavy computation load of error correction, the cost of error detection is small. Adopting a better error detection algorithm may lead to heavier computation for this procedure. However at the same time, the efficiency of other aspects will be improved. For example, unnecessary computation of false errors will be reduced. Thus, the efficiency of error detection should be evaluated based on overall system behavior.

6.4 Chapter Summary

This chapter presents a competitive study. We first compare two types of strategies, namely single-pass strategies and multi-pass strategies, for utilizing advanced linguistic knowledge to benefit LVCSR. Single-pass strategies incorporate advanced models in single-pass decoding, whereas multi-pass strategies apply these models in post-processing. The ED-EC framework is based on a multi-pass strategy. Compared with single-pass strategies, multi-pass strategies have two main advantages: (1) the simplicity to apply a knowledge model and (2) the ability to utilize the right context. We then compare the ED-EC framework with a widely used post-processing technique named *N*-best re-ranking. Both performance and potential benefits are discussed. Finally, we discuss the differences in computational expense between the ED-EC framework and the approaches that equally process entire signals. The ED-EC framework saves computation on detected correct regions in signals while expending additional computation for error detection.

In the rest of this thesis, we will investigate the effectiveness of the ED-EC framework across various baseline LVCSR systems.

Chapter 7

An Approach to Enhancing the Baseline Recognition System

Previous chapters have outlined the development and analysis of an error detection and correction (ED-EC) framework with a state-of-the-art baseline recognizer. The remaining questions are: (1) Will the ED-EC framework be effective for other baseline recognition systems with different performance? (2) Can the baseline recognition system be a multi-pass system that already uses certain techniques to post-process the output of a recognizer? Note that the baseline recognition system for an ED-EC framework does not have to be a single-pass recognizer. To answer the two questions, in this chapter we propose a novel post-processing approach, called Discriminative Lattice Rescoring (DLR), to enhance the recognizer baseline system. The analyses regarding the two questions above can thus be performed based on the discriminatively enhanced baseline systems in the next chapter.

The DLR algorithm is an extension of discriminative n-gram modeling. Discriminative n-gram modeling has been shown to be effective in improving LVCSR when the training and testing conditions are similar in nature [147]. The novel aspects of the DLR algorithm lie in (1) recasting the discriminative n-gram model as a pseudo-conventional n-gram model and (2) using this pseudo model to rescore recognition lattices generated by decoding. In this chapter, Section 7.1 provides an overview of the DLR technique. The motivation and advantages of this technique are addressed. Section 7.2 briefly reviews discriminative n-gram modeling. Section 7.3 discusses the pseudo-conventional n-gram representation of a discriminative n-gram model.

Section 7.4 presents the approach of using the pseudo-conventional n-gram model to rescore recognition lattices. Finally, the experiments and analyses are given in Section 7.5.

7.1 Overview

Recently there has been a growing interest in adopting discriminative training methods to enhance LVCSR performance [8, 76, 137], as discussed in Section 1.3. Among these efforts, one effective method is discriminative n-gram modeling, which selects the n-gram counts along with the recognition scores as features and defines a global linear model to distinguish among the utterance hypotheses in *N*-best lists or recognition lattices. Encouraging results have been reported for using discriminative n-gram modeling to improve English LVCSR [105, 106]. Our own previous effort [147] showed that this technology can also effectively reduce the error rate for Mandarin LVCSR, especially when the training and testing conditions are of similar nature. In addition, discriminative n-gram modeling is efficient in training, compared with those discriminative training methods that require iterative decoding [20, 71].

In this work, we want to use discriminative n-gram modeling to raise the recognizer baseline and apply the ED-EC framework on the raised baseline. However, we notice that it is not straightforward to extend discriminative n-gram modeling with other techniques. Previous works have used the trained discriminative n-gram model to score and rank *N*-best hypotheses. Some works have used a weighted finite-state automaton (WFA) to store the discriminative n-gram model. By viewing a recognition lattice as an acyclic WFA, they then identified the discriminatively top-scoring paths in recognition lattices with a series of WFA operations, including the intersection of two WFAs [106]. In both cases, only the top-scoring hypothesis is generated for each speech utterance. This makes the further application of other techniques difficult, since most post-processing techniques require more than the best scoring hypothesis, e.g., a recognition lattice or the *N*-best hypotheses.

To facilitate the extension of discriminative n-gram modeling, this thesis demonstrates that the linear discriminative n-gram model can be recast as a pseudo-conventional n-gram model if the order of the discriminative n-gram model is lower than or equal to the order of the n-gram model in the baseline recognizer. Using this pseudo-conventional n-gram model, the

power of the discriminative n-gram model can be easily captured in single-pass n-gram decoding or lattice rescoring. Since both single-pass decoding and lattice rescoring can generate lattices, other post-processing techniques can be applied on these discriminatively generated lattices in the same way as the conventional recognition lattices. Extending discriminative n-gram modeling with other techniques to achieve cumulative improvement thus becomes convenient.

In this work, we use the pseudo-conventional n-gram model to rescore recognition lattices using an efficient algorithm, which computes the pseudo-conventional n-gram likelihoods online. Within the discriminatively rescored lattices, the best hypothesis (i.e., the utterance hypothesis scored highest by the discriminative n-gram model) can be efficiently identified by A* search. These discriminatively rescored lattices can also be delivered to provide information for further processing of the ED-EC framework, as will be discussed in the next chapter.

7.2 Discriminative N-Gram Modeling

Discriminative n-gram modeling defines a linear framework that re-ranks utterance hypotheses generated by a baseline recognizer. These utterance hypotheses can either be the *N*-best hypotheses or the ones contained in recognition lattices [105, 106]. Discriminative n-gram modeling can be described as follows:

- We need a training data set with N speech utterances and n_i ($i=1 \ldots N$) hypotheses for each utterance. Define $x_{i,j}$ as the j-th ($j=1 \ldots n_i$) hypothesis of the i-th utterance. Define $x_{i,R}$ as the hypothesis with lowest CER among $\{ x_{i,j} \}$.
- We need a separate test set of $y_{i,j}$ with similar definitions as the training set.
- Define $D+1$ features $f_d(h)$, $d=0 \ldots D$, where h is a recognition hypothesis. The features could be arbitrary functions that map h to real values.
- Define a discriminant function as:

$$g(h, \vec{a}) = \sum_{i=0}^{D} a_i f_i(h) = \vec{a} \cdot \vec{f}(h) \tag{7.1}$$

The task of discriminative training involves a search for a weight vector $\vec{a}$ that satisfies the following conditions on the test set:

$$g(y_{i,R}, \vec{a}) > g(y_{i,j}, \vec{a}) \quad \forall i \forall j \neq R \tag{7.2}$$

For discriminative n-gram modeling, the features are the recognition scores and the n-gram counts. For each utterance hypothesis *h*, the base feature $f_0(h)$ is the recognizer score of *h*. The recognizer score for an utterance hypothesis is the weighted summation of acoustic and linguistic likelihoods that are assigned by the baseline recognizer. The recognizer score of *h* is computed as Equation 6.8. Since the calculation of $f_0(h)$ includes acoustic likelihoods, the discriminative n-gram model is acoustically relevant.

The remaining features are the counts of each n-gram (i.e., an n-word sequence) in *h*. We first assign each selected n-gram with a unique index *i* ($1 \leq i \leq D$). $f_i(h)$ is then defined as the count of the i^{th} n-gram in *h*. For instance, the unigram "*new*" and the bigram "*new solutions*" are assigned with indexes *j* and *k* respectively. Given that *h* is "*There are new ideas and new solutions*", $f_j(h)=2$ and $f_k(h)=1$. Normally, a discriminative *N*-gram model considers all n-grams with order $n \leq N$. For example, a discriminative bigram model usually utilizes both unigrams and bigrams.

The weight vector $\vec{a}$ can be trained by various discriminative training methods (e.g., perceptron and boosting) to minimize the training error [147]. This study adopts the perceptron algorithm. This algorithm optimizes a minimum square error (MSE) loss function [90] to approximate the minimum training error. The MSE loss function can be written as:

$$f_{loss}(\vec{a}) = \frac{1}{2} \sum_{i=1...n} (g(x_{i,R}, \vec{a}) - g(x_{i,k}, \vec{a}))^2 \tag{7.3}$$

where $x_{i,k}$ is the utterance hypothesis having the highest $g(h, \vec{a})$ value among all the candidate hypotheses for the i^{th} speech utterance.

In this study, we follow [39] to use the averaged perceptron algorithm [23, 32] to train the weights. This method first uses the standard perceptron with delta rule to iteratively update the $\vec{a}$, as shown in Figure 7.1. The weights are then averaged to increase model robustness. Define $a_d^{i,j}$ as the value of a_d after processing the j^{th} utterance in the i^{th} iteration. The average weights are calculated using Equation 7.4:

$$(a_d)_{avg} = (\sum_{i=1}^{t} \sum_{j=1}^{n} a_d^{i,j}) / (t \cdot n), \qquad d = 0...D \tag{7.4}$$

1 Initialize the weight vector
2 For $j = 1 \ldots t$ (t is the total number of iterations)
3 For the i^{th} speech utterance, $i = 1 \ldots n$
4 Choose the $x_{i,k}$ with the highest $g(h, \vec{a})$ value among all the $x_{i,j}$s
5 For $d = 0 \ldots D$ (η is the size of the learning step)
6 $a_d = a_d + \eta(g(x_{i,R}, \vec{a}) - g(x_{i,k}, \vec{a}))(f_d(x_{i,R}) - f_d(x_{i,k}))$

Figure 7.1: The standard perceptron algorithm with delta rule

7.3 The Pseudo-Conventional N-Gram Representation

In this section, we first prove that the linear discriminative n-gram model can be recast as a pseudo-conventional n-gram model in Section 7.3.1. The deduction of equations is given. Then we discuss the computation of this pseudo-convention n-gram model in Section 7.3.2. Two computation methods, building a complete pseudo model offline and generating pseudo-conventional n-gram probabilities online, are presented. Some of the material presented in this section was previously published in [148].

7.3.1 Theory

For each speech utterance, the discriminative n-gram model scores each hypothesis as shown in Equation 7.1 and selects the top-ranking hypothesis as the new recognition result. If a_0 is larger than zero, we can rewrite the scoring method as equation 7.5 without changing the ranking of the candidate hypotheses. Note that given a reasonably good baseline recognizer, the base feature $f_0(h)$ (i.e., the recognizer score) is a reliable source of information to distinguish among competing hypotheses, and thus a_0 is always positive.

$$g'(h, \vec{a}) = f_0(h) + \sum_{i=1}^{D} \frac{a_i}{a_0} f_i(h) \tag{7.5}$$

The first part $f_0(h)$ is the score that the baseline recognizer assigns to h, as follows:

$$f_0(h) = \alpha \sum_{i=1}^{m} P_{AM}(w_i) + \beta \sum_{i=1}^{m} P_{LM}(w_i \mid w_1, w_2, ...w_{i-1}) - m \cdot r \tag{7.6}$$

where $w_1 w_2 ... w_m$ is the corresponding word sequence of the utterance hypothesis h; $P_{AM}(w_i)$ and $P_{LM}(w_i|w_1, w_2, ..., w_{i-1})$ are the acoustic likelihood and language model (LM) likelihood in the log domain for the word w_i; α and β are the acoustic and LM weights adopted by the recognizer; and r refers to the word insertion penalty.

For a discriminative N-gram model that considers all n-grams with order $n \le N$, the second part of Equation 7.5 can be written as:

$$\begin{aligned} \sum_{i=1}^{D} \frac{a_i}{a_0} f_i(h) = \frac{1}{a_0} (& a_{w_1} + a_{w_2} + ... + a_{w_m} + a_{w_1 w_2} + a_{w_2 w_3} + ... \\ & + a_{w_{m-1} w_m} + ... + a_{w_1 w_2 .. w_N} + a_{w_2 w_3 .. w_{N+1}} ... + a_{w_{m-N+1} w_{m-N+2} .. w_m}) \end{aligned} \tag{7.7}$$

where $a_{w_i w_{i+1} ... w_{i+k}}$ is the weight of the n-gram ($w_i w_{i+1} ... w_{i+k}$).

Combining Equations 7.6 and 7.7, Equation 7.5 can be rewritten as:

$$g'(h) = f_0(h) + \sum_{i=1}^{D} \frac{a_i}{a_0} f_i(h)$$

$$\begin{aligned} & = \alpha \sum_{i=1}^{m} P_{AM}(w_i) + \beta \sum_{i=1}^{m} P_{LM}(w_i \mid w_1, w_2, ...w_{i-1}) - m \cdot r \\ & + \frac{1}{a_0} (a_{w_1} + a_{w_2} + ... + a_{w_m} + a_{w_1 w_2} + a_{w_2 w_3} + ... \\ & + a_{w_{m-1} w_m} + ... + a_{w_1 w_2 .. w_N} + a_{w_2 w_3 .. w_{N+1}} ... + a_{w_{m-N+1} w_{m-N+2} .. w_m}) \end{aligned} \tag{7.8}$$

$$= \alpha \sum_{i=1}^{m} P_{AM}(w_i) + \beta \sum_{i=1}^{m} P_{LM}'(w_i \mid w_1, w_2, ...w_{i-1}) - m \cdot r$$

where

$$\begin{aligned} P_{LM}'(w_i \mid w_1, .., w_{i-1}) = & P_{LM}(w_i \mid w_1, .., w_{i-1}) + \\ & \frac{1}{a_0 \cdot \beta} (a_{w_i} + a_{w_{i-1} w_i} + ... + a_{w_{i-N+1} w_{i-N+2} ... w_i}) \end{aligned} \tag{7.9}$$

Equations 7.8 and 7.9 indicate that scoring an utterance hypothesis by the discriminative *N*-gram model is equivalent to scoring the hypothesis by an updated recognizer. The updated recognizer replaces the original language model $P_{LM}(w_i|w_1,\ldots,w_{i-1})$ in the baseline recognizer with the modified language model $P_{LM}'(w_i|w_1,\ldots,w_{i-1})$. Suppose that the original language model is a conventional n-gram model with order L. If $N \le L$, we can recast the discriminative N-gram model as a pseudo-conventional L-gram model as the equation below. The power of this discriminative model to distinguish among candidate hypotheses can thus be captured by the pseudo-conventional L-gram model.

$$P_{L-gram}'(w_i \mid w_{i-L+1}, w_{i-L+2}, .., w_{i-1}) = P_{L-gram}(w_i \mid w_{i-L+1}, w_{i-L+2}, .., w_{i-1}) + \frac{1}{a_0 \cdot \beta}(a_{w_i} + a_{w_{i-1}w_i} + \ldots + a_{w_{i-N+1}w_{i-N+2}\ldots w_i}) \quad (7.10)$$

7.3.2 Model Computation

The pseudo-conventional n-gram model can be computed based on Equation 7.10 using two possible methods:

Method 1: Compute the pseudo-conventional n-gram model *offline*

A complete pseudo-conventional n-gram model can be built by modifying the n-gram entries in the original n-gram model incorporated in the baseline recognizer using Equation 7.10. The difficulty lies in the fact that the n-gram model in the recognizer normally does not contain all possible n-grams. This is due to the usage of the back-off strategy for n-gram modeling. Given an n-gram model, an n-gram probability may not be included and may be computed via back-off to lower-order n-grams. For example, a bigram not included is calculated as follows:

$$p_{bigram}(w_2 \mid w_1) = b(w_1)\, p_{unigram}(w_2) \quad (7.11)$$

where $p_{brigram}(w_2|w_1)$ and $p_{unigram}(w_2|w_1)$ are bigram and unigram probabilities respectively. $b(w_1)$ is the back-off weight of w_1.

For n-grams that are absent from the original n-gram model but are updated by Equation 7.10, we can insert them into the model as new entries. But this may cause the resulting model to

be too large. An alternative method is to keep the model size unchanged and adjust the related back-off weights and/or probabilities of lower-order n-grams. However, the adjustment of backup weights and lower-order n-grams is controversial [69].

Method 2: Compute the pseudo-conventional n-gram likelihoods *online*

This approach does not create a physical model and computes the pseudo-conventional n-gram likelihoods only when they are needed for either decoding or lattice rescoring. Thus, the problem caused by the back-off strategy can be circumvented. Section 7.4 describes the details of how to compute the pseudo-conventional n-gram likelihoods online for lattice rescoring. The online calculation of pseudo-conventional n-gram likelihoods for single-pass decoding is similar.

7.4 Discriminative Lattice Rescoring

A pseudo-conventional n-gram model can be applied either in a single-pass decoding procedure or during post-processing. This work uses the pseudo-conventional n-gram model to rescore recognition lattices generated by the baseline recognizer. The pseudo-conventional n-gram likelihoods are computed online when needed. We refer to this process of lattice rescoring as discriminative lattice rescoring (DLR).

In a recognition lattice, each word hypothesis along with its acoustic and LM likelihoods is stored in either a node or a link. As discussed in Section 1.2, if the lattice is generated by a conventional L-gram model, the $(L\text{-}1)$-word history for each word hypothesis is unique. Figure 1.2 provided a sample recognition lattice generated by a recognizer with a trigram model. To facilitate reading, we redraw it as Figure 7.2.

The basic idea of DLR is to replace the original LM likelihood with the pseudo-conventional n-gram likelihood for each word node/link in a lattice based on the word history. As shown in Equation 7.10, the calculation of the pseudo-conventional L-gram likelihood is composed of two parts: (1) the score from the original L-gram model, and (2) the score from the discriminative N-gram model. For each word node/link, the original L-gram score is exactly the original LM likelihood, which has already been stored in the node/link in focus. The

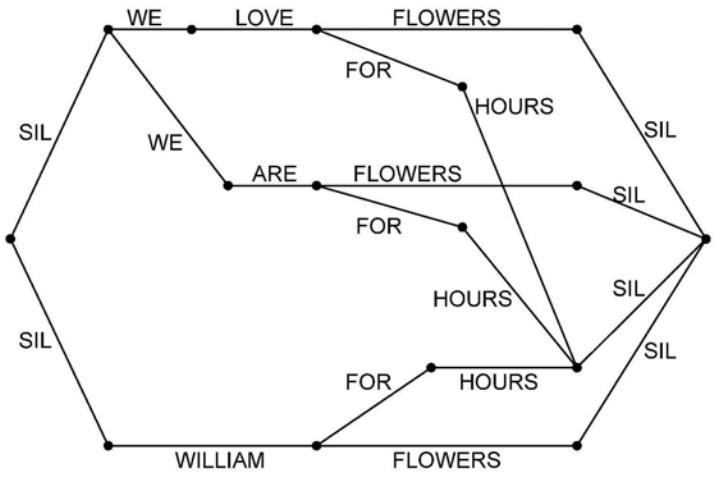

Figure 7.2: A sample recognition lattice generated by trigram decoding ("SIL" marks pauses)

discriminative *N*-gram score for each word node/link can be computed unambiguously based on the corresponding (*N-1*)-word history. This is because the (*N-1*)-word history for every word node/link is always unique. In a recognition lattice generated by *L*-gram decoding, the (*L-1*)-word history for each word node/link is unique. Thus, the (*N-1*)-word history for each word node/link is also unique, since *N* must be no larger than *L* in pseudo-conventional n-gram presentation.

Based on these analyses, we propose the following DLR algorithm. We traverse all word nodes/links in a recognition lattice. For each word node/link, its discriminative *N*-gram score is calculated based on the corresponding (*N-1*)-word history and is then added to the original LM likelihood. The summation is assigned to the word node/link in focus as the new LM likelihood. Having obtained the rescored lattice, the top-scoring utterance hypothesis is identified efficiently by the A* search. This selected hypothesis is the one having the highest $g(h, \vec{a})$ value among all utterance hypotheses in the lattice search space.

As mentioned in Section 1.2, some recognizers may merge two word hypotheses that have the same identified word and the same starting/ending times if (1) their LM likelihoods are the same and (2) the merging will not cause ambiguities in the assignment of *L*-gram likelihoods for the subsequent word hypotheses. In this case, the (*L-1*)-word history for a word node/link may not be unique in a recognition lattice. This problem can be solved by various methods. For example, the function of merging word nodes/links satisfying the two conditions above can be disabled for generating recognition lattices. Another convenient approach is to insert duplicate

word nodes/links for those word nodes/links having ambiguous (*L-1*)-word histories to ensure that each word node/link has a unique (*L-1*)-word history.

7.5 Experiments and Analyses

7.5.1 Settings

We use a disjoint Mandarin speech corpus, referred as DT_Set, to train the discriminative n-gram models. This corpus contains 84,498 read utterances, which are recorded in Microsoft Research Asia. As for the development sets of the ED-EC framework, DT_Set is also in the domain of novels. We evaluate the discriminative models on the general test TestSet_G and the novels-domain test set TestSet_N that are mentioned in Section 4.1. All recognized utterances, *N*-best hypotheses and recognition lattices involved in this chapter are generated by the baseline recognizer described in Section 4.2.

We adopt discriminative bigram modeling in this study. The features include the recognizer score and the counts of unigrams and bigrams. Since it was shown in [105] that the benefit of adding trigram features is limited, we focus on unigrams and bigrams for simplicity and efficiency. We use the lexicon entries in forming unigrams. All the word pairs in the 20-best hypotheses of the training data DT_Set are included as bigrams. There are 3,657,348 bigrams in total.

With these features, we train discriminative models on DT_Set using the average perceptron algorithm. We initialize the weight for the base feature (i.e., the recognition score) at 0.8. The weights for other features are initialized at 0. All the feature weights are updated in the following way during the training procedure: the size of the learning step is set to be 0.01 and the number of iterations is set to be 60. Our previous effort [147] showed that more iterations may lead to better performance, especially if the training and test conditions are similar in nature. However, since the objective here is to investigate the feasibility of discriminative lattice rescoring instead of to develop optimal discriminative n-gram models, we did not optimize the iterations in this study.

Discriminative n-gram models are trained on certain numbers of *N*-best hypotheses. To facilitate the discussion, we denote the number of *N*-best hypotheses for each speech utterance for the *training* of a discriminative model as N_{TR}. A given discriminative model is applied to re-rank the *N*-best hypotheses and/or to rescore the recognition lattices on test data. We denote the number of *N*-best hypotheses for each speech utterance in *testing* as N_{TE}.

7.5.2 Model Development

We investigate the influence of N_{TR} on the effectiveness of discriminative n-gram modeling. First, a series of discriminative bigram models are trained on the 84,498 utterances in DT_Set using different N_{TR}s, as illustrated in Figure 7.3. The model trained with N_{TR} =*m* is named as Model_N*m* (*m=20, 50, 100, 500, 1000*). We then compare the performance of these discriminative models on test data. The five models are used to re-rank the 1000-best testing hypotheses, as illustrated in Figure 7.4. The results are shown in Figure 7.5.

From Figure 7.5, the first observation is that for TestSet_N, adding a greater number of *N*-best hypotheses in training smoothly improves the performance. The CER quickly drops from 19.9% to 17.1% when the training *N*-best number increases from 1 to 100. After that, the CER slowly converges to around 16.3%. The model trained on 1000-best hypotheses achieves a relative CER reduction of 18.1% on TestSet_N over the recognizer baseline.

The second observation is that for TestSet_G, the CER curve fluctuates greatly. Among the five discriminative models, only the model trained on the 20-best hypotheses is beneficial, bringing a very small improvement of 0.08% absolute CER reduction over the baseline performance. All the other models perform worse than the baseline. The poor performance on TestSet_G is due to the domain discrepancy. While the training data DT_Set is in the domain of novels, the TestSet_G is in general domain. As reported in [147], discriminative training with perceptron is domain-sensitive. For testing data similar in nature to training data, perceptron training is very effective. However, for testing data whose domain is different from that of training data, perceptron over-trains easily. This is because of its indiscriminate way of selecting and tuning features. If a general-domain speech corpus that is large enough for discriminative training becomes available, a discriminative n-gram model trained on this corpus can bring substantial improvement on TestSet_G.

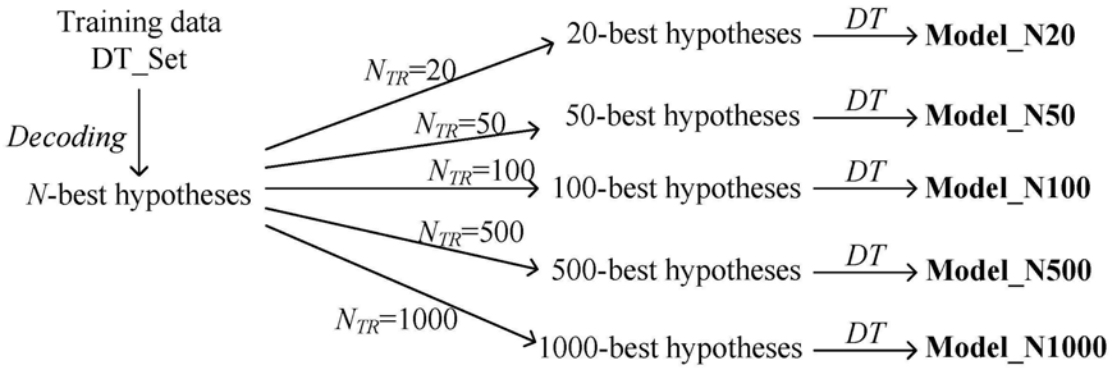

Figure 7.3: The development of discriminative bigram models. N_{TR} refers to the number of *N*-best hypotheses for each speech utterance for the *training* of a discriminative model. *DT* refers to the procedure of training a discriminative n-gram model.

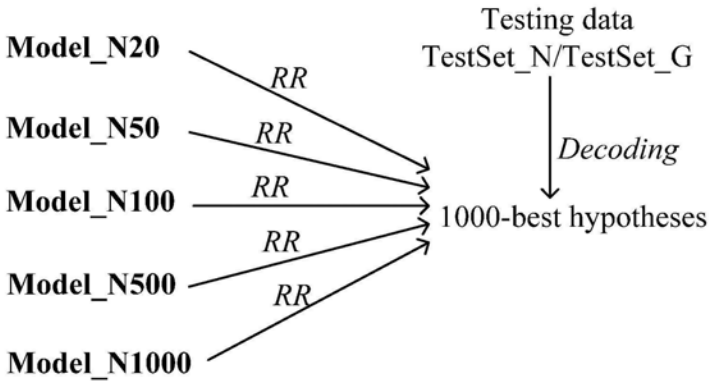

Figure 7.4: The comparison of the discriminative bigram models. *RR* refers to the procedure of re-ranking *N*-best hypotheses

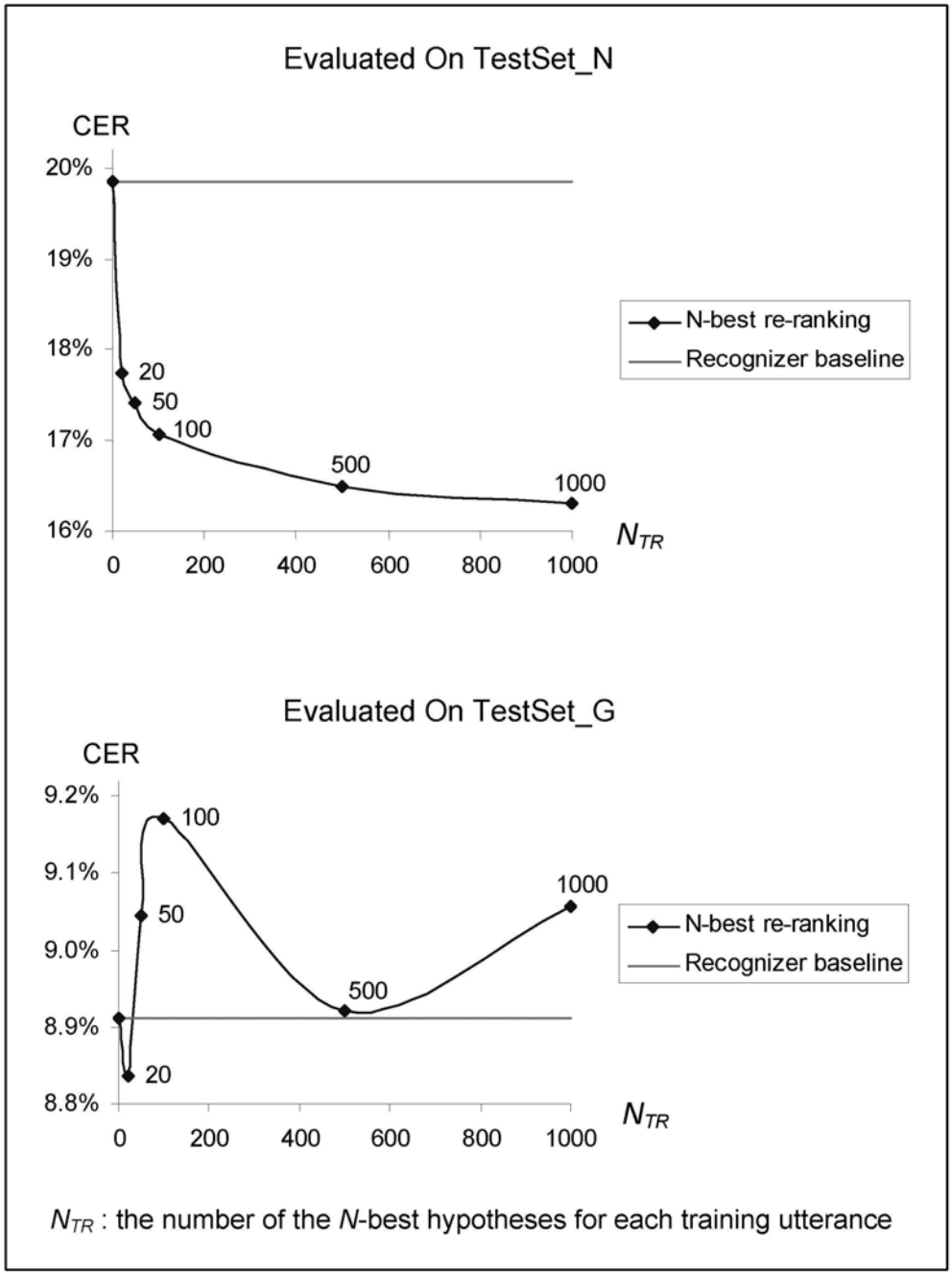

Figure 7.5: The performance of discriminative bigram models trained on various numbers of N-best hypotheses

We selected Model_N20 and Model_N1000, which are trained on the 20-best hypotheses and the 1000-best hypotheses, respectively, to cover discriminative models with various levels of effectiveness. The subsequent experiments are focused on the two models.

The discriminative models are economical in memory consumption. Although the number of total features (i.e., recognizer score, 60,606 unigrams and 3,657,348 bigrams) is very large, the number of active features (i.e., those features whose weights are different from the initial weights after training) is much smaller. Only 12.6% and 17.2% of total features are active for Model_N20 and Model_N1000 respectively. Deleting all inactive features leads to compact discriminative models. The compact models provide the same performance as the corresponding original discriminative models while consuming much less memory.

The cost of perceptron training depends on the number of *N*-best hypotheses adopted in training. For example, while Model_N20 was trained within twelve minutes, the training of Model_N1000 needed nearly two days. The computational load for linguistic scoring and hypothesis ranking increases when the training *N*-best number grows. When N_{TR} is so big that it becomes infeasible to store all training hypotheses in memory, the training procedure will be very slow, because training hypotheses need to be repeatedly read into and removed from memory during iterations.

7.5.3 DLR vs. Discriminative *N*-Best Re-Ranking

In previous works, discriminative n-gram models are used to re-rank the *N*-best hypotheses using Equation 7.1 [106]. We refer to this procedure as discriminative *N*-best re-ranking. In this subsection, we present the performance of discriminative *N*-best re-ranking for Model_N20 and Model_N1000. Then, the performance of discriminative lattice rescoring is evaluated and compared with the discriminative *N*-best re-ranking performance.

- **Discriminative *N*-Best Re-Ranking**

We apply Model_N20 and Model_N1000 in re-ranking various numbers of the testing *N*-best hypotheses. We increase N_{TE} (i.e., the number of *N*-best hypotheses for each speech utterance in testing) from 1 to 1000. The performance of discriminative *N*-best re-ranking on TestSet_N and TestSet_G are illustrated in Figures 7.6 and 7.7 respectively.

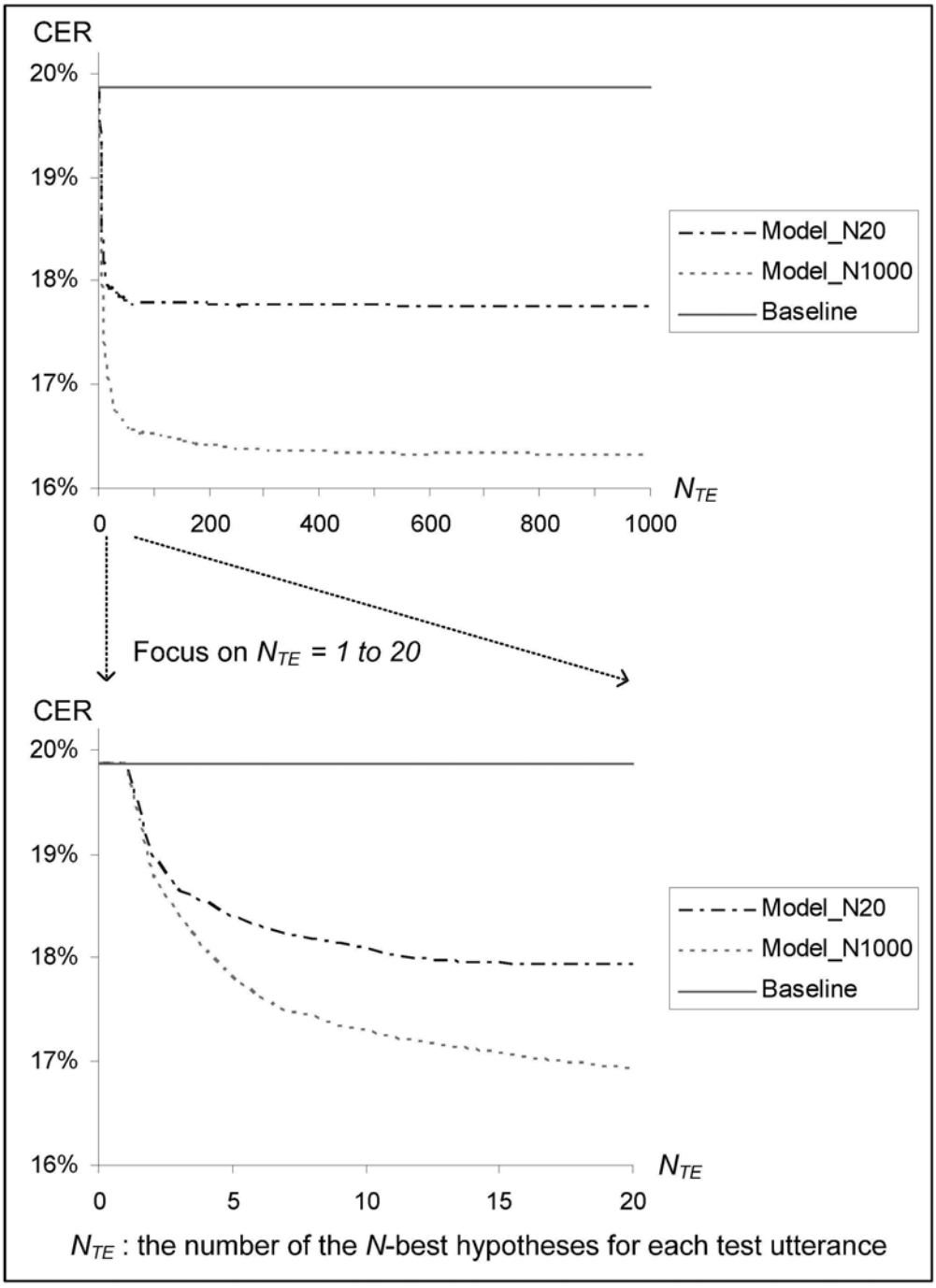

Figure 7.6: Performance of discriminative N-best re-ranking on TestSet_N.

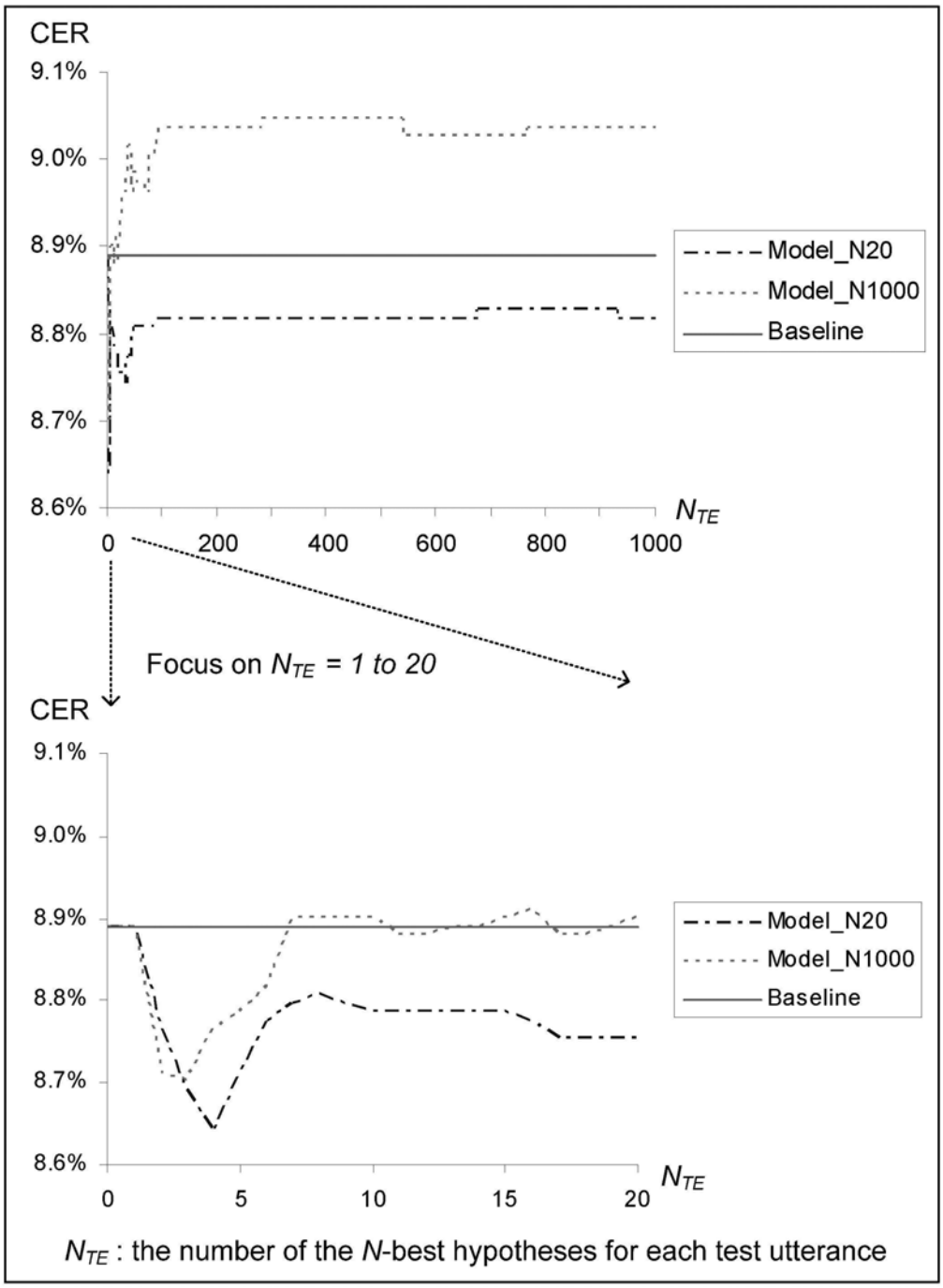

Figure 7.7: Performance of discriminative *N*-best re-ranking on TestSet_G.

Figure 7.6 shows that on TestSet_N, adopting a greater number of *N*-best hypotheses in testing constantly brings better performance for either discriminative model. When the testing *N*-best number increases, the character error rate (CER) drops quickly at the beginning. Then, the rate at which the CER decreases becomes increasingly slow. The decreasing slopes of CER are almost flat for both models when the *N*-best number is larger than 500. For Model_N20 and Model_N1000, re-ranking the top 20 hypotheses provides relative 9.7% and 14.7% CER reductions, respectively, over the baseline performance, while relative CER reductions for re-ranking the 1000-best hypotheses are 10.6% and 18.1%, respectively. This indicates that most of the benefit comes from the top-20 hypotheses.

For TestSet_G, Figure 7.7 shows that the minimum CERs are achieved by adopting only three or four *N*-best hypotheses in testing. When the testing *N*-best number increases to around 100, the CER curve reaches and fluctuates around a relatively high position for each model. For most of the *N*-best numbers tested, using the Model_N1000 to re-rank *N*-best hypotheses always hurts the performance while the Model_N20 only brings a negligible improvement. These observations are consistent with our previous work [147] in that perceptron training is sensitive to the difference in domain between the training and testing data. In this case, finding the optimal testing *N*-best number on a held-out general domain data set may be beneficial.

- **Discriminative Lattice Rescoring**

For both models (i.e., Model_N20 and Model_N1000), we use the algorithm described in Section 7.4 to perform discriminative lattice rescoring. This approach processes each recognition lattice generated by the baseline recognizer in two steps:

Step 1. Rescore the recognition lattice

We represent the discriminative bigram model of interest as a pseudo-conventional trigram model. We then traverse all the word nodes/links in the lattice and modify their LM likelihoods. For each word node/link w_i, if it has no history, we add $a_{w_i}/(a_0 \cdot \beta)$ to the LM likelihood. Otherwise, we add $(a_{w_i} + a_{w_{i-1}w_i})/(a_0 \cdot \beta)$ to the LM likelihood, where w_{i-1} is the previous word hypothesis connected to w_i in this lattice.

Step 2. Find the top-scoring utterance hypothesis

We perform an A* search in the rescored lattice to find the top-scoring utterance hypothesis.

In this study, we store all the unigram weights a_{w_i} and the bigram weights $a_{w_{i-1}w_i}$ in a red-black search tree. These weights can thus be efficiently identified during rescoring. The results of discriminative lattice rescoring on TestSet_N and TestSet_G are shown in Table 7.1. The performance of discriminative 1000-best re-ranking is also listed for comparison.

CERs on **TestSet_N** %			
	Baseline	DLR	Discriminative 1000-best Re-ranking
Model_N20	19.86	17.74	17.75
Model_N1000	19.86	16.27	16.31
CERs on **TestSet_G** %			
	Baseline	DLR	Discriminative 1000-best Re-ranking
Model_N20	8.89	8.83	8.82
Model_N1000	8.89	9.06	9.04

Table 7.1: Performance of the discriminative lattice rescoring (DLR)

From Table 7.1, we can see that discriminative lattice rescoring provides similar CER reductions to discriminative 1000-best re-ranking. On TestSet_N, discriminative lattice rescoring slightly outperforms the 1000-best re-ranking for both discriminative models. This is consistent with the previous observation that adopting a greater number of hypotheses in testing slowly reduces the CER on TestSet_N when the hypothesis number is larger than 500. Since the effect of discriminative lattice rescoring is equivalent to ranking all the utterance hypotheses in the lattice search space, the performance of discriminative lattice rescoring is the upper bound of the discriminative *N*-best re-ranking performance on TestSet_N. For TestSet_G, discriminative lattice rescoring performs slightly worse than discriminative 1000-best re-ranking. Note that the CER curves for TestSet_G fluctuate when the *N*-best number increases during testing. This observation is not surprising, since discriminative lattice rescoring functionally re-ranks a greater number of *N*-best hypotheses than discriminative 1000-best re-ranking.

The experiments also show that discriminative lattice rescoring is efficient. The discriminative top-ranking hypothesis in a lattice is identified within 0.25 second on average. Rescoring a lattice costs 0.22 second on average, and identifying the top-scoring utterance hypothesis in a rescored lattice takes only 0.03 second on average. As a reference,

discriminative 1000-best re-ranking takes 0.78 second on average to process a speech utterance. Re-ranking all hypotheses to find the discriminatively top-ranking hypothesis in a lattice is even more time-consuming. For both discriminative lattice rescoring and *N*-best re-ranking, most of the computation is devoted to calculating the discriminative scores. The computational load is thus mainly determined by the number of word hypotheses in focus. There are on average 2,908 word hypotheses (nodes/links) in a lattice and 12,120 word hypotheses in the *N*-best (*N*=1000) hypothesis lists of an utterance.

In conclusion, the above observations indicate that when the testing data are similar in nature to the training data, discriminative n-gram modeling can bring significant improvement. In this case, performing discriminative lattice rescoring is especially beneficial. First, by applying discriminative lattice rescoring, the performance upper bound of discriminative *N*-best re-ranking can be achieved. Second, discriminative lattice rescoring can be efficiently implemented. Finally, applying other post-processing techniques, such as the ED-EC framework, on top of the rescored lattices to obtain cumulative improvements is convenient.

7.6 Chapter Summary

This chapter proposes a DLR algorithm to post-process the output of a recognizer. DLR is an extension of discriminative n-gram language modeling, which defines a global linear model to re-rank utterance hypotheses generated during decoding. This chapter demonstrates that a linear discriminative n-gram model can be recast as a pseudo-conventional n-gram model if the order of the discriminative n-gram model is no higher than the order of the n-gram model incorporated in the baseline recognizer. The DLR algorithm then uses the pseudo-conventional n-gram model to rescore recognition lattices generated during decoding. Using DLR is functionally equivalent to using the original discriminative n-gram model to re-rank all hypotheses contained in recognition lattices. Compared with the latter, the process of DLR has two main advantages: (1) discriminative top-ranked utterance hypotheses within the lattice search spaces can be efficiently identified by the A* algorithm and (2) the rescored lattices can be further enhanced with other post-processing techniques (e.g., the ED-EC framework) to achieve cumulative improvement conveniently.

In the next chapter, we will use DLR to provide enhanced baseline recognition systems and extend the ED-EC framework to the discriminatively enhanced systems.

Chapter 8

Applying the ED-EC Framework to Enhanced Baseline Systems

The error detection and correction (ED-EC) framework is theoretically applicable to any given baseline recognition system. In other words, the performance of the baseline system can vary, and the baseline recognition can be either a single-pass decoding or a multi-pass recognition procedure. This chapter attempts to (1) analyze the effectiveness of the framework over baseline systems with difference performance and to (2) investigate the feasibility of applying the ED-EC framework on multi-pass baseline systems.

We first apply the Mandarin ED-EC prototype to the enhanced baseline systems that use a discriminative lattice rescoring technique (i.e., the algorithm proposed in Section 7.4) to post-process the output of the recognizer. The experimental details are described in Section 8.1. We then evaluate the effectiveness of the ED-EC prototype for different baseline systems in Section 8.2 by comparing the prototype performance on the recognizer and enhanced baseline systems. The results show that the error detection procedure, the error correction procedure and the overall ED-EC prototype are all effective for different baseline systems. This implies that the ED-EC framework is potentially widely applicable to recognition systems, not only single-pass decoding systems but also multi-pass recognition systems. Finally, we analyze the prototype performance on unseen data in Section 8.3. The experimental results indicate that the ED-EC framework can be robust to the discrepancies between the training and testing conditions.

8.1 Experiments

The ED-EC prototype has been proposed to detect and correct recognition errors in the output of a state-of-the-art single-pass baseline recognizer. For the current prototype, both the error detection and error correction procedures depend on the recognition lattices that are generated during decoding. The error detection procedure uses lattice probabilities as the features, and the error correction procedure selects candidate alternatives for detected errors from the lattices. If an alternative baseline system can provide corresponding lattices, the ED-EC prototype can be applied to this baseline system without any algorithm modifications. Discriminatively enhanced baseline systems provide the corresponding discriminatively rescored lattices in addition to the recognition results (i.e., the top-scoring utterance hypotheses in rescored lattices). We thus extend the ED-EC prototype to discriminatively enhanced baseline systems simply by developing and evaluating the prototype on the discriminatively rescored lattices in the same way as on the original recognition lattices. The procedure is illustrated in Figure 8.1.

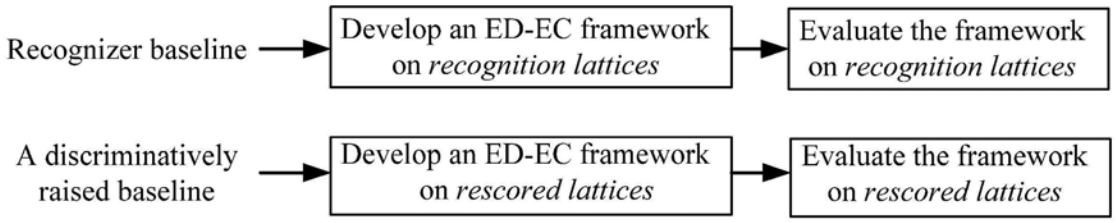

Figure 8.1: The extension of the ED-EC prototype to a discriminatively enhanced baseline recognition system

We develop two discriminatively enhanced baseline systems in this study. The experimental setup is described in Section 8.1.1. Section 8.1.2 presents the training and testing procedures.

8.1.1 Experimental Setup

To obtain two discriminatively enhanced baseline systems, we use two discriminative n-gram models, Model_N20 and Model_N1000 (trained in Section 7.5.2), to post-process the output of the recognizer (trained in Section 4.2) using the discriminative lattice rescoring (DLR) technique.

We refer to the enhanced recognition systems achieved by applying Model_N20 and Model_N1000 as DT_N20 and DT_N1000, respectively. The relationship between the two discriminatively enhanced baseline systems and the original baseline recognizer are illustrated in Figure 8.2.

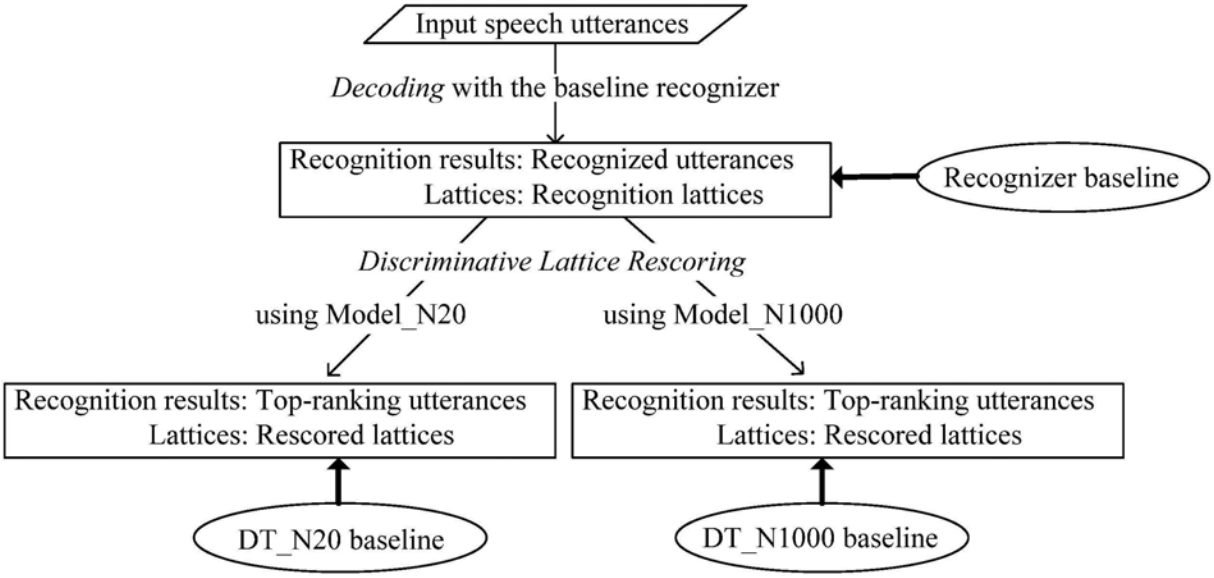

Figure 8.2: The two discriminatively enhanced baselines and the recognizer baseline

To train and test the ED-EC prototype on the discriminatively enhanced baseline systems, we organize the speech and text corpora in the same way as Section 4.1, except that we evaluate the prototype only on the test set TestSet_N in this set of experiments. The other test set TestSet_G is not utilized since TestSet_G is inconsistent with the discriminative training data in domain and DLR consequently fails to improve the recognition performance on this test set, as discussed in Section 7.5. Because this set of experiments aims to investigate the behaviors of the ED-EC framework over baseline systems having different performance, we focus on TestSet_N, on which DLR can effectively bring higher baseline performances.

8.1.2 Training and Testing Procedures

We retrain and test the ED-EC prototype for each enhanced baseline system. For the enhanced system of interest, the recognition lattices generated by the baseline recognizer are rescored by the corresponding discriminative n-gram model. The top-scoring utterances in the rescored

lattices are output as the new recognized utterances (i.e., the recognition results). The ED-EC prototype aims to detect and correct recognition errors in these new recognized utterances. We first retrain the ED-EC prototype on the rescored lattices in the same way as on the original recognition lattices. For the error detection procedure, the acoustic and language model (LM) weights that are used to compute the generalized word posterior probability (GWPP) feature are retuned by performing a ten-fold cross validation on ErrDect_Set. Based on the new acoustic and LM weights, the word verifier is retrained on the whole ErrDetect_Set. For the error correction procedure, we directly limit the size of a candidate list to 20. The interpolation weights of the three individual linguistic models in linguistic scoring as well as the two thresholds (i.e., f_a and f_b in Equation 3.2) in the special mechanism to handle false alarms are retuned on ErrCorr_Set. We then evaluate this retrained ED-EC prototype on TestSet_N based on the new recognized utterances and rescored lattices for the enhanced baseline system of interest. Figure 8.3 lists the procedures to retrain and test the ED-EC prototype for a discriminatively enhance baseline system.

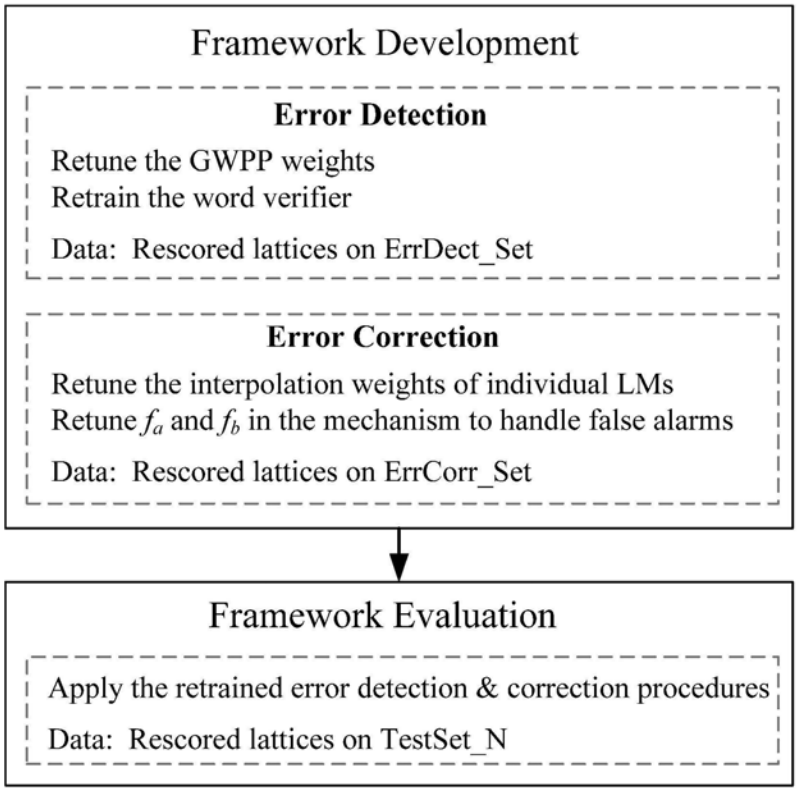

Figure 8.3: Retraining and testing the ED-EC prototype
for a discriminatively enhanced baseline recognition system

8.2 Results and Analyses

8.2.1 Performance of Various Baseline Systems

As discussed in Section 8.1.1, all experimental results are evaluated on TestSet_N in this chapter. The performance of the discriminatively enhanced baseline systems and the recognizer are listed in Table 8.1. Over the recognizer baseline, rescoring recognition lattices using Model_N20 brings a 10.7% relative reduction in CER, and rescoring lattices using Model_N1000 achieves an even larger relative CER reduction of 18.1%. Table 8.1 also shows that for all three baseline systems, insertions and deletions account for only a very small portion of the total recognition errors. This further supports the decision of the ED-EC prototype to focus on substitutions to reduce the system complexity.

Baseline	CER %	Number of Character Errors		
		Substitution	Insertions	Deletion
Recognizer	19.9	11,721	213	516
DT_N20	17.7	10,428	194	501
DT_N1000	16.3	9,535	191	475

Table 8.1: Performance of various baseline systems

8.2.2 Effectiveness of Error Detection

The error detection performance for each of the different baseline systems is shown in Table 8.2. The detection error rates (DERs) and the balanced F-measures are compared. The two error detection related rates R_ED_{NN} and R_ED_{PN}, which are critical for the effectiveness of the ED-EC framework as discussed in Section 4.2, are also considered. R_ED_{NN} is the rate that misrecognized characters are detected as errors, while R_ED_{PN} is the rate that a correct character is wrongly detected as an error.

Baseline	CER %	DER %	F-measure for Correct Characters %	F-measure for Erroneous Characters %	R_ED_{NN} %	R_ED_{PN} %
Recognizer	19.9	14.5	91.2	58.9	54.4	7.1
DT_N20	17.7	13.2	92.2	57.8	53.1	6.3
DT_N1000	16.3	12.1	93.0	56.2	50.1	5.2

Table 8.2: Error detection performance for various baselines

The results above show that for the discriminatively enhanced baselines, the error detection performances are comparable to those for the recognizer baseline. When the baseline decreases, the detection performance for correct characters improves, whereas that for erroneous characters worsens. This is possibly because for enhanced baselines, there are more correct words and the less erroneous words in the training data of the word verifier, making the resulting word verifier perform better on correct words and worse on erroneous words. As illustrated in Table 8.2, when the baseline CER decreases, fewer recognition errors are identified, but a smaller portion of correct characters are wrongly labeled as errors. Since the number of correct characters is relatively large, the improvement for correct characters overcomes the performance decrease for erroneous characters. This leads to the result that the better the baseline, the lower the overall detection error rate.

We further analyze the error detection performance in terms of utterance clustering. For each baseline, we cluster the test utterances into three utterance subsets based on the number of detected errors in the same way as described in Section 4.3.2. The utterances that contain zero, from one to four, and more than four detected errors are labeled as correct, lightly erroneous and seriously erroneous respectively. The performances of utterance clustering are included in the table 8.3. The results show that the error detection procedure performs similarly in terms of utterance clustering for the various baselines. For every type of utterance set, the CERs are comparable. We also notice that when the baseline system is better, more utterances are labeled as correct and fewer utterances are labeled as lightly/seriously erroneous. This reflects the effectiveness of the error detection. In addition, for the discriminatively enhanced baselines, more utterances are filtered from further error correction processing. This raises the efficiency of the ED-EC framework.

Baseline	Correct Utterances (0 error)		Lightly erroneous Utterances (1-4 errors)		Seriously erroneous Utterances (>4 errors)	
	Utt. Num	CER %	Utt. Num	CER %	Utt. Num	CER %
Recognizer	1,362	6.4	1,790	20.4	848	37.1
DT_N20	1,541	5.2	1,764	19.6	695	36.6
DT_N1000	1,776	5.5	1,668	19.1	556	37.3

Table 8.3: The character error rates (CERs) of the utterance subsets of correct, lightly erroneous and seriously erroneous. Utterances are labeled based on the number of detected errors.

8.2.3 Effectiveness of Error Correction

For each enhanced baselines, we apply the retrained error correction procedure on the corresponding lightly erroneous utterance subset. The results are illustrated in Figure 8.4. Figure 8.4 shows that when the baseline CER decreases, the number of detected errors also decreases. However, the ratio of true to false errors is about 1.8 for all the baselines. Table 8.4 lists the two correction rates for the true and false errors. R_T refers to the rate of successfully correcting true errors. R_F refers to the rate of remaining correct after correction for false errors.

Baseline	R_T %	R_F %
Recognizer	26.3	73.9
DT_N20	19.0	82.4
DT_N1000	21.7	71.9

Table 8.4: R_T and R_F on the lightly erroneous utterance sets for various baselines

In Table 8.4, the first observation is that the correction rates for the true errors are relatively low for the discriminatively enhanced baselines compared with that for the recognizer baseline. This is partially because the misrecognitions of the enhanced systems tend to be more difficult to correct. Those that are easy to correct may have already been corrected through discriminative lattice rescoring. The second observation is that both R_T and R_F depend on the specific set of detected errors. DT_N1000 has a lower CER than DT_N20. Intuitively, we

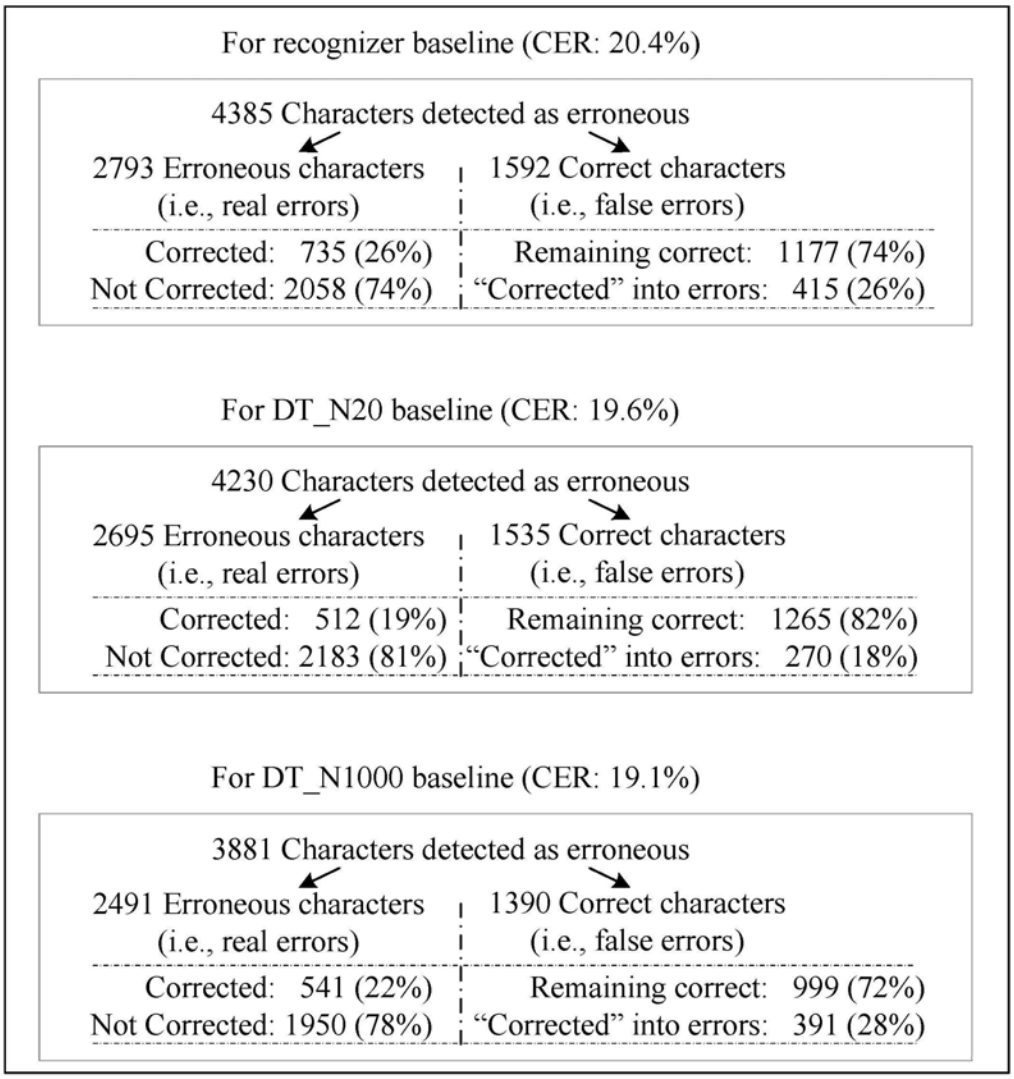

Figure 8.4: Error correction details on the lightly erroneous utterance set for various baselines

expect that the true errors for DT_N1000 are more difficult to correct. However, R_T for DT_N1000 is slightly higher than that for DT_N20. This means that the correction ability is greatly influenced by the specific errors along with their corresponding context. The fluctuation of R_F across various baselines also proves this point.

Evaluated in terms of CER, the performances of error correction on the utterance sets of lightly erroneous are shown in Figure 8.5.

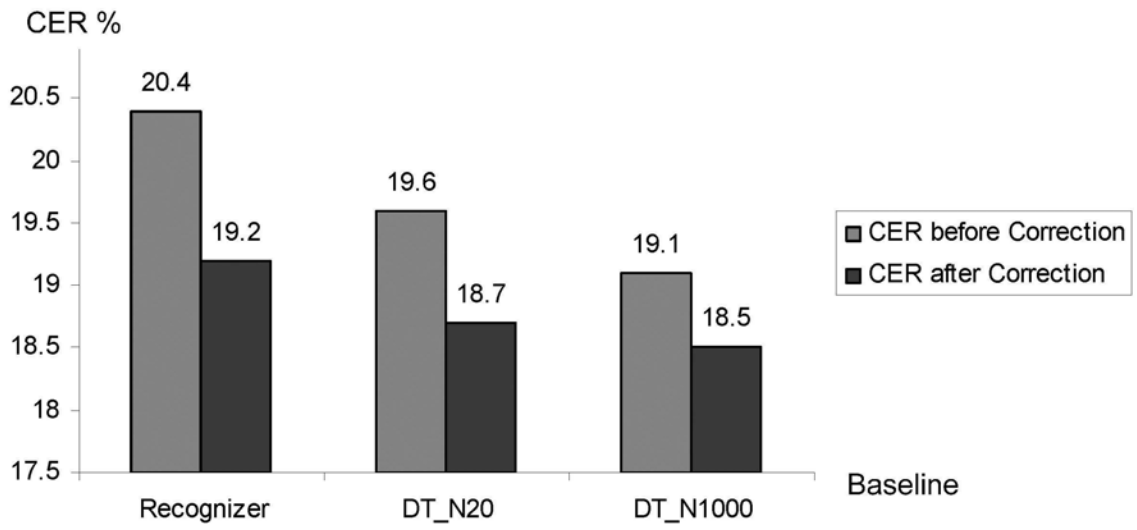

Figure 8.5: The reduction in CER brought by the error correction procedure, evaluated on the lightly erroneous utterances

Figure 8.5 shows that the better the baseline, the smaller the CER reduction. The relative CER reductions are 5.9%, 4.6% and 3.1% over the recognizer baseline, DT_N20 baseline and DT_N1000 baseline respectively. As discussed in Section 5.1.3, the CER reduction is determined by the number of misrecognitions corrected and the number of new wrong characters introduced by "correcting" false errors, along with the total number of characters. The CER reductions over the discriminatively enhanced baselines are smaller than that over the recognizer baseline, mainly because fewer true errors are corrected. Fewer true errors are detected and the correction rates R_T are lower for the enhanced systems. Compared with DT_N20, DT_N1000 has a higher R_T, and its CER reduction is smaller because many more false errors are transferred into new errors, which limits the overall improvement.

8.2.4 Overall Framework Effectiveness

We further evaluate the ED-EC framework on the entire TestSet_N for the various baselines. The results are presented in Table 8.5. We can see that applying the ED-EC framework consistently improves the baseline performance. This verifies the effectiveness of the framework and indicates its feasibility for application to multi-pass baseline recognition systems. Another

observation is that the benefit brought by the framework decreases when the baseline performance increases. Recall from Section 4.4.4 that on TestSet_G, the framework achieves greater improvement (i.e., 6.0% relative CER reduction) over a high recognizer baseline (i.e., 8.9% CER). This conflict shows that the effectiveness of the ED-EC framework depends on multiple factors. These factors include (1) how well the knowledge sources adopted in the ED-EC framework match the test data in domain and (2) how difficult the recognition errors are for the baseline of interest.

Baseline	CER before Correction %	CER after Correction %	Absolute CER Reduction %	Relative CER Reduction %
Recognizer	19.86	19.35	0.51	2.6
DT_N20	17.74	17.36	0.38	2.1
DT_N1000	16.27	16.03	0.24	1.5

Table 8.5: The overall reduction in CER, evaluated on the full test set

8.3 Further Discussions

In previous sections, we view the baseline recognition system as a black box that provides recognized utterances along with corresponding lattices. For each recognizer/enhanced baseline system, an ED-EC prototype is trained and tested in the same way. The training and testing conditions are known to be consistent. This section investigates the performance of the ED-EC framework on unseen data. A trained ED-EC framework may be applied to an unseen set of recognized utterances and corresponding lattices which are generated in a scenario different from the training situation. This mismatch in training and testing conditions leads to performance decline normally. Since in real applications it is hard to guarantee that the input data is similar in nature to the training data, the capability of the ED-EC framework to handle unseen data is an interesting topic. We investigate the effectiveness of the error detection and correction procedures on unseen data in Section 8.3.1 and Section 8.3.2, respectively.

8.3.1 Error Detection on Unseen Data

The error detection procedure detects erroneous characters using the GWPP feature described in Section 3.3. The GWPP feature is calculated based on the acoustic and LM likelihoods in the lattices. If the training and testing scenarios use different methods to assign likelihoods to the nodes/links in the lattices, the effectiveness of error detection may be affected. We evaluate the error detection effectiveness on unseen data by directly applying the error detection procedure trained for the recognizer baseline system to the enhanced baseline systems. In this case, the acoustic and LM likelihoods in the training lattices are directly assigned by the recognizer, whereas the LM likelihoods are modified in the testing lattices via discriminatively lattice rescoring. We calculate the GWPPs for the testing data in the same way as for the training data, as if we do not know the difference in assigning likelihoods. The error detection results are presented in Table 8.6. We refer to the error detection procedure trained for the recognizer baseline system as ED_rec and refer to the error detection procedure retrained for the corresponding discriminatively enhanced baseline system as ED_dt.

On DT_N20 baseline					
	DER %	F-measure for Correct Characters %	F-measure for Erroneous Characters %	R_ED_{NN} %	R_ED_{PN} %
ED_rec	13.20	92.20	57.1	51.6	6.0
ED_dt	13.21	92.17	57.8	53.1	6.3
On DT_N1000 baseline					
	DER %	F-measure for Correct Characters %	F-measure for Erroneous Characters %	R_ED_{NN} %	R_ED_{PN} %
ED_rec	12.19	92.92	56.0	49.7	5.2
ED_dt	12.15	92.95	56.2	50.1	5.2

Table 8.6: The robustness of error detection on unseen data

Table 8.6 shows that the performance achieved by ED_rec is similar to that of ED_dt on both enhanced systems, indicating the robustness of error detection on mismatching unseen data. The difference in F-measures of correct characters is negligible for both baselines. ED_rec even

slightly outperforms ED_dt on the DT_N20 baseline, because it labels fewer correct characters as errors. For erroneous characters, retraining the error detection procedure consistently increases the detection rate R_ED_{NN} and the F-measure is reduced by 1.2% and 0.4% for DT_N20 and DT_N1000 respectively.

8.3.2 Error Correction on Unseen Data

The proposed error correction procedure has three major components: candidate creation, hypothesis re-ranking with linguistic knowledge sources, and the additional mechanism to handle false alarms of error detection. In this subsection, we discuss the effectiveness of these three components on unseen data. Then we evaluate the overall effectiveness of the error correction on unseen data.

- **Candidate creation**

As discussed in Section 8.1.2, we use the same approach to create candidate alternatives for errors on either the recognizer or discriminatively enhanced baselines. For each error, character hypotheses with similar starting and ending times are selected from the corresponding lattice and inserted into the candidate list. The candidate list is then pruned, and only the top 20 characters with the highest generalized character posterior probabilities (GCPPs) are retained. As illustrated in Equation 3.5, GCPPs are derived from word posterior probabilities and thus depend on the acoustic and LM likelihoods in the lattices. When the lattices are rescored, the performance of candidate creation may be affected. To evaluate the influence of lattice rescoring in candidate creation, we compare the performance of candidate creation on the various baselines. For clarity, here we assume that the error detection procedure is perfect, and candidate alternatives for substitution errors are created. The comparison results are presented in Figure 8.6. We can see that when the baseline performance increases, the reference coverage rate (RCR) slightly decreases. Note that the average sizes of the unpruned/pruned candidate lists are comparable for different baselines. The RCR loss caused by pruning is very small for all three baselines.

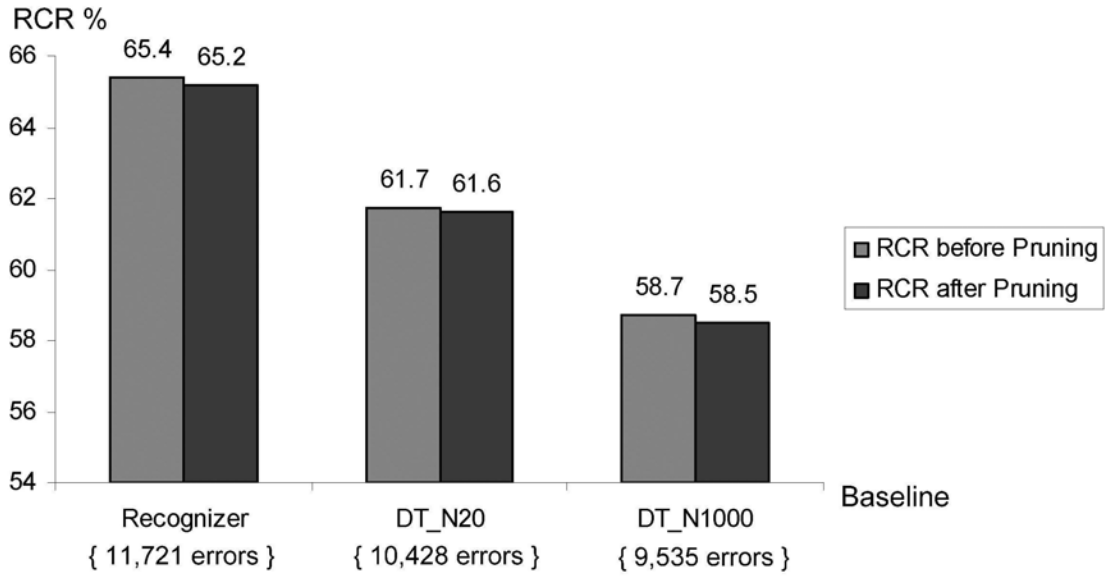

Figure 8.6: Reference coverage rate (RCR) of candidate lists created for substitutions.

The various baselines have different sets of substitution errors. For each baseline, we further divide its substitution errors into two subsets. One subset, referred to as the common subset, stores those errors shared by all three baselines. The other contains the remaining errors and is referred to as the private subset. In other words, the three baselines share the same common subset and have different private subsets. The candidate creation performances for the two subsets are compared in Table 8.7. All values are RCRs after pruning. The performance losses due to pruning are small for both the common and private subsets for every baseline.

Baseline	Overall RCR %	RCR for the Common Subset %	RCR for the Private Subset %
Recognizer	65.2	56.08	78.8
DT_N20	61.6	56.14	72.8
DT_N1000	58.5	56.13	65.2

Table 8.7: Reference coverage rate (RCR) for common and private error subsets

Table 8.7 shows that for all three baselines, the RCRs on the subset of common errors are very similar to one another (about 56.1%). This indicates that for common errors, discriminative lattice rescoring almost has no impact on candidate creation. Another interesting observation is that the RCR for the private subset decreases greatly when the baseline performance improves. This may be because errors tend to be more difficult for a baseline system with better

performance. Many easy errors have already been corrected by performing discriminatively lattice rescoring for the enhanced baseline systems. For the remaining errors, which are relatively difficult, the ability of the decoder to include the correct answers into the lattice search space is reduced. This also explains why the RCRs for the common subset are much lower than those for the private subsets. Generally speaking, common errors are the most difficult ones to correct among all errors. They normally appear in those dramatic expressions (e.g., the utterance "浑身长着金鳞的菠萝堆成一座座小山散发着浓香" Translation: "Pineapples whose bodies are covered by golden scales are piled into small mountains which smell sweet"), which rarely occur and are difficult to model.

- **Knowledge combination for hypothesis re-ranking**

For hypothesis re-ranking, different linguistic knowledge sources are linearly combined using a set of interpolation weights. We investigate the effectiveness of interpolation weights on unseen data by applying a trained set of interpolation weights to data that is different from the training corpus. More specifically, we modify the ED-EC framework developed for a discriminatively enhanced baseline by replacing its interpolation weights with the original interpolation weights (i.e., the ones in the ED-EC framework trained for the recognizer baseline). We then apply the modified ED-EC framework on the enhanced baseline system of interest. The procedures are illustrated in Figure 8.7. The error correction results on the utterance sets of lightly erroneous in TestSet_N are shown in Table 8.8. To clearly show the effectiveness of the interpolation weights, the additional mechanism used to handle false alarms is not utilized in this set of experiments.

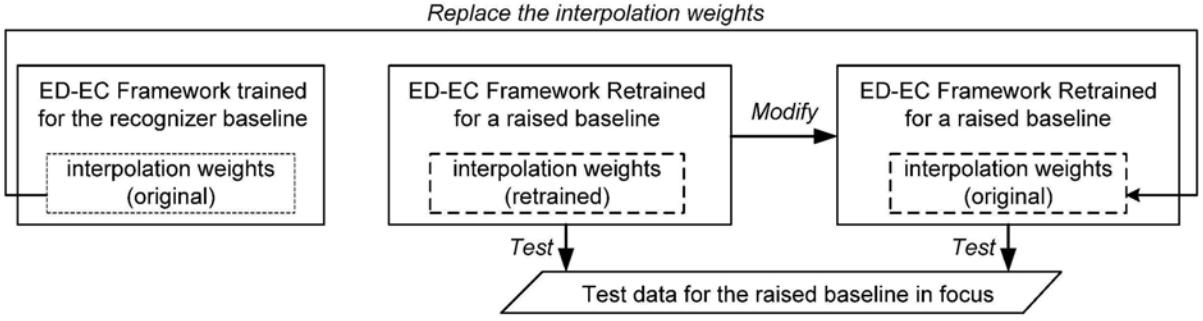

Figure 8.7: The procedures to evaluate the effectiveness of interpolation weights on unseen data.

Baseline	CER before Correction%	CER after Hypothesis Re-ranking %	
		Retrained Weights	Original Weights
DT_N20	19.56	19.04	19.02
DT_N1000	19.09	18.95	18.94

Table 8.8: The effectiveness of linguistic scoring on unseen data, evaluated on the lightly erroneous utterance sets.

In Table 8.8, Original Weights refers to the performance of the retrained ED-EC framework with the original interpolation weights, and Retrained Weights refers to the performance of the retrained framework with retrained interpolation weights. From Table 8.8, we can see that this knowledge combination is effective across various baseline systems. The original weights even slightly outperform the retrained ones for both of the enhanced baselines. This means that when adapting the ED-EC framework to another baseline recognition system, retraining the interpolation weights may be not necessary.

- **Mechanism to handle false alarms**

In the special mechanism to handle error-detection false alarms, the two thresholds f_a and f_b may also be affected as the baseline system varies. Similarly, we analyze the effectiveness of this mechanism on unseen data by replacing the f_a and f_b in the ED-EC framework retrained for an enhanced baseline with the ones trained for the recognizer baseline. The error correction results are shown in Table 8.9, evaluated on the corresponding lightly erroneous utterance sets for each enhanced baseline. Original Thresholds refers to the f_a and f_b trained for the recognizer baseline, and Retrained Thresholds refers to the ones retrained for the corresponding enhanced baseline.

Baseline	CER before Applying this Mechanism %	CER after Applying this Mechanism %	
		Retrained Thresholds	Original Thresholds
DT_N20	19.02	18.71	18.73
DT_N1000	18.94	18.54	18.56

Table 8.9: The effectiveness of the mechanism to handle false alarms on unseen data, evaluated on the lightly erroneous utterance sets

The results in Table 8.9 show that retraining f_a and f_b improves the effectiveness of the framework. However, the differences between the performance of the original and retrained thresholds are very small, being about 0.02% CER for both of the enhanced baselines.

- **Overall error correction procedure**

We analyze the overall effectiveness of the error correction procedure on unseen data by directly applying the error correction procedure trained for the recognizer baseline to the discriminatively enhanced baselines. The CERs on the utterance sets of lightly erroneous are shown in Table 8.10.

Baseline	CER before Correction%	CER after Correction %	
		Retrained ED Proc	Original ED Proc
DT_N20	19.56	18.71	18.75
DT_N1000	19.09	18.54	18.61

Table 8.10: The effectiveness of the overall error correction procedure on unseen data, evaluated on the lightly erroneous utterance sets

In Table 8.10, Original ED Proc refers to the error correction procedure trained for the recognizer baseline, and Retrained ED Proc refers to the error correction procedure retrained for the corresponding enhanced baselines. Table 8.10 illustrates that the original error correction procedure trained for the recognizer system is effective at correcting errors of the enhanced systems. This indicates that the error correction procedure can be effective across various testing scenarios. In addition, the table indicates that retraining the error correction procedure for enhanced baselines is beneficial. Over the DT_N20 baseline, the relative CER reduction using the Original ED Proc is 4.1%, while the Retrained ED Proc achieves a 4.6% relative CER reduction. Over the DT_N1000 baseline, the relative CER reductions are 2.5% and 3.1%, for the Original ED Proc and Retrained ED Proc respectively.

8.4 Chapter Summary

This chapter first investigates the effectiveness of the ED-EC framework across various baseline systems. We post-process the output of the baseline recognizer using the discriminative lattice

rescoring technique described in Chapter 6 to provide discriminatively enhanced baseline systems. We train and test the ED-EC framework for each discriminatively enhanced baseline system in the same way as for the original baseline recognizer. Experiments show that the ED-EC framework achieves consistent improvements over various baseline systems, indicating that the framework can effectively improve the performance of both single-pass and multi-pass recognition systems. Further, this chapter evaluates the effectiveness of the ED-EC framework on unseen data. We directly use the ED-EC framework trained for the baseline recognizer to improve the discriminatively enhanced baseline systems. Experimental results demonstrate that the error detection procedure is robust to the discrepancy in training and testing conditions. For error correction, the influence of this training-testing discrepancy is a bit large.

Chapter 9

Summary and Future Directions

9.1 Thesis Summary

For large vocabulary continuous speech recognition (LVCSR), the language models utilized in state-of-the-art recognizers are still elementary. The word trigram models which have been prevalent for decades only capture local constraints. To achieve breakthroughs for LVCSR, incorporating advanced language models that model long-distance semantic/syntactic constraints is promising. However, most such efforts face a serious efficiency problem. This is mainly due to the intense computation and high complexity associated with the application of sophisticated models. This thesis proposes an error detection and correction (ED-EC) framework with the aim of taking advantage of advanced knowledge in LVCSR while maintaining efficiency. The framework post-processes the output of a state-of-the-art baseline recognizer by (1) detecting recognition errors and (2) correcting detected errors with the aid of sophisticated language models. Basically, the framework attempts to only apply computationally expensive models selectively, i.e., to the parts of the signals where an error is detected.

The ED-EC framework offers the following main advantages. First, the framework reduces the efficiency problem by selectively applying advanced language models on erroneous regions of signals. This selectiveness in model application differentiates the ED-EC framework from previous efforts that use sophisticated models to indiscriminately process speech segments. Second, using this framework, the optimal gain of the advanced models in focus for LVCSR is theoretically achievable. Third, the framework is general in the sense that the implementation of

the error detection/correction procedures is flexible, and incorporating new linguistic knowledge sources into the error correction is convenient.

In this study, we design a prototype of the ED-EC framework for Mandarin LVCSR. The prototype focuses on detecting/correcting character substitutions, which are the majority of character-based errors for Mandarin recognition. The error detection procedure classifies each recognized character as either correct or erroneous based on posterior probabilities. Those detected erroneous characters are passed on to the subsequent error correction procedure. For error correction, a candidate list of character alternatives is created for every detected character error. The candidate lists are then connected with the utterance context to construct new search spaces, as shown in Figure 9.1. Within the new search spaces, utterance hypotheses are scored by an advanced language model that combines three linguistic knowledge sources: mutual information, word trigrams and POS trigrams. The candidate alternatives contained in top-scoring utterance hypotheses are viewed as the results of error correction. To handle error-detection false alarms, the error correction procedure also adopts an additional mechanism to accept/reject the correction results based on error-detection confidence scores and linguistic scores.

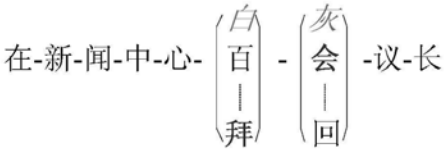

Figure 9.1: A sausage-type search network

The experimental results demonstrate that the ED-EC framework is feasible. On a standard general-domain test set, applying the trained prototype brings a 6.0% relative reduction in character error rate (CER) over the performance of the baseline recognizer. Among the utterances, 50.8% are detected as error-free (i.e., containing no detected character errors). Consequently, the additional computation of the advanced language model (LM) on these utterances is avoided. For those utterances on which the error correction procedure is applied, the relative CER reduction is as high as 10.9%.

We analyze the factors that affect the effectiveness of the framework. These factors include the linguistic knowledge sources adopted, the properties of the sausage-type search spaces and the existence of error-detection false alarms. We observe that the error correction performance varies for different linguistic knowledge sources. Combining multiple sources is

beneficial. For the search networks, both the network size and erroneous status (e.g., the number of undetected errors in the utterance context) have an impact on error correction. The error-detection false alarms lead to a major problem that correct characters may be wrongly "corrected" into new errors. The special mechanism to handle false alarms has been shown to be effective in relieving this problem. Based on these observations, we propose a single equation (i.e., Equation 5.3) to describe the ability of the ED-EC framework to reduce the recognition error rate.

We further analyze other characteristics of the ED-EC framework. First, we evaluate the performance upper bounds of the framework. The results demonstrate the potential benefit of the framework. If both error detection and candidate creation were perfect, the Mandarin prototype would achieve a 36.0% relative CER reduction over the baseline performance on the general-domain test set. Adopting better/more linguistic knowledge sources in error correction may further increase the performance upper bound. Second, we investigate the framework efficiency. We observe that the framework computational load while processing an utterance depends heavily on the number of detected errors. Using efficient algorithms to search for the top-scoring hypotheses in the networks will be especially beneficial. Third, we investigate the effectiveness of the ED-EC framework across various baseline recognition systems. We propose a discriminative lattice rescoring technique to obtain discriminatively enhanced baseline systems. The experimental results show that the ED-EC prototype proposed for the recognizer baseline system can be easily extended to the discriminatively enhanced baseline systems, and the framework improves the recognition performance for both the recognizer and discriminatively enhanced systems. In addition, analyses show that the framework can be robust for unseen data.

We compare the ED-EC framework with other approaches that use advanced linguistic knowledge to benefit LVCSR. Compared with the methods that incorporate advanced LMs into a single-pass decoding procedure, the ED-EC framework is advantageous for two reasons: (1) the difficulty in merging long-distance constraints into the decoding structure can be circumvented and (2) the context for each erroneous region detected can be better utilized. We demonstrate that the framework should outperform the single-pass methods when the error

detection and candidate creation procedures approach perfection [1]. Compared with the algorithms that post-process the recognizer output with sophisticated LMs, the framework can theoretically also achieve better performance. This is because the search spaces of the framework will guarantee the inclusion of correct utterances if error detection and candidate creation are perfect, while other post-processing spaces are normally suboptimal. We use the *N*-best re-ranking technique as an example. The experimental results illustrate that *N*-best re-ranking with the identical advanced LM performs much worse than the ED-EC prototype. Finally, we present an analysis of computation expenses. Both single-pass methods and post-processing approaches typically process all signals indiscriminately. In contrast, the ED-EC framework concentrates the application of sophisticated LMs on detected erroneous regions in signals. The computation saved and expended for the framework is analyzed in detail.

The discriminative lattice rescoring (DLR) algorithm, which is proposed to provide the discriminatively enhanced baseline recognition systems, is another contribution of the thesis. The DLR approach is an extension of discriminative n-gram modeling. It recasts a linear discriminative n-gram model as a pseudo-conventional n-gram model and uses this pseudo model to rescore recognition lattices generated during decoding. Performing DLR is functionally equivalent to using the discriminative n-gram modeling method to rank all utterance hypotheses in recognition lattices, but is much more efficient. It is also possible to apply the pseudo-conventional n-gram model in a single-pass decoding procedure. Section 9.3 will further discuss this possibility.

[1] In this chapter, "Perfect error detection" refers to the case that all recognition errors are correctly detected; "Perfect candidate creation" refers to the case that the reference characters are guaranteed to be present in the candidate lists.

9.2 Contributions

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

- **Propose a novel framework, the error detection and correction (ED-EC) framework, to incorporate advanced linguistic knowledge sources into LVCSR**

  This framework aims to achieve the optimal gain of sophisticated language models at the lowest computational cost. The framework consists of two sequential procedures, error detection and error correction, as described in the previous section. The main idea is to only apply computationally expensive models where an efficient state-of-the-art recognizer fails. This selectiveness in knowledge application differentiates the framework from those previous efforts that process speech segments indiscriminately.

- **Implement a prototype of the framework**

  A prototype of the ED-EC framework on the task of Mandarin dictation is implemented. Note that the ED-EC framework is conceptually language-independent. The Mandarin prototype involves an error detection procedure based on posterior probabilities and a novel error correction procedure that creates/ranks alternatives for every detected error. An advanced LM that combines mutual information, word trigrams and POS trigrams is adopted in error correction.

- **Investigate the characteristics of the framework systematically**

  The ED-EC framework has been analyzed in many aspects with the example of the Mandarin prototype. These aspects include the feasibility, effectiveness, efficiency, performance upper bounds, extensibility to alternative recognition baselines and robustness of the framework. Competitive analyses have also been performed to compare the ED-EC framework with other techniques that adopt advanced linguistic knowledge in LVCSR.

- **Propose a discriminative lattice rescoring (DLR) algorithm**

  The DLR algorithm is proposed to facilitate the investigation of framework extensibility. It can also be used as a general approach to improve LVCSR. This approach is an extension of

discriminative n-gram modeling. It recasts discriminative n-grams as pseudo-conventional n-grams and then performs efficient lattice rescoring. Compared with discriminative n-gram modeling, DLR can achieve equal performance with better efficiency. DLR also facilitates the combination of discriminative n-gram modeling with other post-processing techniques.

9.3 Future Directions for the ED-EC Framework

This thesis proposes and implements an initial prototype of the ED-EC framework for Mandarin LVCSR. This prototype can be improved in various directions for both error detection and error correction. For example, the error detection procedure may utilize more features and/or adopt alternative classification algorithms for the word verifier that classifies each recognized word as either correct or erroneous. The performance of the error correction procedure can be enhanced by refining the candidate creation algorithm, incorporating new knowledge sources and/or using better mechanisms to handle false alarms of error detection. Adopting more efficient approaches to identify the top-scoring utterances in search networks created in error correction will benefit the system efficiency. In this subsection, we discuss in detail possible research efforts for error detection and error correction in Section 9.3.1 and Section 9.3.2 respectively.

Additionally, the application of the ED-EC framework is not limited to Mandarin LVCSR. We demonstrate that the Mandarin ED-ED prototype can be adapted to LVCSR for other languages, such as English, in Section 9.3.3. We also illustrate that it is possible to use the ED-EC framework to benefit spoken dialogue systems in Section 9.3.4.

9.3.1 Improving Error Detection

The current error detection algorithm is relatively simple. It first uses a word verifier to detect erroneous words based on a single feature of Generalized Word Posterior Probability (GWPP). Then, it simply labels every character in detected erroneous words as erroneous. Improving the error detection procedure can significantly benefit the overall effectiveness of the ED-EC framework, as discussed in Section 5.3. The improvement may be due to adopting more features in word verification. Our previous experiments demonstrated that the error detection

performance can be effectively enhanced by combining the GWPP feature with *N*-best based features (i.e., features extracted based on acoustic/LM scores and the purity information from the *N*-best hypotheses) [151]. This illustrates that using multiple information sources in error detection is beneficial. In addition, adopting features that capture higher-level knowledge may be especially helpful. For example, Guo et al. [47] reported that incorporating a mutual information (MI) based feature brought a 7.4% reduction in equal error rate for word verification over the performance of using a word posterior probability feature alone.

Another possible direction for improving error detection is to use better classification algorithms. Choosing a classification method that is both effective and efficient is an interesting topic, especially when multiple features are adopted in error detection. Advanced techniques, such as support vector machines, may lead to better decisions than the simple linear interpolation method [148]. Moreover, involving utterance verification into the error detection procedure may improve erroneous word detection because utterance-based features can capture some useful information that is unavailable for word-based features [51, 148].

It is also possible to design a new scheme for error detection. The current error detection algorithm assumes that (1) all recognition errors are substitutions and (2) all characters in erroneous words are erroneous. Designing more delicate error detection schemes without these two assumptions is possible. First, deletions/insertions may be explicitly identified and handled. Detected insertions can be simply deleted. Each detected deletion can be processed by re-decoding its neighborhood area in the signals to recover what is missing. Second, a Character Verifier (CV) may be utilized to classify each character in erroneous words as either correct or wrong. For the development of CV, only using features that are based on decoding information may be insufficient due to the lack of linguistic constraints. When verifying a character, introducing constraints based on the context of the character in focus may be a necessity.

9.3.2 Improving Error Correction

The proposed error correction procedure can be improved mainly in four aspects. First, the candidate creation approach can be refined to increase the reference coverage rate (RCR) of candidate lists and/or to enhance efficiency. Given a certain list size (i.e., the number of alternatives in a candidate list), the higher the RCR, the better the correction effect. Both the

efficiency of candidate creation and the size of resulting candidate lists will impact the overall efficiency of the error correction procedure. Second, more/better knowledge sources can be adopted in the procedure of linguistic scoring. The power to correct recognition errors mainly comes from the knowledge models utilized. Thus, introducing models that can effectively distinguish between the correct hypothesis and the competing ones is critical to improve the error correction effect. Third, efficient algorithms that search for the top-scoring hypotheses in search networks can be designed. The current search method is slow, especially for utterances with multiple errors. This also limits the application scope of the error correction procedure to a subset of recognized utterances due to the efficiency problem. Adopting more efficient search algorithms will benefit not only the system efficiency, but also the correction effect. Finally, better mechanisms can be used to minimize the influence of error-detection false alarms on error correction. In the rest of this subsection, we will discuss the above four aspects in detail and also briefly mention other possibilities that may improve error correction.

- **Candidate creation**

For a detected character error, the current algorithm of candidate creation selects those character hypotheses having a similar time period from recognition lattices to form the candidate list. However in the recognition lattices, the starting and ending times of character hypotheses are unknown and thus roughly estimated based on a simplifying assumption. Both the effectiveness and efficiency of candidate creation would be improved if accurate times of character hypotheses are available. This can be realized by modifying the decoder to annotate times for characters in recognition lattices. It is also possible to perform a forced alignment to estimate the time information for characters. However, this procedure can be computationally expensive.

Novel candidate creation algorithms that do not rely on recognition lattices may be used. For example, the candidate list for a character error can be obtained by re-decoding the corresponding speech segment using a character-based recognizer (i.e., a recognizer whose recognition network is a list of all Chinese characters and contains no LM probabilities). The *N*-best character hypotheses generated by this recognizer can be utilized as the candidates for the error under consideration. In this way, a higher RCR may be achieved, since those references that are not included in recognition lattices may be recovered by this re-decoding procedure. Note that a reference character with high acoustic likelihood may be pruned during the baseline

decoding due to the application of language model. Since the character-based recognizer has a simple structure, the re-decoding based candidate creation method would be efficient. The created candidate lists should be pruned to a suitable size to strike a balance between the efficiency of subsequent processing and the RCR.

- **Knowledge usage**

Adopting more powerful knowledge sources in linguistic scoring may further increase the error correction rate. Among the three linguistic knowledge sources utilized in the proposed error correction procedure, only mutual information captures long-distance constraints. Word trigrams and POS trigrams model local semantic and syntactic constraints. Incorporating other high-level knowledge sources, such as long-distance syntactic structure [18, 104] and semantic document information [7, 144], in error correction may be helpful. Besides, acoustic similarity may also be a usable source to distinguish among competing hypotheses.

In the current framework, we interpolate various knowledge sources linearly. Using more advanced combination algorithms, such as the maximum entropy approach [68, 107], may further enhance the ability to correct errors.

- **Search algorithm**

The task of search in error correction is to identify the utterance hypothesis that is scored highest by the advanced linguistic model for each sausage-type search space. The current search algorithm simply enumerates all utterance hypotheses in the search spaces. As discussed in Section 5.4, this method is inefficient for utterances with multiple errors. For those seriously erroneous utterances, inefficiency makes the error correction procedure unfeasible. Designing an efficient lattice search algorithm is thus important. When the search speed becomes sufficiently high, the application scope of the error correction procedure can be extended to all erroneous utterances. This will further enhance the effectiveness of the ED-EC framework.

The major difficulty in the design of efficient lattice search algorithms lies in the segmentation problem for Chinese. A candidate character may belong to various words in different utterance hypotheses in a sausage-type search space. For the example of Figure 3.1, the candidate character 会 belongs to the word 拜会 in the utterance hypothesis 在/新闻/中心/**拜会**/议长, but belongs to the word, 会议, in the utterance 在/新闻/中心/白/**会议**/长. Note that

effective language models are normally word-based. This segmentation problem makes it controversial to assign a unique LM likelihood to each candidate character. Without a unique likelihood stored in each node (i.e., each candidate character) in the search space, existing efficient lattice search methods are inapplicable. This problem may be solved by calculating a unique likelihood for each candidate based on some rules (e.g., selecting the highest one among the LM likelihoods of the words that contain the candidate in focus). Designing new lattice search algorithms to circumvent this problem is also possible.

- **Method to handle error-detection false alarms**

Experiments showed that the special mechanism introduced to handle false alarms of error detection brings a non-negligible improvement. It substantially reduces the rate that false errors are "corrected" into new misrecognitions, as was shown in Section 5.1.3. This special mechanism utilizes two thresholds to accept/reject the error correction results based on the confidence scores of error detection. It may be helpful if we adopt more training data to estimate a smooth threshold function of confidence score. Designing new algorithms to reduce the influence of false alarms is also possible.

The discussions above demonstrate the possible ways to improve error correction under the proposed error correction scheme. Adopting a totally different error correction scheme is another option. For instance, it is possible to change an *N*-best re-ranking technique into an error correction procedure. *N*-best re-ranking scores the *N*-best hypotheses using some method and output the top-ranking hypotheses as the new result. When adapting *N*-best re-ranking to an error correction method, we can first identify modifications made by *N*-best re-ranking by comparing the new results with the original recognition results. On the erroneous regions detected by error detection, all modifications can be viewed as the error correction results. On the remaining regions deemed to be correct, modifications can simply be discarded. Using such an error correction procedure, the ED-EC framework will outperform the original *N*-best re-ranking approach if the error detection performance is good enough. If error detection is perfect, all incorrect modifications can be avoided.

9.3.3 Adapting to Other Languages

The basic idea of the ED-EC framework is language independent. This is because, theoretically, LVCSR for whatever language can be improved by detecting and correcting recognition errors. However the detailed implementation of the framework for a specific language should consider the characteristics of the language. For example, characters are linguistically meaningful for Chinese, and most recognition errors for Mandarin LVCSR are character substitutions. We thus focus the Mandarin ED-EC prototype on detecting/correcting character substitutions. For English, the word is the smallest linguistic unit and insertions/deletions cannot be ignored for LVCSR. An ED-EC framework for English LVCSR needs to handle word insertions and deletions as well as word substitutions.

It is possible to adapt the proposed Mandarin ED-EC prototype to other languages by modifying the component algorithms. We use the adaptation to English LVCSR as an example. For English LVCSR, between two correctly recognized words, insertions/deletions of short words may happen. A more common type of misrecognition is to transcribe a word sequence that contains one or more words into another word sequence that may have a different number of words. For instance, "*Toledo*" may be recognized as "*to leave*", which has two words. If focusing on this type of misrecognition, the current ED-EC prototype can be adapted to English LVCSR by (1) simplifying the error detection procedure, and (2) slightly modifying the candidate creation algorithm along with the mechanism to handle false alarms in error correction. For error detection, those erroneous words detected by the word verifier can be directly passed to the subsequent error correction procedure. The modifications needed for error correction mainly lie in candidate creation. Instead of creating a candidate list for each erroneous character, a candidate list should be created for each erroneous region (i.e., each sequence of erroneous words). Candidate alternatives can still be selected from recognition lattices generated during decoding. Those word sequence hypotheses having the same starting and ending times with the erroneous region in focus can be included into the candidate list as alternatives. With candidate lists created, the subsequent processes of constructing search networks and scoring/ranking utterance hypotheses can remain the same. The special mechanism to handle error-detection false alarms has to be adjusted correspondingly. A confidence score can be assigned to each erroneous region based on the confidence scores of the component words by some approach.

Then, the threshold function of confident score can be re-estimated to accept/reject the correction results for erroneous regions.

9.3.4 Incorporation in Spoken Dialogue Systems

Spoken dialogue systems (SDSs) provide real-time interfaces that allow users to *speak* to computers for various tasks (e.g., air traffic information retrieval) [44, 112, 139]. Speech recognition is one important component of an SDS because (1) high recognition accuracy is the basis for the subsequent processing (i.e., language understanding, dialogue management) and (2) the recognition efficiency heavily impacts the overall system efficiency. The ED-EC framework is especially suitable to be applied in SDSs. The framework increases the recognition accuracy while maintaining the efficiency as much as possible. In addition, the interaction between users and computers can be an additional information source to facilitate error detection. For example, a multi-modal dialogue system can display the suspicious areas in recognized utterances on the screen and let users confirm which of these areas are indeed erroneous through clicking the mouse or circling with an electronic pen. Such interactions can improve error detection and eventually greatly enhance the effectiveness of the ED-EC framework. Although these interactions increase the user load, they tend to be acceptable. When people hear something difficult and are not sure whether they hear it correctly, they may repeat what they heard and ask for confirmation. For a dialogue system, users may also tolerate this kind of behavior from machines. We only need to design natural system actions to simulate the communications between human-beings.

9.4 Future Directions for Discriminative Language Modeling

This thesis proposes an algorithm for recasting a discriminative n-gram model as a pseudo-conventional n-gram model. In this work, this pseudo model is used to rescore recognition lattices that are generated during decoding. It is also possible to incorporate this pseudo model into a single-pass decoding procedure. There are two ways to implement such discriminative decoding. First, we can build a complete pseudo-conventional n-gram model, replace the

original language model of the baseline recognizer with this pseudo model and then simply perform decoding using the modified recognizer. Second, the recognizer may adopt an additional mechanism to compute pseudo n-grams online during decoding. The pseudo n-grams may be calculated online for decoding in a similar way as that for discriminative lattice rescoring. In addition, since discriminative training has been shown to be sensitive to domains, selectively applying the pseudo n-grams to certain domains should be beneficial. This selectiveness may be realized by selectively turning this additional mechanism on or off based on the domain information.

Bibliography

[1] M. Afify, F. Liu, H. Jiang and O. Siohan, "A new verification-based fast-match for large vocabulary continuous speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 13, no. 4, pp. 546-553, 2005.

[2] C. Allauzen, M. Mohri and B. Roark, "Generalized algorithms for constructing statistical language models," in *Proc. ACL'03*, Sapporo, Japan, 2003.

[3] B. S. Atal and S. L. Hanauer, "Speech analysis and synthesis by linear prediction of the speech wave," *The Journal of the Acoustical Society of America*, vol. 50, no. 2, pp. 637-655, Aug. 1971.

[4] J. K. Baker, "The DRAGON system: An overview," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 23, no. 1, pp. 24-29, 1975.

[5] M. Balakrishna, D. Moldovan and E. K. Cave, "N-Best list reranking using higher level phonetic, lexical, syntactic and semantic knowledge," in *Proc. ICASSP'06*, Toulouse, France, 2006.

[6] J. R. Bellegarda, "Exploiting both local and global constraints for multi-span statistical language modeling," in *Proc. IPCSSP'98*, Cupertino, CA, USA, 1998.

[7] J. R. Bellegarda, "Exploiting latent semantic information in statistical language modeling," *Proceedings of IEEE*, vol. 88, no. 8, pp. 1279-1296, Aug. 2000.

[8] A. Ben-Yishai and D. Burshtein, "A discriminative training algorithm for hidden markov models," *IEEE Trans. on Speech and Audio Processing*, vol. 12, no. 3, pp.204-217, 2004.

[9] J. Bilmes, "Buried Markov models: a graphical modeling approach for automatic speech recognition," *Computer Speech and Language*, vol. 17, no. 2, pp. 213-231, 2003.

[10] J. A. Bilmes, G. Zweig, T. Richardson, K. Filali, K. Livescu, P. Xu, K. Jackson, Y. Brandman, E. Sandness, E. Holtz, J. Torres and B. Byrne, "Discriminatively structured graphical models for speech recognition," *Report of the JHU 2001 Summer Workshop*, 2001.

[11] E. Bocchieri and S. Parthasarathy, "Rejection using rank statistics based on HMM state shortlists," in *Proc. ICASSP'05*, Philadelphia, PA, USA, 2005.

[12] R. Bod, "Combining semantic and syntactic structure for language modeling", in *Proc. ICSLP'00*, Beijing, China, 2000.

[13] A. Cardenal-Lopez, P. Dieguez-Tirado and C. Garcia-Mateo, "Fast LM look-ahead for large vocabulary continuous speech recognition using perfect hashing," in *Proc. ICASSP'02*, Orlando, FL, USA, 2002.

[14] P. Carpenter, C. Jin, D. Wilson, R. Zhang, D. Bohus and A. Rudnicky, "Is this conversation on track?" in *Proc. Eurospeech'01*, Aalborg, Denmark, 2001.

[15] E. Chang, Y. Shi, J. Zhou and C. Huang, "Speech lab in a box: A Mandarin speech toolbox to jumpstart speech related research," in *Proc. Eurospeech'01*, Aalborg, Denmark, 2001.

[16] L. Chase, "Word and acoustic confidence annotation for large vocabulary speech recognition," in *Proc. Eurospeech'97*, Rhodes, Greece, 1997.

[17] C. Chelba, D. Engle, F. Jelinek, V. Jimenez, S. Khudanpur, L. Mangu, H. Printz, E. S. Ristad, R. Rosenfeld, A. Stolcke and D. Wu. "Structure and performance of a dependency language model," in *Proc. Eurospeech'97*, Rhodes, Greece, 1997.

[18] C. Chelba and F. Jelinek, "Structured language modeling," *Computer Speech and Language*, vol. 14, no. 4, pp. 283-332, Oct. 2000.

[19] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proc. ACL'96*, Santa Cruz, USA, 1996.

[20] Z. Chen, K. F. Lee and M. J. Li, "Discriminative training on language model," in *Proc. ICSLP'02*, Denver, Colorado, USA, 2002.

[21] J. T. Chien, M. Wu and H. Peng, "Latent semantic language modeling and smoothing," *International Journal of Computational Linguistics and Chinese Language Processing,* vol. 9, no. 2, pp. 29-44, August, 2004.

[22] P. Clarkson and R. Rosenfeld, "Statistical language modeling using the CMU-Cambridge toolkit," in *Proc. Eurospeech'97*, Rhodes, Greece, 1997.

[23] M. Collins, "Discriminative training methods for Hidden Markov Models: theory and experiments with the perceptron algorithm," in *EMNLP'02*, Philadelphia, PA, USA, 2002.

[24] M. Collins, B. Roark and M. Saraclar, "Discriminative syntactic language modeling for speech recognition," in *Proc. ACL'05*, AnnArbor, Michigan, USA, 2005.

[25] S. Cox and S. Dasmahapatra, "High-level approaches to confidence estimation in speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 10, no. 7, pp. 460-471, 2002.

[26] K. H. Davis, R. Biddulph and S. Balashek, "Automatic recognition of spoken digits," *The Journal of the Acoustical Society of America*, vol. 24, no. 6, pp. 627-642, 1952.

[27] H. Dudley and S. Balashek, "Automatic recognition of phonetic patterns in speech," *The Journal of the Acoustical Society of America*, vol. 30, no. 8, pp. 721-739, Aug. 1958.

[28] G. Evermann, H. Y. Chan, M. J. F. Gales, T. Hain, X. Liu, D. Mrva, L. Wang and P. C. Woodl, "Development of the 2003 CU-HTK conversational telephone speech transcription system," in *Proc. ICASSP'04*, Montreal, Canada, 2004.

[29] G. Evermann and P. Woodland, "Posterior probability decoding, confidence estimation and system combination," in *NIST Speech Transcription Workshop*, College Park, MD, USA, 2000.

[30] M. Finke, T. Zeppenfeld, M. Maier, L. Mayfield, K. Ries, P. Zhan, J. Lafferty and A. Waibel, "Switchboard April 1996 Evaluation Report," in *Proc. LVCSR Hub 5 Workshop*, Linthicum Heights, Maryland, April 1996.

[31] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (ROVER)," in *Proc. IEEE ASRU Workshop*, Santa Barbara, 1997.

[32] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm," in *Machine Learning*, vol. 37, no. 3, pp. 277-296, 1999.

[33] D. B. Fry and P. Denes, *The Design and Operation of the Mechanical Speech Recognizer at University College London*, J. British Inst. Radio Engr., vol. 19, no. 4, pp. 211-229, 1959.

[34] Y. S. Fu, Y. Pan and L. Lee, "Improved large vocabulary continuous Chinese speech recognition by character-based consensus networks," in *Proc. ISCSLP'06*, Singapore, 2006.

[35] M. J. F. Gales, "Semi-tied covariance matrices for hidden Markov models," *IEEE Trans. on Speech and Audio Processing*, vol. 7, no. 3, pp. 272-281, 1999.

[36] M. J. F. Gales and M. I. Layton, "SVMs, score-spaces and maximum margin statistical models," in *Beyond HMM workshop, ATR*, 2004

[37] M. Gales and S. Young, "The theory of segmental hidden Markov models," *Tech. Rep. CUED/F-INFENG/TR.133*, Cambridge University, 1993.

[38] J. Gao, J. Goodman and J. Miao, "The use of clustering techniques for language modeling - application to Asian language," *Computational Linguistics and Chinese Language Processing*, vol. 6, no. 1, pp. 27-60, 2001.

[39] J. Gao, H. Yu, W. Yuan and P. Xu, "Minimum sample risk methods for language modeling," in *Proc. HLT/EMNLP*, Vancouver, B.C., Canada, 2005.

[40] J. L. Gauvain and H. Schwenk, "Neural Network Language Models for Conversational Speech Recognition," in *Proc. ICSLP'04*, Jeju Island, Korea, 2004.

[41] J. L. Gauvain and H. Schwenk, "Using neural network language models for LVCSR," in *Proc. DARPA RT04*, Palisades, NY, November 2004.

[42] D. Gildea and T. Hoffman, "Topic-based language models using EM," in *Proc. Eurospeech'99*, Budapest, Hungary, 1999.

[43] L. Gillick, Y. Ito and J. Young, "A probabilistic approach to confidence measure estimation and evaluation," in *Proc. ICASSP'97*, Munich, Germany, April 1997.

[44] J. Glass, "Challenges for spoken dialogue systems," in *Proc. ASRU*, Keystone, CO, December 1999.

[45] A. R. Golding and Y. Schabes, "Combining trigram-based and feature-based methods for context-sensitive spelling correction," in *Proc. ACL'96*, Santa Cruz, CA, USA, 1996.

[46] J. T. Goodman, "A bit of progress in language modeling," *Computer Speech and Language*, vol. 15, no. 4, pp. 403-434, 2001.

[47] G. Guo, C. Huang, H. Jiang and R. Wang, "A comparative study on various confidence measures in large vocabulary speech recognition," in *Proc. ISCSLP'04*, Hong Kong, 2004.

[48] K. Hacioglu and W. Ward, "A concept graph based confidence measure," in *Proc. ICASSP'02*, Orlando, FL, USA, 2002.

[49] T. Hain, P. C. Woodland, G. Evermann and D. Povey, "The CU-HTK March 2000 Hub5E transcription system," in *NIST Speech Transcription Workshop*, College Park, MD, USA, 2000.

[50] M. P. Harper and R. A. Helzerman, "Extensions to constraint dependency parsing for spoken language processing," *Computer Speech and Language*, vol. 9, no. 3, pp. 187-234, 1995.

[51] T. J. Hazen, S. Seneff and J. Polifroni, "Recognition confidence scoring and its use in speech understanding systems," *Computer Speech and Language*, vol. 16, no. 1, pp. 49-67, 2002.

[52] P. A. Heeman, "POS tagging versus classes in language modeling," in *Proc. the 6th Workshop on Very Large Corpora*, Montreal, Canada, 1998.

[53] W. Holmes and M. Russel, "Probabilistic-trajectory segmental HMMs," *Computer Speech and Language*, vol. 13, no. 1, pp. 3-37, 1999.

[54] J. Huang, V. Goel, R. Gopinath, B. Kingsbury, P. Olsen and K. Visweswariah, "Large vocabulary conversational speech recognition with the extended maximum likelihood linear transformation (EMLLT) model," in *Proc. ICSLP'02*, Denver, Colorado, USA, 2002.

[55] X. Huang, A. Acero and H. W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm and System Development*, Prentice Hall, ISBN: 0-13-022616-5, 2001.

[56] X. Huang, F. Alleva, H. -W. Hon, M. -Y. Hwang, K. -F. Lee and R. Rosenfeld, "The SPHINX-II speech recognition system: An overview," *Computer Speech and Language*, vol. 7, no. 2, pp. 137-148, 1993.

[57] H. Huning, M. Kirschner, F. Class, A. Berton and U. Haiber, "Embedding grammars into statistical language models," in *Proc. Interspeech'05*, Lissabon, Portugal, September 2005.

[58] F. Itakura and S. Saito, "A statistical method for estimation of speech spectral density and formant frequencies," *Electronics and Communications in Japan*, vol. 53A, pp. 36-43, 1970.

[59] R. Iyer and M. Ostendorf, "Modeling long distance dependence in language: Topic mixtures versus dynamic cache models," *IEEE Trans. on Acoustics, Speech and Audio Processing*, vol. 7, pp. 30-39, January. 1999.

[60] F. Jelinek, "Self-organized language modeling for speech recognition," in *Readings in Speech Recognition*, pp. 450-505, Morgan Kaufman Publishers, Inc., San Mateo, CA, USA, 1990.

[61] M. Jeong, S. Jung and G. G. Lee, "Speech recognition error correction using maximum entropy language model, " in *Proc. ICSLP'04*, Jeju, Korea, Oct 2004.

[62] M. Jeong and G. G. Lee, "Experiments on error corrective language model adaptation," in *Proc. of 9th Western Pacific Acoustics Conference (Westpac IX 2006)*, Seoul, Korea, June 2006.

[63] H. Jiang and C. H. Lee, "A new approach to utterance verification based on neighborhood information in model space," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 5, pp. 425-434, Sept 2003.

[64] M. P. Jones and J. H. Martin, "Contextual spelling correction using latent semantic analysis," in *Proc. of the Fifth Conference on Applied Natural Language Processing*, Washington, D.C., USA, 1997.

[65] B. -H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol.40, no. 12, pp. 3043-3054, Dec. 1992.

[66] S. Kamppari, *Word and phone level acoustic confidence scoring for speech understanding systems*. Master's Thesis, MIT, 1999.

[67] T. Kawabata and K. Shikano, "Island-driven continuous speech recognizer using phone-based HMM word spotting," in *Proc. ICASSP'89*, Glasgow, UK, May 1989.

[68] S. Khudanpur and J. Wu, "Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling, " *Computer Speech and Language*, vol. 14, no. 4, pp. 355-372, 2000.

[69] H. -K. Kuo, E. Fosler-Lussier, H. Jiang and C. -H. Lee, "Discriminative training of language models for speech recognition," *International Conference on Acoustics, Speech, and Signal Processing*, Orlando, Florida, 2002.

[70] H. Kuo, G. Zweig and B. Kingsbury, "Discriminative training of decoding graphs for large vocabulary continuous speech recognition," in *Proc. ICASSP'07*, Honolulu, Hawai'i, USA, 2007.

[71] J. W. Kuo and B. Chen, "Minimum word error based discriminative training of language models," in *Proc. Eurospeech'05*, Lisbon, Portugal, 2005.

[72] M. I. Layton and M. J. F. Gales, "Augmented Statistical Models for Speech Recognition," in *Proc. ICASSP'05*, Philadelphia, PA, USA, 2005.

[73] L. Lee, H. Attias, L. Deng and P. Fieguth, "A multimodal variational approach to learning and inference in switching state space models," in *Proc. ICASSP'04*, Montreal, Canada, 2004.

[74] K. Y. Leung and M. Siu, "Articulatory-feature-based confidence measures," *Computer Speech and Language*, vol. 20, no. 4, pp. 542-562, 2006.

[75] X. Li and Y. Zhao, "A fast and memory-efficient N-gram language model lookup method for large vocabulary continuous speech recognition," *Computer Speech and Language*, vol. 21, iss. 1, pp. 1-25, Jan 2007.

[76] S. S. Lin and F. Yvon, "Discriminative training of finite-state decoding graphs," in *Proc. Interspeech'05*, Lissabon, Portugal, September 2005.

[77] W. K. Lo and F. K. Soong, "Generalized posterior probability for minimum error verification of recognized sentences," in *Proc. ICASSP'05*, Philadelphia, PA, USA, 2005.

[78] B. Lowerre, *The HARPY Speech Understanding System*, *Trends in Speech Recognition*, W. Lea, Editor, Speech Science Publications, 1986, reprinted in Readings in Speech Recognition, A. Waibel and K. F. Lee, Editors, pp. 576-586, Morgan Kaufmann Publishers, 1990.

[79] C. Ma, M. Randolph and J. Drish, "A support vector machines-based rejection technique for speech recognition," in *Proc. ICASSP'01*, Salt Lake City, Utah, USA, 2001.

[80] L. Mangu, *Finding Consensus in Speech Recognition* , PhD thesis, 1999.

[81] L. Mangu, E. Brill and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," in *Computer Speech and Language*, vol. 14, no. 4, pp. 373-400, 2000.

[82] L. Mangu and M. Padmanabhan, "Error corrective mechanisms for speech recognition," in *Proc. ICASSP'01*, Salt Lake City, Utah, USA, 2001.

[83] T. B. Martin, A. L. Nelson and H. J. Zadell, "Speech recognition by feature abstraction techniques," *Tech. Report AL-TDR-64-176*, Air Force Avionics Lab, 1964.

[84] S. Martin, C. Hamacher, J. Liermann, F. Wessel and H. Ney, "Assessment of smoothing methods and complex stochastic language modeling," in *6th European Conference on Speech Communication and Technology*, Budapest, Hungary, September 1999.

[85] S. Matsoukas, J. Gauvain, G. Adda, T. Colthurst, C. Kao, O. Kimball, L. Lamel, F. Lefevre, J. Z. Ma, J. Makhoul, L. Nguyen, R. Prasad, R. M. Schwartz, H. Schwenk and B. Xiang, "Advances in transcription of broadcast news and conversational telephone speech within the combined EARS BBN/LIMSI system," *IEEE Trans. on Audio, Speech and Language Processing*， vol. 14, no. 5, pp. 1541-1556, 2006.

[86] E. Mays, F. J. Damerau and R. L. Mercer, "Context based spelling correction," *Information Processing and Management*, vol. 27, no. 5, pp. 517-522, 1991.

[87] E. McDermott, T. J. Hazen, J. L. Roux, A. Nakamura and S. Katagiri, "Discriminative training for large vocabulary speech recognition using minimum classification error," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 15, no. 1, pp. 203-223, January 2007.

[88] H. Meng, B. Chen, S. Khudanpur, G. Levow, W. Lo, D. Oard, P. Schone, K. Tang, H. Wang and J. Wang, "Mandarin-English information (MEI): Investigating translingual speech retrieval," in *Proc. of the Human Language Technology Conference (HLT 2001)*, San Diego, March 2001.

[89] H. Meng and C. W. Ip, "An analytical study of transformational tagging on Chinese text," in *Proc. of the 1999 ROCLING conference*, Taiwan, August 1999.

[90] T. M. Mitchell, *Machine learning*. The McGraw-Hill Companies, Inc., 1997.

[91] R. Morton, D. Whitehouse and D. Ollason, *The HAPI Book*. Cambridge, U.K.: Entropic Cambridge Research Laboratory.

[92] X. Mou, S. Seneff and V. Zue, "Integration of supra-lexical linguistic models with speech recognition using shallow parsing and finite state transducers," in *Proc. ICSLP'02*, Denver, CO, September 2002.

[93] M. Nagata, "Context-Based spelling correction for Japanese OCR," in *Proc. of the 16th conference on Computational linguistics*, Copenhagen, Denmark, 1996.

[94] T. Niesler and P. Woodland, "A variable-length category-based N-gram language model," in *Proc. ICASSP'06*, Toulouse, France, 2006.

[95] M. Ostendorf, V. V. Digalakis and O. A. Kimball, "From HMM's to segment models: A unified view of stochastic modeling for speech recognition," *IEEE Trans. on Speech Audio Processing*, vol. 4, no. 5, pp. 360-378, Sep. 1996.

[96] C. Pao, P. Schmid and J. Glass, "Confidence scoring for speech understanding systems," in *Proc. ICSLP'98*, Sydney, Australia, Dec. 1998.

[97] D. Povey, "SPAM and full covariance for speech recognition," in *Proc. ICSLP'06*, Pittsburgh, Pennsylvania, 2006.

[98] D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, G. Zweig, "fMPE: Discriminatively trained features for speech recognition," RT'04 meeting, 2004.

[99] D. Povey and P. C. Woodland, "Minimum phone error and I-Smoothing for improved discrimative training," in *Proc. ICASSP'02*, Orlando, FL, USA, 2002.

[100] S. S. Pradhan and W. H. Ward, "Estimating semantic confidence for spoken dialogue systems," in *Proc. ICASSP'02*, Orlando, FL, USA, 2002.

[101] J. Razik, O. Mella, D. Fohr and J. P. Haton, "LocalWord confidence measure using word graph and N-best list," in *Proc. Interspeech'05*, Lissabon, Portugal, September 2005.

[102] E. K. Ringger and J. F. Allen, "Error correction via a post-processor for continuous speech recognition," in *Proc. ICASSP'96*, Atlanta, GA, May 1996.

[103] E. K. Ringger and J. F. Allen, "Robust error correction of continuous speech recognition," in *Proc. of the ESCA-NATO Workshop on Robust Speech Recognition for Unknown Communication Channels*, Pont-a-Mousson, France, April 1997.

[104] B. Roark, "Probabilistic top-down parsing and language modeling," *Computational Linguistics*, vol. 27, no. 2, pp. 249-276, 2001.

[105] B. Roark, M. Saraclar and M. Collins, "Corrective language modeling for large vocabulary ASR with the perceptron algorithm," in *Proc. ICASSP'04*, Montreal, Canada, 2004.

[106] B. Roark, M. Saraclar and M. Collins, "Discriminative n-gram language modeling," *Computer Speech and Language*, vol. 21, no. 2, pp. 373-392, 2007.

[107] R. Rosenfeld, "A maximum entropy approach to adaptive statistical language modeling," *Computer Speech and Language*, vol. 10, no. 3, pp. 187-228, 1996.

[108] A. Rosti and M. Gales, "Switching linear dynamical systems for speech recognition," *Tech. Rep. CUED/F-INFENG/TR.461*, Cambridge University, 2003.

[109] A. Sankar, "Bayesian model combination (BAYCOM) for improved recognition," in *Proc. ICASSP'05*, Philadelphia, PA, USA, 2005.

[110] R. Sarikaya, Y. Gao, M. Picheny and H. Erdogan, "Semantic confidence measurement for spoken dialog systems," *IEEE Trans. Speech and Audio Processing*, vol. 13, no. 4, July 2005.

[111] T. Schaaf and T. Kemp, "Confidence measures for spontaneous speech recognition," in *Proc. ICASSP'97*, Munich, Germany, April 1997.

[112] J. Schatzmann, K. Weilhammer, M. Stuttle and S. Young, "A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies," *Knowledge Engineering Review*, vol. 21, no. 2, pp. 97-126, 2006.

[113] H. Schwenk, "Continuous space language models," *Computer Speech and Language*, vol. 21, no. 3, pp. 492-518, 2007.

[114] H. Schwenk and J. -L. Gauvain, "Connectionist language modeling for large vocabulary continuous speech recognition," in *Proc. ICASSP'02*, Orlando, FL, USA, 2002.

[115] F. Seide, "The use of virtual hypothesis copies in decoding of large-vocabulary continuous speech," *IEEE Trans. on Speech and Audio Processing*, vol. 13, no. 4, Jul. 2005.

[116] V. Seneviratne and S. Young, "The hidden vector state language model," in *Proc. Interspeech'05*, Lissabon, Portugal, September 2005.

[117] N. Singh-Miller and C. Collins, "Trigger-based language modeling using a loss-sensitive perceptron algorithm," in *Proc. ICASSP'07*, Honolulu, Hawai'i, USA, 2007.

[118] O. Siohan, B. Ramabhadran and B. Kingsbury, "Constructing ensembles of ASR systems using randomized decision trees," in *Proc. ICASSP'05*, Philadelphia, PA, USA, 2005.

[119] M. Siu and M. Ostendorf, "Variable n-grams and extensions for conversational speech language modeling," *IEEE Trans. on Speech and Audio Processing*, vol. 8, no. 1, pp. 63-75, 2000.

[120] N. D. Smith, *Using Augmented Statistical Models and Score Spaces for Classification*, Ph.D. thesis, University of Cambridge, September 2003.

[121] H. Soltau, F. Metze, C. Fugen and A. Waibel, "Efficient language model lookahead through polymorphic linguistic context assignment," in *Proc. ICASSP'02*, Orlando, FL, USA, 2002.

[122] F. K. Soong, W. K. Lo and S. Nakamura, "Optimal acoustic and language model weights for minimizing word verification errors," in *Proc. ICSLP'04*, Jeju Island, Korea, 2004.

[123] A. Stolcke, C. Chelba, D. Engle, V. Jimenez, L. Mangu, H. Printz, E. Ristad, R. Rosenfeld, D. Wu, F. Jelinek and S. Khudanpur, *WS96 project report: dependency language modeling*, 1997.

[124] A. Stolcke, B. Chen, H. Franco, V. R. R. Gadde, M. Graciarena, M. -Y. Hwang, K. Kirchhoff, N. Morgan, X. Lin, T. Ng, M. Ostendorf, K. Sönmez, A. Venkataraman, D. Vergyri, W. Wang, J. Zheng and Q. Zhu, "Recent innovations in speech-to-text transcription at SRI-ICSI-UW," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1729-1744, September 2006.

[125] D. H. V. Uytsel and D. V. Compernolle, "Language modeling with probabilistic left corner parsing," *Computer Speech and Language*, vol. 19, no. 2, pp. 171-204, 2005.

[126] V. Venkataramani and W. Byrne, "Lattice segmentation and support vector machines for large vocabulary continuous speech recognition," in *Proc. ICASSP'05*, Philadelphia, PA, USA, 2005.

[127] T. K. Vintsyuk, *Speech Discrimination by Dynamic Programming*, Kibernetika, vol. 4, no. 2, pp. 81-88, Jan.-Feb. 1968.

[128] M. D. Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools and D. V. Compernolle, "Template-based continuous speech recognition," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1377-1390 , May 2007.

[129] Y. Wakita, H. Singer and Y. Sagisaka, "Multiple pronunciation dictionary using HMM-state confusion characteristics," *Computer Speech and Language*, vol. 13, no. 2, pp. 143-153, 1999.

[130] L. Wang, M. Gales and P. C. Woodland, "Unsupervised training for Mandarin broadcast news and conversation transcription," in *Proc. ICASSP'07*, Honolulu, Hawai'i, USA, 2007.

[131] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig and A. Stolcke, "Neural-network based measures of confidence for word recognition," in *Proc. ICASSP'97*, Germany, 1997.

[132] A. Wendemuth, G. Rose and J. Dolfing, "Advances in confidence measures for large vocabulary," in *Proc. ICASSP'99*, Phoenix, Arizona, USA, 1999.

[133] F. Wessel, R. Schluter, K. Macherey and H. Ney, "Confidence measures for large vocabulary continuous speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 9, no. 3, pp. 288-298, Mar. 2001.

[134] P. Wiggers and L. J. M. Rothkrantz, "Topic-based language modeling with dynamic bayesian networks," in *Proc. ICSLP'06*, Pittsburgh, Pennsylvania, 2006.

[135] J. Wiren and H. L. Stubbs, "Electronic binary selection system for phoneme classification," *Journal of the Acoustical Society of America*, vol. 28, no. 4, pp. 1082-1091., 1956.

[136] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[137] P. C. Woodland and D Povey, "Large scale discriminative training of hidden markov models for speech recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 25-47, 2002.

[138] S. Young, "Statistical modeling in continuous speech recognition", in *Proc. UAI*, Seattle, 2001.

[139] S. Young, "The statistical approach to the design of spoken dialogue systems." *Tech Report CUED/F-INFENG/TR.433*, Cambridge University Engineering Department, 2002.

[140] H. Zen, K. Tokuda and T. Kitamura, "Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic feature vector sequences," *Computer Speech and Language*, vol. 21, no. 1, pp. 153-173, January 2007.

[141] R. Zhang, E. Black, A. Finch and Y. Sagisaka, "A tagger-aided language model with a stack decoder," in *Proc. ICSLP'00*, Beijing, China, 2000.

[142] R. Zhang, E. Black, A. Finch and Y. Sagisaka, "Integrating detailed information into a language model," in *Proc. ICASSP'00*, Istanbul, Turkey, 2000.

[143] R. Zhang and A. I. Rudnicky, "Word level confidence annotation using combinations of features," in *Proc. Eurospeech'01*, Aalborg, Denmark, 2001.

[144] R. Zhang and A. I. Rudnicky, "Improve latent semantic analysis based language model by integrating multiple level knowledge proceedings," in *Proc. ICSLP'02*, Denver, Colorado, USA, 2002.

[145] R. Zhang and A. I. Rudnicky, "A frame level boosting training scheme for acoustic modeling," in *Proc. ICSLP'04*, Jeju Island, Korea, 2004.

[146] L. Zhou, Y. Shi, J. Feng and A. Sears, "Data mining for detecting errors in dictation speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 13, no. 5, pp. 681-688, Sep. 2005.

[147] Z. Zhou, J. Gao, F. K. Soong and H. Meng, "A comparative study of discriminative methods for reranking LVCSR N-best hypotheses in domain adaptation and generalization," in *Proc. ICASSP'06*, Toulouse, France, 2006.

[148] Z. Zhou and H. Meng, "A two-level schema for detecting recognition errors," in *Proc. ICSLP'04*, Jeju Island, Korea, 2004.

[149] Z. Zhou and H. Meng, "Error identification for large vocabulary speech recognition," in *Proc. ISCSLP'04*, Hong Kong, 2004.

[150] Z. Zhou and H. Meng, "Recasting the discriminative N-gram model as a pseudo-conventional N-gram model for LVCSR," in *Proc. ICASSP'08*, Las Vegas, Nevada, USA, March 2008.

[151] Z. Zhou, H. Meng and W. K. Lo, "A multi-pass error detection and correction framework for Mandarin LVCSR," in *Proc. ICSLP'06*, Pittsburgh, Pennsylvania, 2006.

[152] http://htk.eng.cam.ac.uk/docs/docs.shtml

[153] http://projects.ldc.upenn.edu/Chinese/segmenter/Mandarin.fre

[154] http://research.microsoft.com/srg/srproject.aspx

[155] http://www-306.ibm.com/software/pervasive/embedded_viavoice/

[156] http://www.nuance.com/naturallyspeaking/